현장애로기술개발
최종보고서

시설 토마토 재배 최적환경구현을 위한 자동제어 논리개발

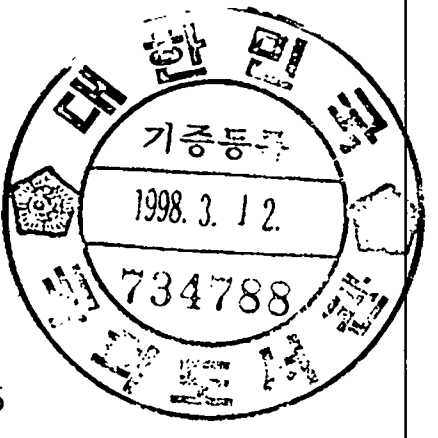
Development of Control Logic for Greenhouse Climate Optimization

연구기관
서울대학교 농업생명과학대학

농 림 부

최종 보고서

'94 - '96 농림수산 현장애로 기술개발사업에 의하여 개발 완료한 "시설 토마토재배 최적환경구현을 위한 자동제어 논리개발"에 관한 최종보고서를 별첨과 같이 제출합니다.



첨부: 1. 최종 연구보고서 8부
2. 최종보고서 디스켓 1매

1 9 9 7. 12. 15

주관 연구 기관 : 서울대학교

총괄연구책임자 : 이 변 우 (인)

주관연구기관장: 서울대학교 총장(직인)

농 림 부 장 관 귀 하

제 출 문

농림부장관 귀하

본 보고서를 "시설 토마토재배 최적환경 구현을 위한 자동제어 논리개발" 과제의 최종보고서로 제출합니다.

1997년 12월 15일

주관연구기관명 : 서울대학교

총괄연구책임자 : 이 변 우

연 구 원 : 남 택수, 신 재훈, 정 실환
하 종력, 이 충근, 숭 병숙
서 미순, 윤 미란, 명 올재
한 상준, 김 시동, 김 수형
이 정양, 정 이호, 김 시정

협동연구기관명 : 농업과 컴퓨터

협동연구책임자 : 조 홍 석

요 약 문

Ⅰ. 제 목

시설 토마토재배 최적환경 구현을 위한 자동제어 논리개발

Ⅱ. 연구개발의 목적 및 중요성

우리 나라의 시설재배 면적은 정부의 적극적인 지원 정책으로 최근 들어 급격한 신장을 보이고 있으나 시설재배작물의 생산성은 선진국에 비하여 매우 낙후되었을 뿐만 아니라 또한 경영비의 과중으로 채산성이 낮다. 생산성과 채산성이 낮은 원인은 온실환경 관리 기술의 낙후로 인하여 작물의 생장과 경영비를 고려한 온실의 최적 환경관리가 이루어지지 못하고 있기 때문으로 생산성과 품질을 증진시키고 난방비 등 경영비를 줄여 수익성을 극대화할 수 있는 과학적인 재배환경 제어 기술의 개발은 매우 중요한 당면과제이다. 본 연구는 시설 작물의 생산성과 수익성 향상을 위한 시설내 복합환경의 최적조건 탐색에 근거한 자동제어 논리를 개발하여 실용화하는데 목적이 있다.

Ⅲ. 연구개발내용 및 범위

본 연구는 시설재배 작물의 생산성과 수익성 향상을 위한 시설내 복합환경 제어 논리의 개발 및 실용화를 최종목표로 하고 있으며 연구개발 내용과 범위는 다음과 같다.

1. 시설재배 토마토의 생육 모델 개발
   - o 온실내 일사량, 온도, 습도, CO_2농도 등을 입력변수로 하는 토마토 생육 모델의 이론적 설계
   - o 생육 모델의 매개변수 정량화 및 검증을 위한 토마토 재배실험 및 환경계측
   - o 모델의 검증 및 복합환경제어에의 응용
2. 온실 미기상 예측모델 개발
   - o 온실 외부의 온도, 습도, 광, 바람 등의 변화에 따른 온실내의 미기상 변화 예측 모델의 이론적 설계
   - o 미기상 모델의 매개변수정량화 검증을 위한 온실 미기상 환경계측
   - o 모델의 검증 및 환경제어에의 응용

3. 시설 최적복합환경 제어시스템 개발

o 최적 복합환경 탐색 및 제어논리 개발

o 최적 복합환경제어시스템 개발 및 실용화

4. 시설 토마토 최적관개모델 개발

o 경직경 미소변화 monitoring에 의한 관개시기 결정방법 개발

o 관개량 결정을 위한 증산모델의 개발

o 최적관개제어 시스템의 개발 및 실용화

Ⅳ. 연구개발 결과 및 활용에 대한 건의

1. 온실내부의 온도, 습도, CO_2, 일사량, 풍속 등을 입력변수로 하여 토마토의 잎, 줄기, 과실 등의 생장과 잎의 출현, 화방과 마디의 형성, 착과 등 발육을 예측할 수 있는 토마토 생육모델의 이론적 설계 및 모델 프로그램을 완료하였고 다양한 조건에서 모델을 검증하였다. 모델검증결과 다양한 온도, 습도, CO_2 환경조건에서 토마토 생장과 발육을 정확하게 예측할 수 있는 것으로 나타났다. 완성된 모델을 온실복합환경 제어 및 최적관개제어 시스템의 submodel로 구성하였다.

2. 온실을 지상부7층 지하부 20층으로 구분하여 각 층에서의 에너지 및 물질 수지를 근간으로 하는 온실 미기상예측 모델의 이론적 설계 및 program을 완성하였으며 모델을 검증한 결과 온실 미기상예측 정확도가 매우 높았다. 완성된 모델을 온실 복합 환경제어 및 최적 관개제어 시스템의 submodel로 구성하였다.

3. 온실 토마토의 생장 및 수익성을 최대로 하기 위한 온실의 온도, 습도, 탄산가스 등의 최적 복합환경 탐색.제어 알고리즘 및 이를 탑재한 복합환경 제어 시스템을 개발하였다 토마토 생장모델과 온실 미기상 모델을 이용하여 외부 환경, 온실 시설 등의 제약조건하에서 토마토의 생장을 최적화 할 수 있는 온실환경의 설정 범위를 탐색하고 인공지능을 이용하여 수익과 제어 비용의 차를 최대화할 수 있도록 온실의 환경을 제어하도록 시스템을 구성하였다.

4. 시설재배 토마토의 최적 관개시기 및 관개량 결정을 위하여 토마토 경직경 미소변화 측정 장치를 스트레인게이지를 이용하여 제작하였으며, 개발된 경직경 측정장치로 식물

체 수분변화에 따른 경직경 미소변화를 정확하게 측정하며 또한 증산 모델은 온실재배 토마토의 증산량을 정확하게 예측하는 것으로 나타났다. 경직경 미소변화에 의한 관개시기 결정 및 증산량예측에 의한 관개량 결정논리를 개발하여 최적관개시스템을 구축하였다.

5. 본 연구에서 개발된 온실 토마토 재배 복합환경제어 시스템과 관개제어 시스템을 상업화하여 농가에 보급할 수 있을 것이다.

6. 본 연구에서 개발된 온실미기상모델, 토마토 생장모델, 온실 복합환경제어모델, 최적관개제어 모델등은 토마토 이외의 타작물의 복합환경 제어시스템의 확대 개발에 응용될 수 있을 것이다.

SUMMARY

Ⅰ. Title

Development of Control Logic for Greenhouse Climate Optimization

Ⅱ. Objectives and Significance

The horticultural crop production in greenhouses have been increasing very rapidly owing much to the financial subsidy policy of the government. However, the farmer's income is not satisfactory because of the low crop productivity and heavy management cost. The low productivity and high cost of management are mainly due to the failure of optimum control of the greenhouse environment of which technologies have been developed poorly. Thus, the development of control logic to optimize the greenhouse environment is needed urgently for the realization of high productivity and cost cut-down in greenhouse crop production.

The Objectives of the project are a) to develop the searching logic for the optimum greenhouse climate to maximize the crop productivity and to minimize the control cost, and 2) to develop the control alogorithm for the realization of optimum greenhouse climate.

Ⅲ. Research Scopes and Perspectives

This research project aims at developing the methods of greenhouse climate optimization for the enhancement of crop productivity and control cost effectiveness, and the control algorithms. The overall objectives of the project are as follows;

- o Development of control logic and system for the optimization of greenhouse climate
- o Development of irrigation control logic and system based on the crop water demand.

Major research scopes are as follows:

1. Development of greenhouse tomato growth model
   - o Developing the theoretical model to simulate the tomato growth and development in the greenhouse.

o Calibrating and validating the model in various environmental conditions.

2. Development of greenhouse microclimate model
   - o Developing the theoretical model to simulate the microclimate by using the outside weather variables as forcing variables
   - o Calibrating and validating the model in various climatic conditions.
3. Development of control logic and system to optimize the greenhouse climate
   - o Designing the algorithm to search the optimal setpoints of climatic variables in the greenhouse
   - o Developing and operationalizing the optimal control system for greenhouse climate.
4. Development of irrigation control model
   - o Developing the decision method of irrigation timing based on stem diameter microvariation
   - o Developing the transpiration model to decide the amount of irrigation
   - o Developing the optimal control logic and system of tomato irrigation

Ⅳ. Results and Suggestions

1. The mechanistic tomato growth and development model were developed, to simulate the growth of stem, leaf, root and fruit, and the development of leaves, leaf area, flowers and fruits by using the weather data inside the greenhouse as forcing variables. A series of experiments for the model parameterization, calibration and validation were conducted during spring to summer season in 1995 and 1996. The model developed were proved to simulate the growth and development of tomato very realistically in various greenhouse climatic condition. The model was incorporated as a component model of the optimal control systems for greenhouse climate and irrigation.

2. The theoretical microclimate model based on the energy and mass balances in the greenhouses, which were divided into 7 above-ground and 20 soil layers, were designed and programmed. The model developed was and calibrated and validated in various climatic conditions. The model was found to perform the realistic simulation of climate inside the greenhouse in various outside climatic conditions. The model

was incorporated as a component model as the optimal control systems for greenhouse climate and irrigation.

3. Algorithm and control system for greenhouse climate optimization were developed. Using mathematical process models to predict tomato growth and greenhouse climate, and neural net model to search the optimal greenhouse climate, the optimal ranges of greenhouse climatic variables which are able to be achieved under the restrictions of outside weather and greenhouse facilities are calculated, and within the predefined ranges the optimal set-points to maximize the benefit are determined and controled.

4. For measuring the microvariation of stem diameter due to crop water status, a measuring device was developed by using strain gauges. The devices were accurate enough to be utilized in measuring the microvariation of the stem diameter. The transpiration model was developed and found to predict the transpiration of greenhouse tomato. The algorithm and system for timely and accurate irrigation were developed based on the microvariation of stem diameter monitoring and transpiration prediction.

5. The systems for optimal greenhouse climate irrigation control are able to be commercialized and to be introduced in farmers' greenhouses.

6. The models and systems which were developed for tomato culture in this project are able to be applied in developing the optimal control systems for other crops.

여 백

목 차

CONTENTS

제 1 장 서 론

제 1 절. 연구개발의 배경 및 필요성

1. 연구의 배경

WTO출범에 따른 농산물 교역의 국제적 자유 경쟁 체제에 대응하기 위하여 농업기반시설 확충, 생산규모의 적정화, 생산시설의 자동화, 생산물의 고품질화, 유통구조의 혁신적 개선 등 당면한 문제들이 산적해 있으며 이를 해결하기 위해서는 보다 체계적이고 과학적인 기술개발이 진행되어야 한다. 시설채소의 경우 생산구조의 개선과 생산성 향상을 통하여 생산원가를 낮추고, 고품질의 상품을 연중 생산할 수 있다면 값싼 해외 상품의 유입을 차단하고 우리의 채소 생산농가를 보호할 가능성이 크다고 할 수 있다. 정부는 채소의 시설재배가 국제경쟁력이 있다고 판단하여 1992년부터 대규모 예산을 편성해 농민의 현대화 시설재배를 지원하고 있으나 이 같은 국가적 시설재배 장려정책 이면에는 무분별한 해외 자재의 수입으로 인한 외화낭비 및 유지보수의 어려움, 최적의 재배조건을 유지하지 못하고 단순한 교과서적 제어만으로 일관함으로 생기는 생산물의 생산성 및 품질의 낙후, 비 효율적 냉난방 관리로 인한 경영비의 압박, 지역기상조건을 고려한 적기 적작 기준의 미비, 토경재배 및 수경재배에 대한 과학적 재배기준 미비, 등 체계적으로 개발되어야 할 기반기술이 산적한 상태이다.

또한 시설재배는 노지재배와는 매우 다른 환경 속에서 작물을 생산하기 때문에 우리의 고유한 기상 환경 조건과 재배환경을 과학적으로 평가·분석하여야 하며, 특히 노동력을 적게 들이고 생산원가를 낮추면서도 생산성의 제고와 생산물의 고품질화를 달성할 수 있는 자동화 재배기술의 선진화가 요구되는 시점이라 할 수 있다.

한편 화란 등 선진 농업국들은 우리의 시설재배자동화기술과 관련 자재산업 후진성을 벗어나지 못한 틈을 이용해 그들의 상품을 본격적으로 국내에 판매하려 안간힘을 쏟고 있다. 그들의 상품은 오랫동안 많은 연구와 개발을 거쳐 나온 상품이기 때문에 우리의 기술에 비해 한 단계 앞섰다는 점은 부인할 수 없으나 우리나라 특유의 기상조건과 품종, 그리고 재배법에 비추어 한계점을 갖고 있어 무분별한 해외상품의 도입보다는 선진기술을 도입·접목하여 우리의 기상조건, 작물, 재배형태에 맞는 우리고유의 기술체계를 확립하는 것이 바람직하다.

2. 연구개발의 필요성

우리나라가 시설 재배 작물의 생산성은 매우 낙후되어 있음. 토마토의 경우 1990년 기준으로 단위면적당 생산량이 시설+노지재배의 경우 연평균 8,606Kg/10a로서 화란의

43,000Kg/10a, 일본의 15,662Kg/10a에 비해 각각 1/5배, 1/1.8배의 생산성에 머무르고 있으며 생산성이 낮은 이유는 작형의 차이, 시설과 관리기술의 낙후 때문이다.

따라서 시설내 최적재배법이 정립되어야 하며, 농가의 시설재배의 경영수지 면에서도 노동력·난방비등 경영비를 최소화하고 수확량과 품질을 증진시켜 최대의 수익을 올릴 수 있는 과학적인 재배환경제어기법의 개발이 필요한 실정이다.

시설토마토의 경우 생산량은 '70년 139.5천톤으로 전체 채소 생산량의 5.3%에 불과하였으나 '92년 1,400천톤으로 16.3%에 달해 날로 그 비중이 높아지고 있다. 그러나, 시설의 환경·재배관리용 시설자재, 환경제어시스템등에 대한 수입의존도가 커서 매년 막대한 량의 와화가 낭비될 뿐 아니라 유지보수나 사용상의 어려움이 있다.

또한 조수입에 대한 경영비가 차지하는 비중이 90년도기준 70.7%에 달해(화란:24.2%) 농가의 경영압박을 가중시키고 있으므로 생산성향상을 위해 생력화된 시설재배면적의 확대와 시설내 과학적 재배환경 자동제어 기술확립이 매우 중요하다.

화란 등 선진 유럽국은 자국의 시설농업기술을 한국·일본을 중심으로 한 동북아 지역을 주요 수출시장으로 삼고 있으며 특히 한국은 범국가적 지원정책에 힘입어 매년 약 500ha이상의 유리온실 재배면적이 증가하는 추세이며 광양의 포항제철 유리온실 시공 등을 계기로 유럽형 온실에 대한 선호도도 높은 편이라 고도의 영업적 전략을 수립해 놓고 추진중인 상태이다. 국내에서는 해외의 제어 기술을 단순히 도입·보급하는 선에서 국내 시설재배 농가를 지원하려는 시도를 하고 있어 우리 기술의 개발이 매우 시급한 실정이다.

제 2 절. 현기술 상태의 취약성

1. 온도 · 습도 조절

우리나라 대부분 온실의 환경제어가 온도에 국한되어 있다. 또한 임의 설정된 최고 · 최저한계치 위주의 단순제어에 머무르고 있는 실정이다. 변온관리가 필요한 과채류 생산에 있어서 4단변온장치 등을 사용하고 있으나 설정온도와 실제온도간에 차이가 있어서 개선이 필요하며 특히 온도계에 의한 Analog방식 제어의 경우 제어기와 센서간의 임계치 범위가 너무 좁아서 개폐장치가 계속 열림 · 닫힘을 반복하여(Hunting 현상) 모터의 고장을 유발하고 있는 상황이다. 한편 선진국에서 활발히 개발 실용화되고 있는 식물생장의 온 · 습도 반응에 근거한 제어 논리의 개발이 전무한 상태이다.

2. CO_2 시비

광조건, 식물생육상태, 환기량 등을 고려한 환경제어 논리의 개발이 되어 있지 않으며 대개의 경우 Timer에 의해 아침 9시-11시 사이에 가동하는 단순제어에 불과하며 CO_2 센서 가격이 고가이고 CO_2 공급효과에 대한 인식부족 등으로 CO_2 제어를 하지 않는 농가도 많은 실정으로서, CO_2시비에 의한 생산성 향상과 CO_2제어에 드는 비용을 고려하여 수익성을 최대화할 수 있는 CO_2시비 제어 논리의 개발이 절실히 요청되고 있다.

3. 관개

온실에서 작물을 재배하는 경우 강우가 차단되어 인위적 관개 시기와 관개량은 생산성과 품질에 지대한 영향을 미치나 우리나라의 경우 이에 대한 기준이 마련되어 있지 않다. 따라서 식물의 수분상태, 토양의 수분상태등과는 무관하게 Timer에 의한 관개를 실시하는 것이 주종을 이루며 특히 관수 개시점과 관개량 결정은 논리적 근거없이 보통 일주일에 1-2회 관개하며 일부 농가는 농민의 경험이나 달관적 조사에 의해 관개를 하고 있는 실정이다.

제 3절 연구 목적과 범위

우리나라의 시설재배 면적은 정부의 적극적인 지원 정책으로 최근 들어 급격한 신장을 보이고 있으나 시설재배작물의 생산성은 선진국에 비하여 매우 낙후되었을 뿐만 아니라 또한 경영비의 과중으로 채산성이 낮다. 생산성과 채산성이 낮은 원인은 온실환경 관리기의 낙후로 인하여 작물의 생장과 경영비를 고려한 온실의 최적환경관리가 이루어지지 못하고 있기 때문으로 생산성과 품질을 증진시키고 난방비 등 경영비를 줄여 수익성을 극대화할 수 있는 과학적인 재배환경 제어 기술의 개발은 매우 중요한 당면과제이다. 본 연구는 시설 작물의 생산성과 수익성 향상을 위한 시설내 복합환경의 최적조건 탐색에 근거한 자동제어 논리를 개발하여 실용화하는데 목적이 있다.

본 연구는 시설재배 작물의 생산성과 수익성 향상을 위한 시설내 복합환경 제어논리의 개발 및 실용화를 최종목표로 하고 있으며 연구범위는 다음과 같다.

1. 시설재배 토마토의 생육 모델 개발
   - o 일사량, 온도, 습도, CO_2농도 등을 입력변수로 하는 토마토 생육모델 설계
   - o 생육 모델의 매개변수 정량화 및 검증을 위한 토마토 재배실험 및 환경계측
   - o 모델의 검증 및 복합환경제어에의 응용
2. 온실 미기상 예측모델 개발
   - o 온실 외부의 온도, 습도, 광, 바람 등의 변화에 따른 온실내의 미기상 변화 예측 모델의 설계
   - o 미기상 모델의 매개변수정량화 검증을 위한 온실 미기상 환경계측
   - o 모델의 검증 및 환경제어에의 응용
3. 시설 최적복합환경 제어모델 개발
   - o 최적 복합환경 탐색 및 제어논리 개발
   - o 최적 복합환경제어 시스템의 개발 및 실용화
4. 시설 토마토 최적관개모델 개발
   - o 경직경 미소변화 monitoring에 의한 관개시기 결정방법 개발
   - o 관개량 결정을 위한 증산모델의 개발
   - o 최적 관개제어 시스템의 개발 및 실용화

제 2장. 토마토 생장모델의 개발

제 1절 서 언

영양, 수분 및 병충해 관리가 적절하게 이루어지는 조건에서 온실작물의 생장과 수량은 주로 작물체 주위 즉 온실내의 온도, 일사량, 습도 등 미기상조건에 의하여 지배를 받는다. 그런데 이들 환경요소가 작물의 생장과 발육에 미치는 영향은 독립적이 아니라 상호복합적으로 영향을 미치기 때문에, 온실의 환경을 작물의 생장과 발육에 최적인 상태로 조절하기 위해서는 이들 상호복합적인 영향을 반영한 작물의 생장모델의 개발은 필수적이다.

작물생장모델은 지난 20여년간 많은 사람들에 의하여 각 작물을 대상으로 개발되어 왔으나(Wit et al, 1970, 1978 ; Spitters et al, 1989 ; Yang et al, 1990 ; Hand et al, 1972), 토마토 생장모델은 Jones등(1991)이 개발한 TOMGRO와 SUCROS87(Spitter et al, 1989)을 근간으로 하여 Bertin등(1993)이 개발한 TOMSIM이 있을 뿐이다. TOMGRO는 토마토의 발육은 매우 상세하게 설계되어 있으나 TOMSIM은 물질생산은 비교적 다양한 조건에서 토마토의 생장을 simulation할 수 있도록 설계되어 있는 반면에 토마토의 발육모델이 없는 것이 단점이다.

본 연구의 최종 목적은 TOMSIM과 TOMGRO의 장점을 취하여 새로운 토마토 생장모델로 구성하고 이를 궁극적으로는 토마토 온실재배에서 온실 환경관리의 최적화 논리개발에 응용하고자 하는 것이다. 그런데 이러한 생장모델들이 온실환경의 최적화 논리개발에 현실적으로 이용될 수 있기 위하여 다양한 환경조건에서 모델의 parameter들이 추정되어야 하며 또한 calibration과 validation과정을 거쳐야 한다. 본 연구에서는 토마토 생장 및 발육 이론 모델을 개발하고 다양한 환경 관리조건에서 토마토 생장 및 발육을 조사하여 모델에 필요한 parameter를 추정하였으며 또한 모델의 검증을 실시하였다.

제 2절 토마토 생장 모델

토마토 생장모델의 종합적 구조를 나타낸 것이 그림 4.1이다. 이 모델은 토양수분, 식물영양, 병충해관리가 최적수준으로 관리되는 조건에서 토마토의 생장 및 발육속도를 지배하는 주요인을 온실의 기온, 습도, 일사량, CO_2농도인 것으로 가정하였다. 이 모델은 일중 시각에 따라 광합성을 계산하고 이를 적산하여 일당 건물축적속도와 각 기관의 sink strength를 계산하여 assimilate pool에 대한 각기관의 sink strength비에 따라서 각 기관으로 assimilate가 배분되도록 하여 각 기관의 생장율을 계산한다. 또한 출엽, 화방의 생성, 과실의`성숙 등 발육속도는 환경 요소로부터 매일 계산한다. 일당 생장 및 발육속도를 생육기간에 따라 적산하여 작물의 생장과 발육을 simulation한다.

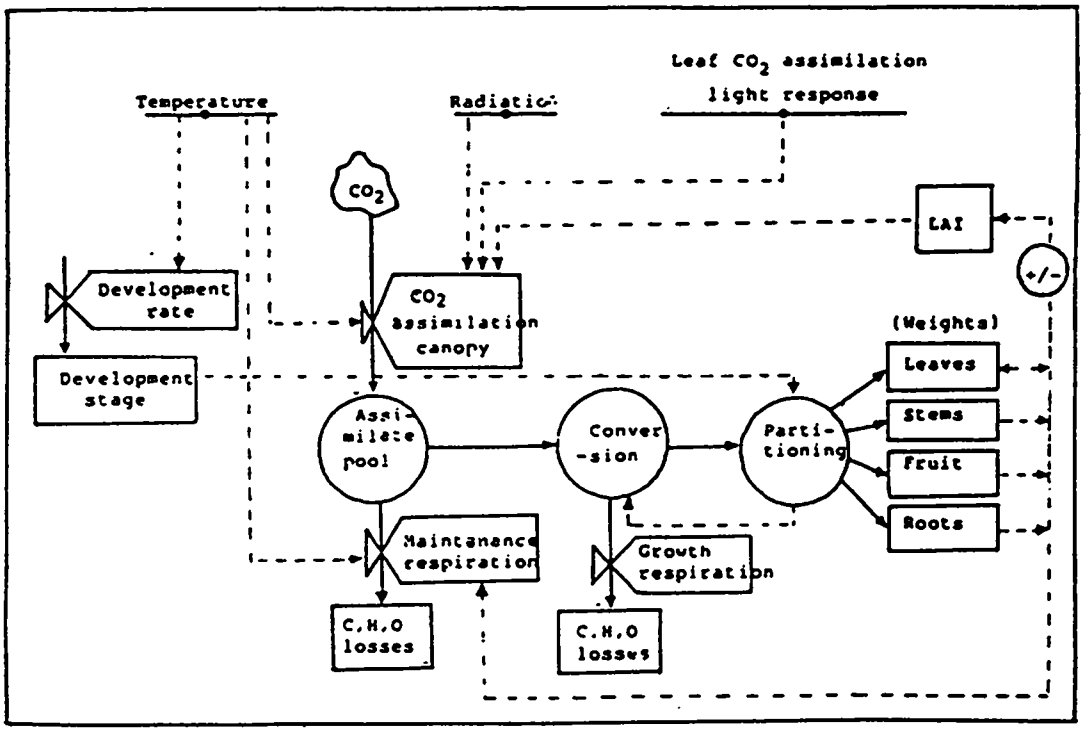


Fig. 2.1 Schematic diagram of tomato growth model. Boxes are state variables, valves are rate variables, circles are intermediate variables. Solid lines are flows of material, dotted lines are flows of information.

1. 군락생장율

토마토 군락의 일당 생장율은 Gijzen(1992)에 따라서 다음과 같이 계산하였으며 각 기관의 일당 생장율은 군락 생장율에 광합성 산물의 각 기관으로 분배계수를 곱하여 계산하였다.

$$\frac{dw}{dt} = GREF*(GP - RMAINT)$$

$\frac{dw}{dt}$: Crop growth rate ($g/m^2/day$)
GREF : Conversion efficiency from assimilate to dry matter
GP : Gross Assimilation rate ($gCH_2O/m^2/day$)
RMAINT: Maintenance respiration rate ($gCH_2O/m^2/day$)

여기서 전환효율(GREF)은 광합성산물의 건물로의 전환비율을 나타내는 것으로서 건물의 화학적 조성에 의하여 결정이 되며 전 식물에의 전환효율은 각 기관의 전환효율과 광합성 산물의 각 기관으로의 배분계수로부터 다음과 같이 계산한다.

$$GREF = PNGP/(ASR_L*PTNLVS + ASR_S*PTNSTM + ASR_F*PTNFRT)$$

ASR : Assimilate requirement (gCH_2O/g dry matter)
PTNGP, PTNLVS, PTNSTM, PTNFRT: potential growth rate of shoot, leaf, stem and fruit, respictively
L, S, F : represent leave, stem, root, and fruit, respectively

전 식물체의 유지 호흡속도는 각 기관의 건물중, 각기관의 유지호흡속도와 유지호흡의 온도의존성으로부터 다음과 같이 계산된다.

$$Rm(T) = (RMR_L*TDM_L + RMR_S*TDM_S + RMR_F*DMG_F)Q_{10}^{(0.1(T-Tr))}$$

RMR : Maintenance respiration rate($gCH_2O/g/day$)
TDM : Organ dry weight(g/m^2)
Tr : Reference temperature
Q_{10} : Temperature coefficient for RMR
L, S, F : represent leave, stem, root, and fruit, respectively

여기서 $Q_{10,c}$는 유지호흡의 온도계수로서 각 기관에 따른 차이는 없는 것으로 가정하였다.

가. 군락의 순간광합성속도

군락내의 엽층 L에서 순간 순광합성속도는 Gourdrian 등(1985)에 따라서 다음과 같이 계산하였다.

$$A_L = A_m(1-\exp(-\varepsilon I_a/A_m))$$

A_m : the gross assimilation rate at light saturation ($gCH_2O/m^2leaf/h$)

ε : the initial light use efficiency ($molCO_2$/mol absorbed photon of PAR)

I_a : the amount of absorbed PAR ($mol/m^2leaf/s$)

여기서 광화학적 이용효율(ε)는 Bertin et al(1993)에 따라서 다음과 같이 계산하였다.

$$\varepsilon = \varepsilon_o(C_a - \Gamma)/(C_a + 2\Gamma)$$

ε_o : 0.084 $molCO_2$/mol photon absorbed
Γ : Carbon dioxide compensation point ($\mu\ell/\ell$)

광보상점은 Brook 등(1985)의 측정에 의하면 엽온의 증가에 따라 증대하며 다음과 같이 온도의존성을 계산하였다.

$$\Gamma = 42.7 + 1.68(T_l-25) + 0.012(T_l-25)^2$$

T_l : leaf temperature

군락의 순간 총광합성속도는 광포화상태에서 CO_2 농도에 따라서 달라지는 최대순광합성속도와 엽온에서의 암호흡속도로부터 다음과 같이 계산하였다.

$$A_m = P_{n,\,c} + R_{d,\,Tl}$$

$P_{n,\,c}$: Maximum net photosynthetic rate limitted by CO_2 (μmol $CO_2/m^2leaf/sec$)

$R_{d,\,Tl}$: Dark respiration rate at leaf temperature (T_l)($\mu molCO_2/m^2leaf/sec$)

광포화상태에서 CO_2농도에 따라서 달라지는 최대순광합성속도는 Goudrian 등(1985)에 따라 다음과 같이 계산하였다.

$$P_{n,\,c} = 41.6(C_a - \Gamma)/(r_m + 1.36r_b + 1.6r_s)$$

r_m : mesophyll resistance(s/m)

r_b : boundary layer resistance(s/m) for water vapor

r_s : stomatal resistance(s/m) for water vapor

여기서 각 저항에 곱해진 상수들은 CO_2와 수증기 확산속도의 각 저항에 대한 비를 말하며, 41.6은 20℃에서 CO_2의 밀도(mmol/ℓ)이다. 엽육저항은 Bertin et al(1993)의 값을 이용하였으며 엽면 경계층저항과 기공저항은 본 연구(제4장의 증산 모델) 에서 유도한 계산식을 이용하였다.

한편 암호흡속도의 온도의존성은 다음과 같이 계산하였다.

$$R_{d,\,T_l} = R_{d,\,20}Q^{(0.1(T_l-20)}$$

$R_{d,\,20}$: Dark respiration at 20℃ leaf temperature(T_l)

Q_{10} : Temperature coefficient for dark respiration

군락의 각 엽층(L, 상층부로부터의 누적엽면적지수)에서의 광흡수는 Monsi 등(1953)에 의하여 다음과 같이 계산하였다.

$$I_{a,\,L} = K(1-\rho)I_o\exp(-KL)$$

K : Light extinction coefficient of canopy

L : Cummulative leaf area index from canopy top

ρ : Canopy reflection coefficient

군락의 흡광계수(K)와 반사율(ρ)는 Goudrian(1977)에 따라 다음과 같이 계산하였다.

*. $K = o.8\sqrt{(1-\sigma)}$, for spherical angle distribution
σ : leaf scattering coefficient (0.2)

*. $\rho = (1-\sqrt{(1-\sigma)})/(1+\sqrt{(1-\sigma)})\cdot[2/(1+2\sin\beta)]$
β : solar elevation

작물 군락에 의한 순간 총광합성속도(A_c)는 각 엽층에서의 순간 총광합성속도(A_L)을 군락의 전체 엽면적지수에 대하여 다음과 같이 계산하였다.

$$A_c = \int_0^{LAI} A_L dL, \qquad LAI : \text{Leaf Area Index}$$

윗 식의 적분을 Gausian Integration Method에 의하여 수치적분하였다.

나. 군락의 일당 총광합성

군락의 일당 총광합성은 순간 총광합성 속도를 일출시각에서 일몰시각까지 적산하여 계산하였다.

$$GP = \int_{sr}^{ss} Ac \, dt$$

ss : sunset hour

sr : sunrise hour

2. 기관의 분화발육

토마토기관의 분화와 발육은 Jones 등(1991)이 개발한 TOMGRO모델을 변형하여 적용하였다. 주간의 마디수, 엽수, 과실수의 가장 어린 age class에서 순변화율은 각 기관의 분화율에서 다음 age class로의 전이율을 빼서 계산하였다. age class는 각 화방이 붙어 있는 마디의 위 한 마디와 아래 두 마디에 속하는 엽 및 과실은 같은 age class에 속하는 것으로 하였다.

$$\frac{dN_s(1)}{dt} = GENR * PLM2 - RDVLVT * n_L * N_s(1)$$

$$\frac{dN_L(1)}{dt} = GENR * PLM2/(1+TPL) - RDVFRT * n_L * N_L(1)$$

$$\frac{dN_F(1)}{dt} = GENR * PLM2 * FPN(PLSTN) * SOSIR - RDVFRT * n_F * N_F(1)$$

$N_s(i)$, $N_L(i)$
$N_F(i)$ = the numers of stems, leaves, and fruit/m², respectively, for age class i

GENR = rate of new node appearanec, no./plant-d

PLM2 = plant density (no./m²)

FPN = fruit initiated per new node depending on PLSTN

PLSTN = total numer of nodes on the plant on a given day

TPL = ratio of new fruiting trusses to new leaves

SOSIR = ratio of carbon supply to demand for the crop

RDVLVT = overall rate of development of leaves at temperature T(1/d)

RDVFRT = rate of development of fruit at temperature T(1/d)

n_L = numer of leaf age classes, 20 in the present model

n_F = number of fruit age classes, 20 in the present model

RDVLT * n_L은 각 age class i에서의 엽수에 대한 다음의 age class i+1로 전이하는 엽수의 비율을 나타낸다.

GENR은 개체당 마디의 분화속도를 나타내는 것으로서 다음과 같이 계산된다.

GENR = GENRAT*GENTEM

GENERAT는 마디 최대분화속도이며, GENTEM은 분화속도의 온도 의존성을 나타내는 계수이다.

특정 엽령의 엽수, $N_L(i)$의 순변화율은 다음식에 의하여 계산된다.

$$\frac{dN_L(i)}{dt} = RDVLVT * (n_L * N_L(i-1) - n_L * N_L(i))$$

마디수와 과실의 발육도도 잎의 발육을 나타내는 윗 식과 기본적으로 동일하게 표현하였다.

각 age class에서 최대 엽면적 신장율, $PLE(i) = \frac{dA_{LP}(i)}{dt}$는 다음과 같이 계산된다.

$$PLE(i) = N_L(i) * POL(i) * GENTEM$$

여기서 POL(i)는 적온에서 age class i에 있는 잎의 신장율을 나타낸다.

3. 각 기관의 생장

가. Sink strength

각 age class에 속하는 경[PGS(i)], 엽[PGL(i)], 과실[PGF(i)]의 잠재 건물생장율(sink strength)은 다음과 같이 계산하였다.

$$PGL(i) = (1 + FRPT) * PLE(i)/SLAMN$$
$$PGS(i) = PGL(i) * FRST$$
$$PGF(i) = N_F(i) * POF(i) * GENTEM$$

여기서 SLAMN은 최소 비엽면적, FRPT는 엽신중에 대한 엽병중의 비율, FRST는 엽중 생장율에 대한 경중 생장율의 비, POF(i)는 적온에서 각 age class에 속하는 과실의 잠재생장율이다.

한편 지상부 전체의 잠재생장율(sink strength, PNGP)는 다음과 같이 계산된다.

$$PNGP = \sum_{i=1}^{n_l} [PGL(i) + PGS(i)] + \sum_{i=1}^{n_f} PGF(i)$$

나. 지상부 전체 및 각 기관의 건물 생장율

토마토가 생장함에 따라 각 발육단계에서는 전체생장량 중 일정비율(PROOT)이 뿌리로 배분 것으로 가정하여 지상부 건물 생장율(RCDRW)은 다음과 같이 계산하였다.

RCDRW = GREF*(GP - RMAINT)*(1-PROOT)

각 age class에 속하는 경[DWS(i)], 엽[DWL(i)], 과실[DWF(i)]의 건물 생장율은 각기관의 잠재생장율과 source/sink비(SOSIR = RCDRW/PNGP)에 의하여 다음과 같이 계산하였다.

DWS(i) = PGS(i)*SOSIR
DWS(i) = PGL(i)*SOSIR
DWS(i) = PGF(i)*SOSIR

각 지상부 기관의 전체 생장율은 각 age class에서의 생장율을 적산하여 계산한다. 한편 각 age class에 속하는 잎의 엽면적 생장율은 다음과 같이 계산하였으며 총 엽면적 생장율은 age class에 속하는 잎의 엽면적 생장율을 적산하여 계산한다.

XLA(i) = PGL(i)*SOSIR*SLAMX/(1+FRPT)

여기서 SLAMX는 최대 비엽면적이다.

제 3 절. 토마토 생장모델의 검증

위에서 설계된 모델을 현실적으로 이용하기 위해서는 다양한 환경조건에서 실험된 토마토 생장 및 발육자료가 수집되어 이를 이용하여 모델의 매개변수들이 정량화되고 calibration과 validation이 되어야 한다. 이와 같은 목적으로 다양한 환경조건에서 토마토의 생장·발육자료를 얻기 위하여 일련의 작기 이동 및 CO_2와 온도 조절실험을 실시하여 모델의 매개변수 추정 및 검증을 하였다.

1. 모델 매개변수 정량화 및 검증을 위한 토마토 재배실험

가. 재배시기 및 처리

1995년 봄 작기와 가을 작기, 1996년 봄 작기에 실험을 수행하였다. 1995년 봄 작기 실험에서는 Momotaro와 Arletta를 공시하였는데 60일간 육묘하여 5월 28일에 재식 거리를 휴간30cm, 25cm(2열)x45cm로 하여 정식하였다. 재배기간 중 측창을 개방한 상태에서 재배하였다. 1995년 가을 작기 실험에서는 서광을 공시하여 9월29일에 재식 거리를 휴간을 60cm로 하고 휴폭을 90cm로 하여 25cm(2열)x45cm로 정식하였다. 한편 야간에는 최저 온도를 11°C로 설정하였으며 주간에는 최고 온도가 25°C 이상 되면 측창을 개방하는 방법으로 온실 환경관리를 하였다. 주간에 CO_2농도가 800ppm(주간 평균)이 되도록 액화 탄산가스로 시비한 처리와 탄산가스를 시용하지 않은 처리를 두었다. 1996년 봄 작기 실험에서는 서광을 공시하여 3월 15일에 재식 거리를 휴간을 60cm로 하고 휴폭을 90cm로 하여 25cm(2열)x45cm로 정식하였다. 한편 야간에는 최저 온도를 15°C로 설정하였으며 주간에는 최고 온도가 30°C 이상 되면 측창을 개방하는 방법으로 온실 환경관리를 하였으며 오전 8시에서 11시 사이에 CO_2농도가 1000ppm이 되도록 액화 탄산가스로 시비한 처리와 탄산가스를 시용하지 않은 처리를 두었다.

나. 재배 관리

관개는 tensiometer를 이용하여 토양수분 장력이 0.2bar 이하로 유지되도록 점적관개 시스템으로 자동 관개하였으며 시비는 질소, 인산, 칼리를 성분량으로 각각 30kg/10a 시용하였는데 인산과 칼리는 전량 기비로 시용하였으며, 질소는 15kg/10a는 기비로 나머지 15kg/10a는 3회에 걸쳐 관비하였다. 토마토는 주간만을 남기고 측지는 나오는 대로 제거하였다.

다. 생장 및 발육조사

정식 후 1주일 간격으로 처리당 4개체씩을 채취하여 엽, 엽병, 경, 과실을 80°C에서 48시간 건조시켜 건물중을 측정하였으며 또한 엽면적 및 수량을 조사하였다. 한편 각 처리당 4개체를 지정하여 동일 개체에 대하여 매주 2회씩 엽수, 마디수, 화방수, 개화수, 착과수, 낙과수 등을 조사하였다.

라. 온실 미기상 측정

온실내의 온습도(각 온실당 3조의 psychrometer 설치), 일사량, 풍속(hotwire anemometer), 토양수분장력(10cm, 20cm, 30cm), 토양온도(5, 10, 20, 30, 50cm), CO_2 농도(Infrared gas analyzer)를 측정하였는데 모든 측정항목은 10초마다 측정하여 10분간의 평균치를 data logger로 集錄하였다.

2. 모델의 매개변수 추정 및 calibration

토마토 생장 simulation에 필요한 parameter들을 토마토 환경반응 실험자료들로 부터 추정하거나 모델 calibration을 통하여 구한 값들을 표 2.1 - 표 2.3에 나타내었다. 표의 약어에 대한 설명은 부록에 수록하였다.

Table 2.1 Constants measured or estimated for simulating growth and yield of tomato, cv. Momotaro, Arletta and Seokwang by the model developed.

Constants	Values for variety,			Unit
	Momotaro	Arletta	Seokwang	
ABORMX	0.73	0.73	0.55	
FRLG	6.0	6.0	4.0	nodes
FRPT	0.590	0.578	0.436	
FRST	0.990	0.973	0.459	
FTRUSN	6.0	6.0	10.0	nodes
GREF	0.75	0.75	0.75	gDM/gCH_2O
GENRAT	0.55	0.55	0.73	nodes/day
KDF	0.68	0.68	0.68	
Q10	2.0	2.0	2.0	
RM	250	250	250	s/m
RMRF	0.01	0.01	0.01	gCH_2O/gDM-day
RMRL	0.03	0.03	0.03	"
RMRS	0.015	0.015	0.015	"
SLAMN	0.0066	0.0066	0.013	m^2/g
SLAMX	0.054	0.054	0.039	m^2/g
TPL	0.333	0.333	0.333	trusses/leaves

* Explainations of the variables refer to the Appendix.

Table 2.2 Values for temperature functions in the tomato growth model developed

Temperature	RDVLVT (1/days)	RDVFRT(1/days)		GENTEM
		Momotaro & Arletta	Seokwang	
9	0.0	0.0	0.0	0.0
12	0.0096			0.55
15	0.0126	0.007	0.009	
18		0.016	0.022	
21	0.0190	0.021	0.028	
24		0.032	0.043	
28	0.0260	0.032	0.043	1.00
38	0.0260	0.020	0.027	
50	0.0	0.0	0.0	0.0

* Values between temterature classes are linearly interpolated
* RDVLVT: the rate of leaf development or aging at temterature T
* RDVFRT: the rate of fruit development or aging at temterature T
* GENTEM: the function to reduce rate of node initiation when T is outside the optimum range, and to reduce the sink strength of growing organs
* RDVLVT and GENTEM are common to three varieties used.

Table 2.3 Values for potential leaf area expansion(POL, cm^2/leaf-day) and potential fruit growth rate(POF, g/fruit_day) as functions of percent development in the tomato growth model developed.

Percent of development	POL		POF	
	Momotaro & Arletta	Seokwang	Momotaro & Arletta	Seokwang
0	0.0	0.0	0.0	0.0
5	3.85	4.37	0.14	0.05
15	18.15	20.61	0.38	0.14
25	28.05	31.86	0.68	0.25
35	25.30	28.74	0.86	0.31
45	17.05	19.35	0.87	0.32
55	9.90	11.24	0.77	0.28
65	5.50	6.25	0.62	0.23
75	2.75	3.12	0.47	0.17
85	1.68	1.87	0.33	0.12
95	0.55	0.57	0.17	0.06
100	0.0	0.0	0.0	0.0

3. 모델의 검증

실측한 온실의 기온, 습도, 일사량, CO_2농도를 입력변수로하고 실험자료로부터 추정한 공시 품종들의 매개변수값(표 2.1 - 2.3)을 이용하여 토마토 생장 및 발육을 simulation하여 실측치와 비교검토 하였다.

가. 토마토 발육

다음의 그림2.2 - 2.11에서 보는 바와 같이 토마토 생장 모델은 마디(그림2.2 - 2.4), 엽(그림2.5 - 2.7), 화방(그림2.8 - 2.10), 과실(그림2.11)의 분화 발육을 다양한 환경 조건에서 정확하게 simulation할 수 있는 것으로 나타났다.

(1) 마디수

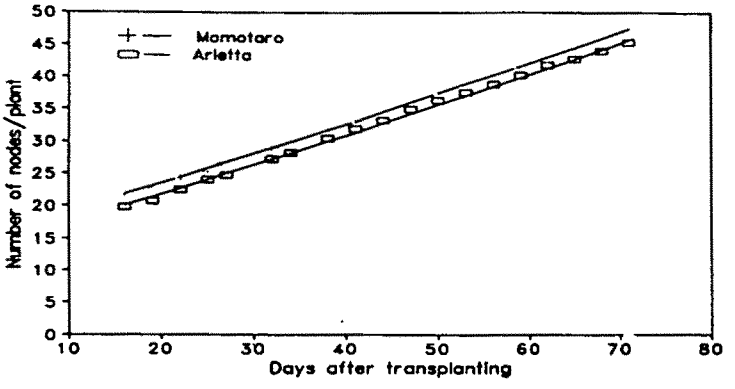


Fig.2.2 Comparison of measured and simulated number of nodes per plant in two tomato cultivars, Momotaro and Arletta in spring season experiments, 1995.(line: simulated, symbol: measured)

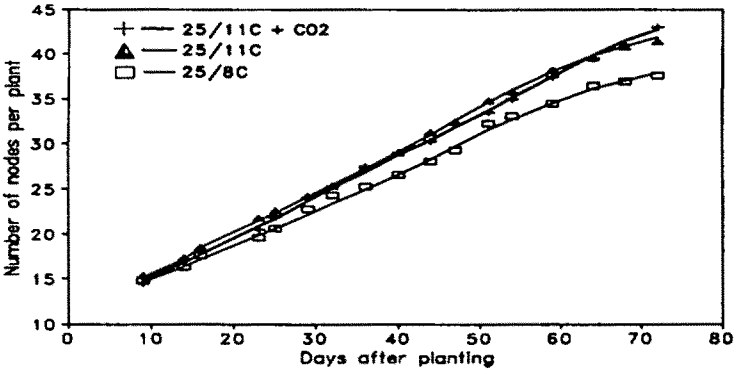


Fig.2.3 Comparison of measured and simulated number of nodes per plant in cv. Seokwang at different temperature and CO_2(800ppm) enriched conditions in autumn season experiments,1995.

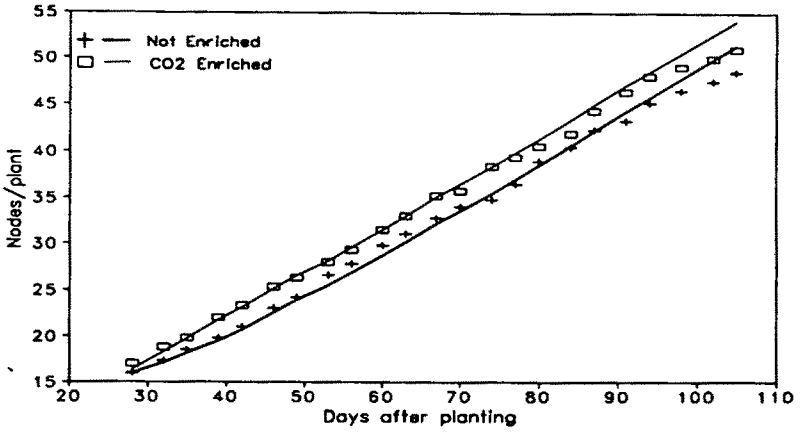


Fig.2.4 Comparison of measured and simulated number of nodes per plant in cv. Seokwang at natural and CO_2(1,000ppm) enriched conditions in spring season experiments,1996.

(2) 엽 수

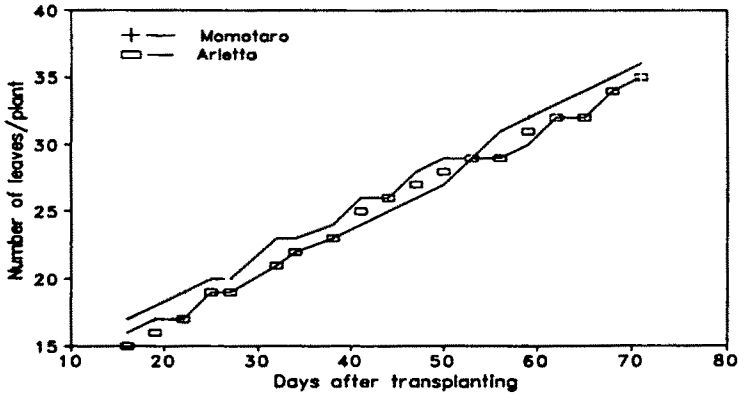


Fig.2.5 Comparison of measured and simulated number of leaves per plant in two tomato cultivars, Momotaro and Arletta in spring season experiments, 1995.(line: simulated, symbol: measured)

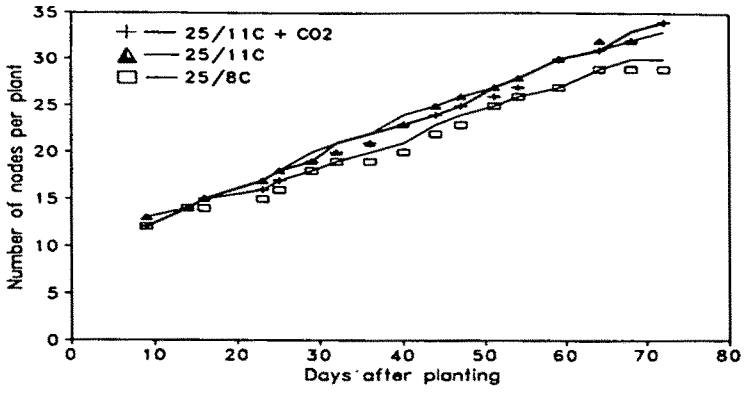


Fig.2.6 Comparison of measured and simulated number of leaves per plant in cv. Seokwang at different temperature and CO_2(800ppm) enriched conditions in autumn season experiments, 1995.

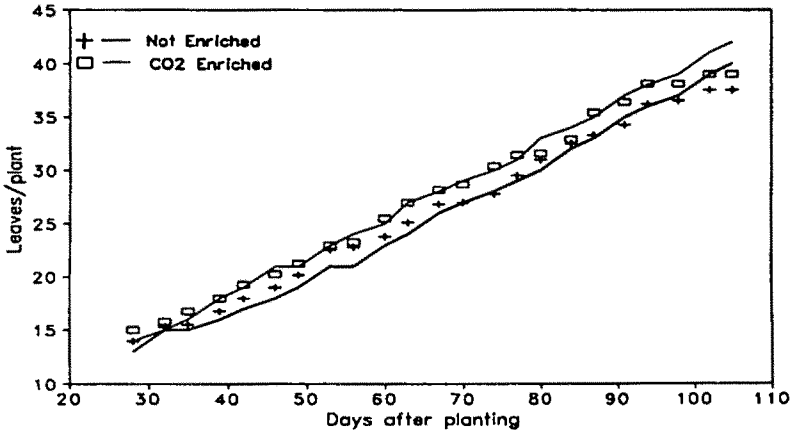


Fig.2.7 Comparison of measured and simulated number of leaves per plant in cv. Seokwang at natural and CO_2(1,000ppm) enriched conditions in spring season experiments, 1996.

(3) 화방수

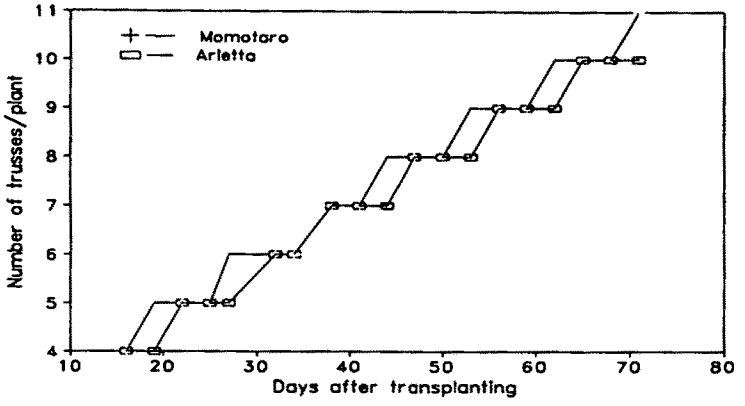


Fig.2.8 Comparison of measured and simulated number of truss per plant in two tomato cultivar, Momotaro and Arletta in spring season experiments, 1995. (line: simulated, symbol: measured)

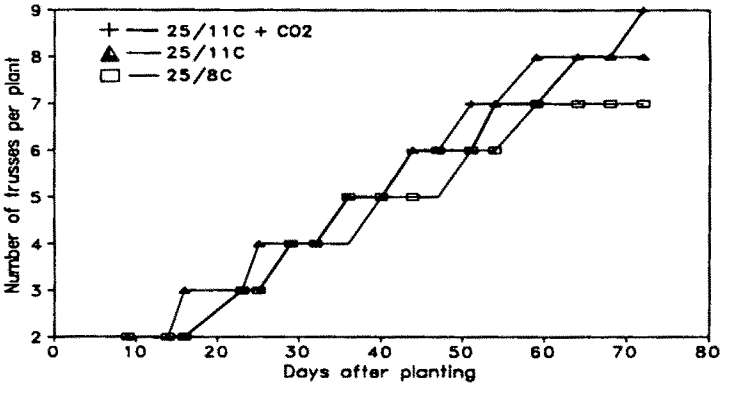


Fig.2.9 Comparison of measured and simulated number of truss per plant in cv. Seokwang at different temperature and CO_2(800ppm) enriched conditions in autumn season experiments, 1995.

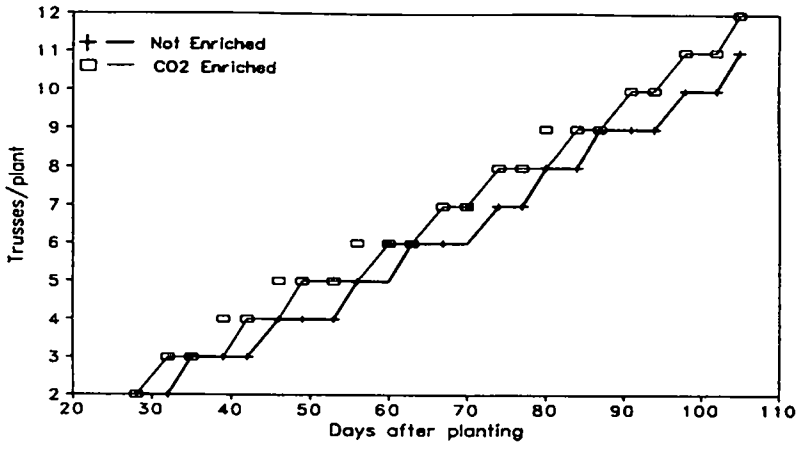


Fig.2.10 Comparison of measured and simulated number of truss per plant in cv. Seokwang at natural and CO_2(1,000ppm) enriched conditions in spring season experiments, 1996.

(4) 착과수

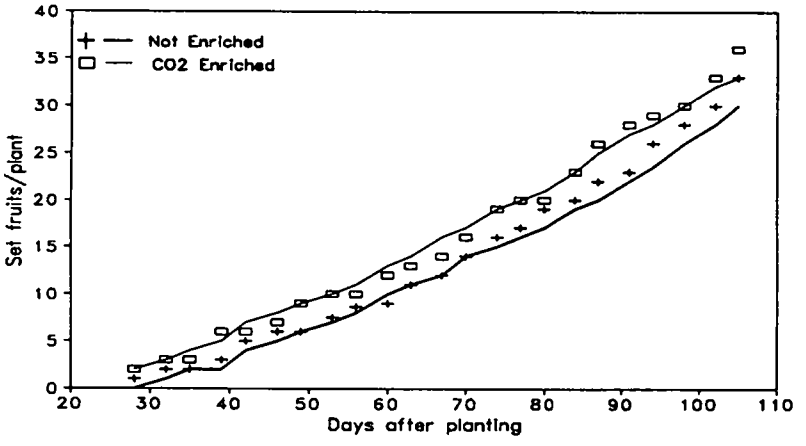


Fig.2.11 Comparison of measured and simulated number of set fruits per plant in cv. Seokwang at natural and CO_2(1,000ppm) enriched conditions in spring season experiments, 1996.

나. 토마토 생장

다음의 그림2.12 - 2. 28에서 보는 바와 같이 토마토 생장모델은 엽면적(그림2.12 - 2.14), 엽중(그림2.15 - 2.16), 경중(그림2.17 - 2.20), 과실중(그림2.21 - 2.23), 성숙과실중(그림2.24 - 2.25) 및 지상부 전체 건물중(그림2.26 - 2.28)을 정확하게 simulation할 수 있는 것으로 나타났다. 다만 생육 후반기에는 엽면적, 엽중 등이 실측치에 비하여 예측치가 다소 컸는데 이는 생육 후반기의 엽노화 및 이병에 따른 하위엽의 고사 탈락이 모델에서 반영되지 않았기 때문인 것으로 생각된다.

(1) 엽면적지수

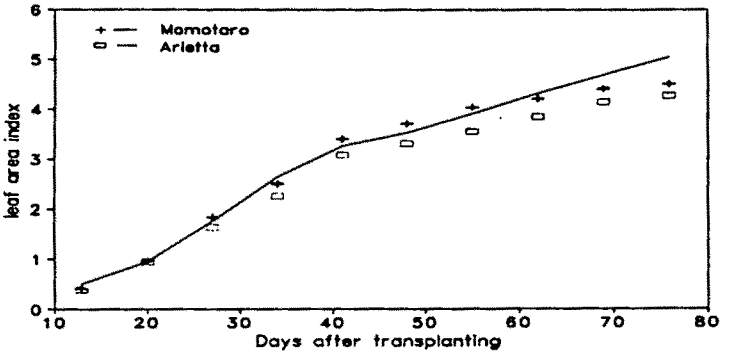


Fig.2.12 Comparison of measured and simulated leaf area index in two tomato cultivars, Momotaro and Arletta.(line: simulated, symbol: measured)

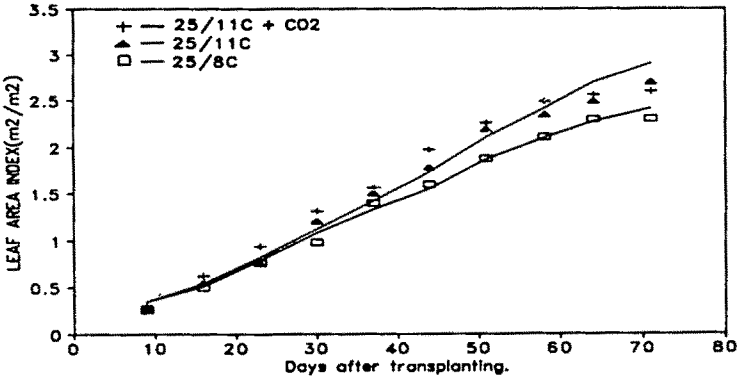


Fig.2.13 Comparison of measured and simulated leaf area index in cv. Seokwang at different temperature and CO_2(800ppm) enriched conditions.

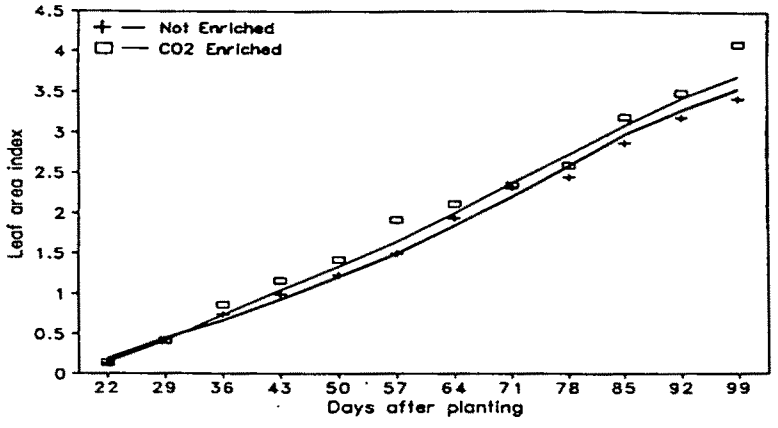


Fig.2.14 Comparison of measured and simulated leaf area index in cv. Seokwang at natural and CO_2(1,000ppm) enriched conditions.

(2) 엽 중

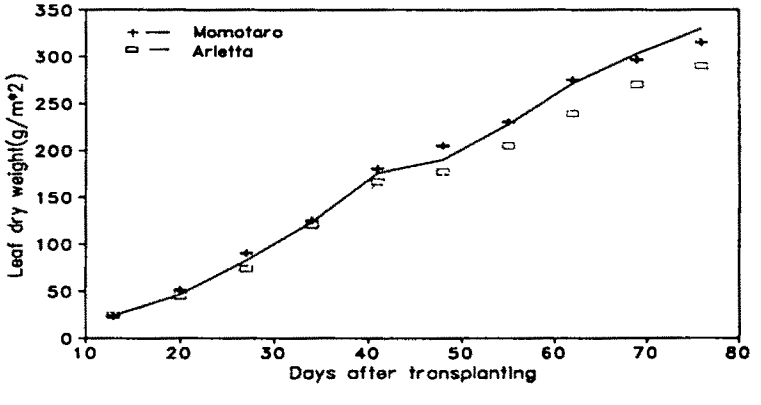


Fig.2.15 Comparison of measured and simulated leaf dry weight in two tomato cultivar, Momotaro and Arletta.(line: simulated, symbol: measured)

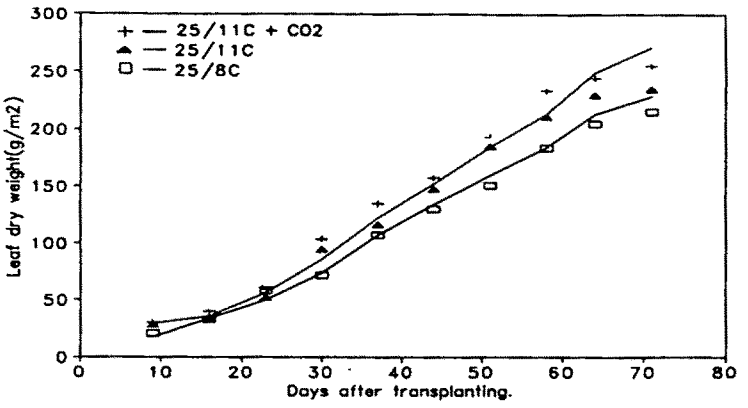


Fig.2.16 Comparison of measured and simulated leaf dry weight in cv. Seokwang at different temperature and CO_2(800ppm) enriched conditions.

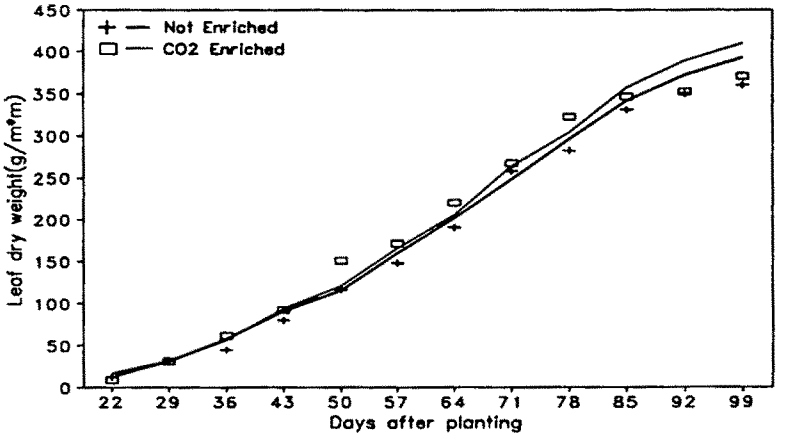


Fig.2.17 Comparison of measured and simulated leaf dry weight in cv. Seokwang at natural and CO_2(1,000ppm) enriched conditions.

(3) 경 중

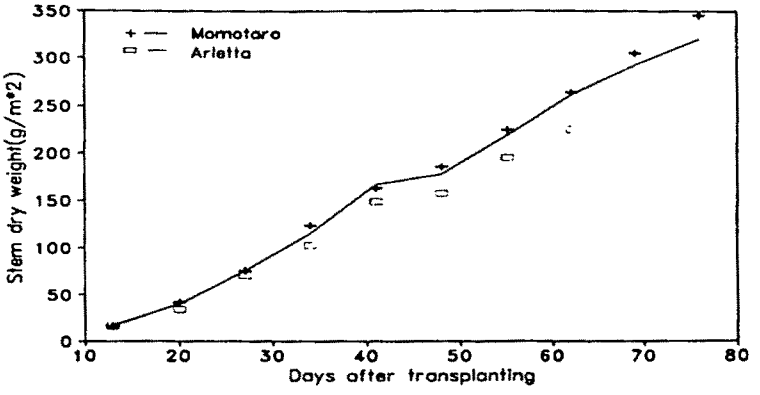


Fig.2.18 Comparison of measured and simulated stem dry weight in two tomato cultivars, Momotaro and Arletta.(line: simulated, symbol: measured)

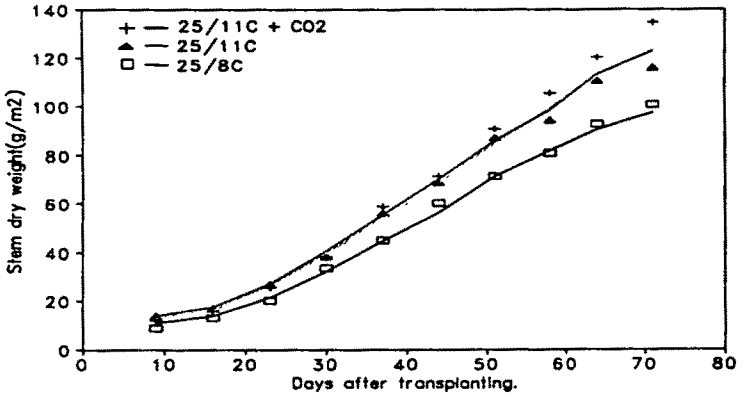


Fig.2.19 Comparison of measured and simulated stem dry weight in cv. Seokwang at different temperature and CO_2(800ppm) enriched conditions.

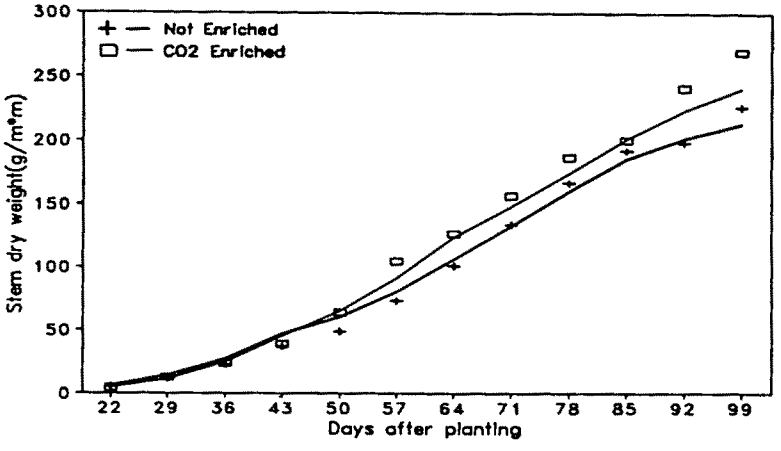


Fig.2.20 Comparison of measured and simulated stem dry weight in cv. Seokwang at natural and CO_2(1,000ppm) enriched conditions.

(4) 과실중

o 전체 과실중

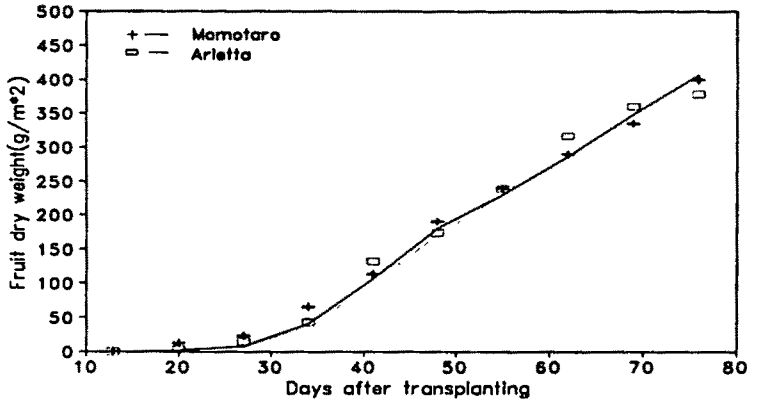


Fig.2.21 Comparison of measured and simulated fruit dry weight in two tomato cultivar, Momotaro and Arletta.(line: simulated, symbol: measured)

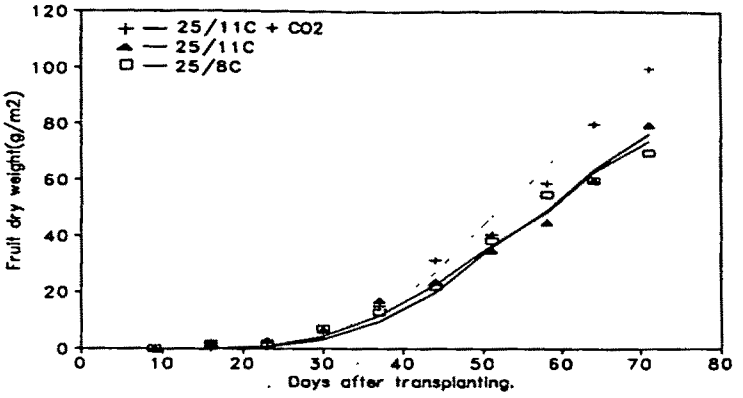


Fig.2.22 Comparison of measured and simulated fruit dry weight in cv Seokwang at different temperature and CO_2(800ppm) enriched conditions.

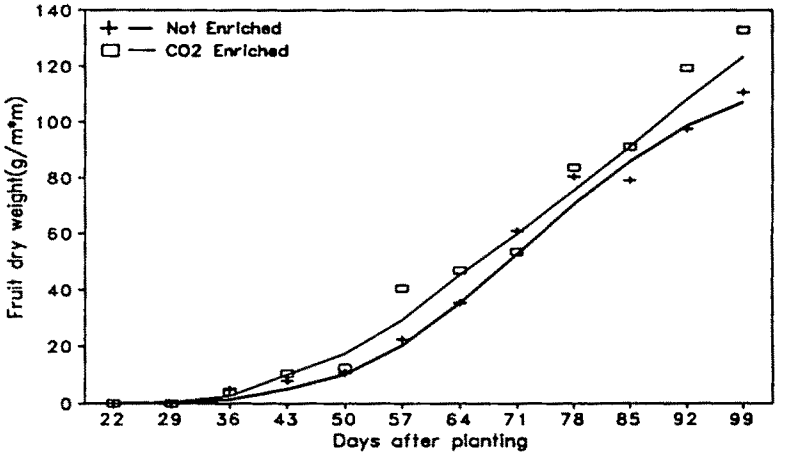


Fig.2.23 Comparison of measured and simulated fruit dry weight in cv. Seokwang at natural and CO_2(1,000ppm) enriched conditions.

o 성숙 과실중

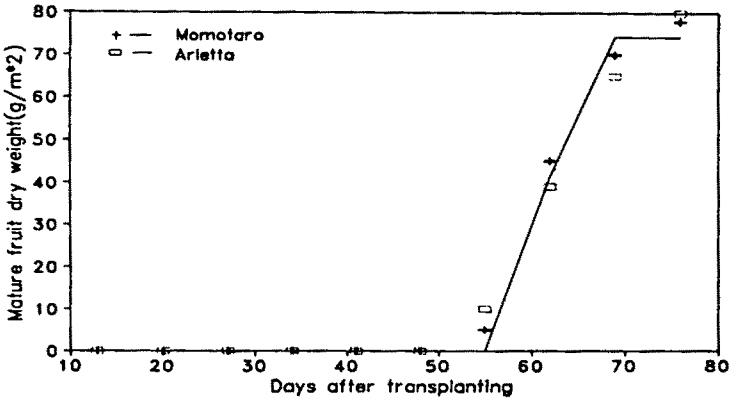


Fig.2.24 Comparison of measured and simulated mature fruit dry weight in two tomato cultivar, Momotaro and Arletta.(line: simulated, symbol: measured)

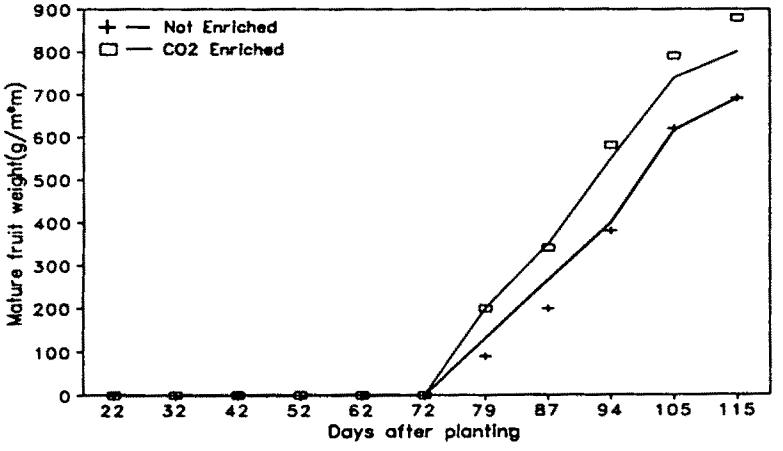


Fig.2.25 Comparison of measured and simulated mature fruit dry weight in cv. Seokwang at natural and CO_2(1,000ppm) enriched conditions.

(4) 지상부 전체 건물중

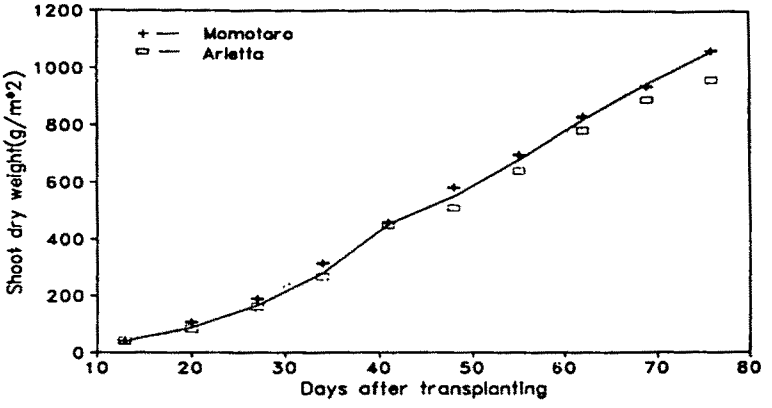


Fig.2.26 Comparison of measured and simulated shoot dry weight in two tomato cultivars, Momotaro and Arletta.(line: simulated, symbol: measured)

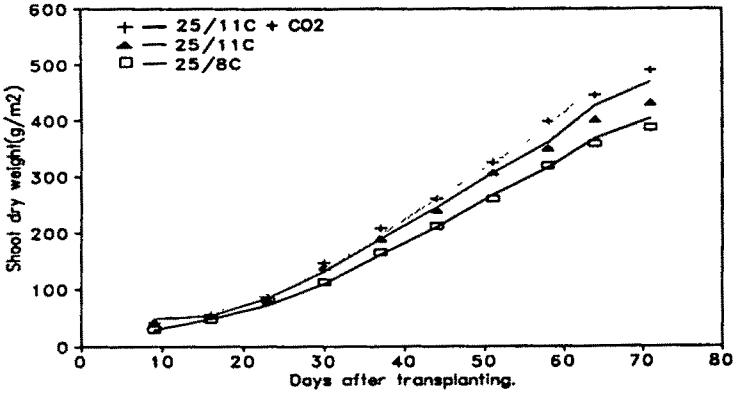


Fig.2.27 Comparison of measured and simulated shoot dry weight in cv. Seokwang at different temperature and CO_2(800ppm) enriched conditions.

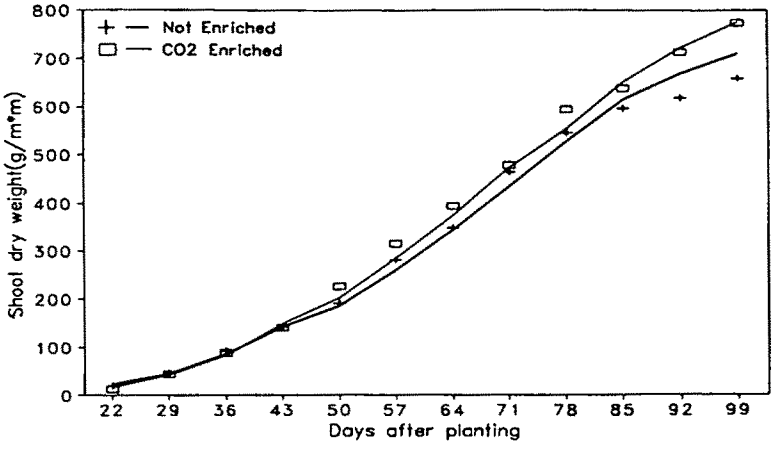


Fig.2.28 Comparison of measured and simulated shoot dry weight in cv. Seokwang at natural and CO_2(1,000ppm) enriched conditions.

제 4절 결 론

토마토 생장 및 발육 모델의 설계 및 program을 완성하여 실험자료를 이용하여 매개변수 추정 및calibration을 하였으며, 또한 다양한 환경조건에서 토마토 재배실험을 하여 얻은 자료를 이용하여 모델의 검증을 수행하였다. 모델의 검증 결과 온실내부의 일사량, 기온, 습도, CO_2 농도 및 토마토 생장 및 발육과 관련된 매개변수 값을 이용하여 온실 재배 토마토의 생장, 발육, 수량을 정확하게 예측할 수 있는 것으로 판단되었다. 따라서 이 모델을 온실 복합환경 제어시스템 및 최적관개 제어시스템의 submodel로 이용할 수 있을 것으로 판단되었다.

제 3 장 온실 미기상 예측모델의 개발

제 1 절 서 언

온실은 피복재에 의하여 외부와 격리되어 외부기상과는 매우 다른 환경이 조성되며, 이로 인하여 외부기상이 부적절한 시기에도 작물재배가 가능하다. 온실기후나 외부기후와 달리 조성되는 것은 주로 다음의 두 가지 원인에 기인된다. 즉 온실은 피복재에 의하여 공기가 갇힘으로서 그렇지 않은 외부공기에 비해 주위 공기와의 교환이 현저하게 감소되며 또한 풍속이 매우 약화된다. 공기교환의 감소는 직접적으로 온실의 에너지와 물질수지에 영향하며 풍속의 감소는 온실공기와 온실에서 자라는 식물, 토양, 피복재 등 온실 구성체와의 에너지, 수증기, 탄산가스 등의 교환에 영향하기 때문이며, 두번째는 온실피복재의 광학적 등성에 기인하는 것으로서 피복재는 온실 안에서의 모든 에너지 수지에 직접적으로 영향하기 때문이다. 온실의 미기상은 직접적으로 작물의 생장에 영향을 하기 때문에 미기상을 정확히 예측하고 작물의 생장에 가장 알맞은 환경이 되도록 조절하는 것은 매우 중요하며 이를 위해서 온실에서 일어나는 물리적 과정을 수식화하여 외부기상 조건의 변화에 따른 온실기상의 변화를 예측하기 위한 모델 개발 연구가 활발히 진행되어 왔으며 (Avissar et al, 1982; Takakura et al, 1971; Kindelan, 1980; Bot, 1989; Kimball, 197) 이 모델들을 온실의 설계와 온실미기상의 최적화를 위한 도구로 이용하기 위한 연구도 이루어지고 있다. (Houter, 1990; Van Henten and Bontsema, 1991; Gitzen et al, 1990; Gitzen and Cate, 1988). 미기상모델을 이와 같은 목적에 이용하기 위해서는 첫째 정확하게 미기상을 예측할 수 있어야 하며 두번째는 다양한 환경조건에서 특정 시스템에도 적용될 수 있도록 가변성 있게 설계가 되어야 한다(Avissar and Mahrer, 1982). 이와 같은 기준에 부합되기 위해서는 쉽게 관측이 되고 온실에 의하여 영향을 받지 않는 환경인자를 1차적인 경계조건으로 이용하여야 하며 또한 온실시스템을 쉽게 측정이 될 수 있는 물리적·생리적 특성으로 명료하게 정의되어야 한다.

본 연구에서는 온실의 각 층위에서의 에너지 및 물질수지 방정식들을 구성하고 기온, 습도, 풍속, 일사량 등 종관기상자료와 온실요소의 열적·광학적 특성을 제 1차 경계조건으로 하여 방정식들을 풀어서 온실내의 미기상 요소들을 예측할 수 있는 모델을 개발하고자 하였으며 궁극적으로는 이 모델을 온실 복합환경최적화 및 최적 관개제어 논리 개발에 이용하고자 하는 것이다.

제 2 절 모델의 설계

1. 모델의 가정

온실은 피복재, 보온 피복재, 작물군락, 멀칭, 각 층 사이의 공기층, 토양층위들로 구성되어 있으며 이들은 수직층위를 이루고 있으며, 토양층을 제외한 모든 층위는 수직·수평적으로 균질한 것으로 가정하였으며 토양 층위는 균질한 20개의 층위로 세분하였다. 시스템의 상부 경계조건은 온실외부의 기온, 습도, 풍속, 단파복사, 대기의 장파복사로 하며 하부 경계조건은 simulation기간 중 변화하지 않는 지하 1m 토양층으로 하였다. 에너지와 물질 flux는 수직방향으로만 일어나고 수평방향 flux는 일어나지 않는 것으로 가정하였으며, 온실 대기중의 수증기뿐만 아니라 식물체, 내부공기, 피복재는 온실에서 일어나는 열류속에 비하면 매우 적기 때문에 이들에 의한 저열은 무시하였다.

2. 온실의 에너지 flux와 수지

그림 3.1은 본 모델에서 고려한 모든 에너지 flux를 나타낸 것이며 각 층(layer)에서 에너지 수지식은 다음과 같다. 그림에서 각 화살표는 개개의 에너지 또는 물질 flux를 나타내며 이들 각각에 대하여 flux식을 유도하였다.

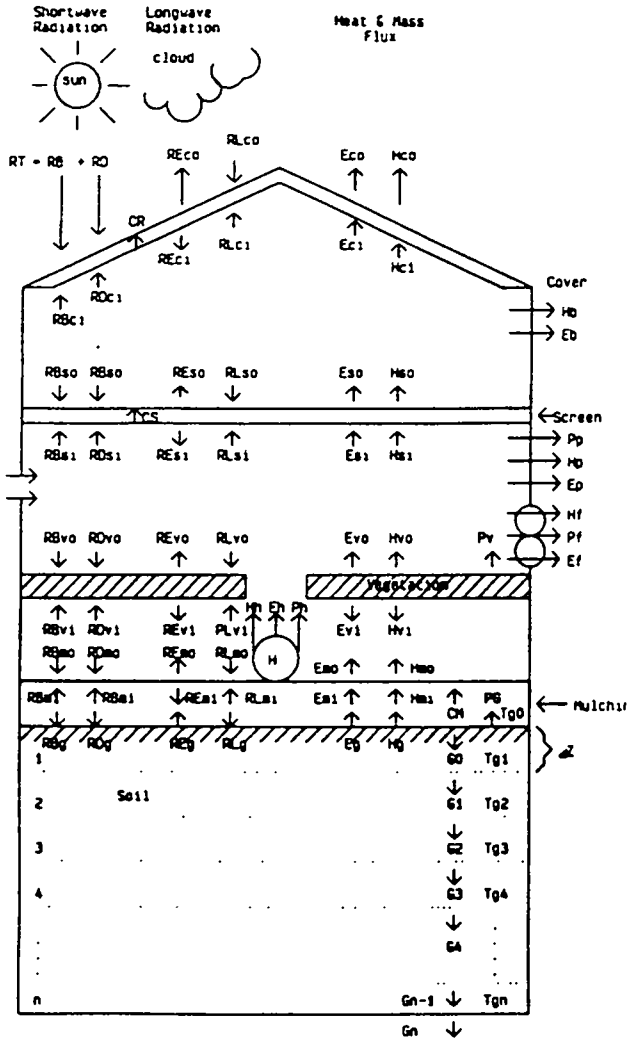


Fig. 3.1 Schematic representation of heat and mass fluxes in the greenhouses

가. 외피복층

온실은 1층의 외부 피복재로 되어 있으며 피복재에는 저열이 되지 않는 것으로 가정하면 피복의 상부와 하부에서의 에너지 수지식은 다음과 같이 쓸 수 있다.

- 상부

$$RBco + RDco + REco - RLco + CR - Eco - Hco = 0 \quad \cdots\cdots 1)$$

- 하부

$$RBci + RDci + REci - RLci - CR + Eci + Hci = 0 \quad \cdots\cdots 2)$$

나. 보온 피복층

외피복내부에 1층의 차광막 또는 보온 피복재가 있는 경우를 가정하여 다음과 같이 피복 상하부에서의 에너지 수지식을 구성하였다.

- 상부

$$RBso + RDso + REso - RLso + CS - Eso - Hso = 0 \quad \cdots\cdots 3)$$

- 하부

$$RBsi + RDsi + REsi - RLsi - CS + Esi + Hsi = 0 \quad \cdots\cdots 4)$$

다. 외피복과 보온 피복재 사이의 공기층

이 층의 공기에 의한 열과 물질의 저장이 일어나지 않는 것으로 가정하였다.

- 현열수지

$$Hso - Hb - Hci = 0 \quad \cdots\cdots 5)$$

- 잠열수지

$$Eso - Eb - Eci = 0 \quad \cdots\cdots 6)$$

라. 작물군락층

작물체온, 기온, 습도, 일사량 이외의 식물생장에 영향하는 요소들은 이상적인 것으로 가정하여 본 모델에서는 고려를 하지 않았으며 식물체에 의한 저열은 없는 것으로 가정하면 작물군락의 에너지 수지식은 다음과 같이 쓸 수 있다.

$$(RBvo + RBvi) + (RDvo + RDvi) + (REvo + REvi) - (RLvo + RLvi) - (Evo + Evi) - (Evo + Evi) - (Hvo + Hvi) = 0 \quad \cdots\cdots 7)$$

마. 멀칭과 보온 피복재 사이의 공기층

공기에 의한 저열이 없는 것으로 가정하였으며 온실의 구성요소들과 열과 물질의 교환을 고려하여 다음과 같이 에너지 수지식을 구성하였다.

- 현열수지

$$(Hvo + Hvi) + Hmo + Hh - Hp - Hf - Hsi = 0 \quad \cdots\cdots 8)$$

- 잠열수지

$$(Evo + Evi) + Emo + Eh - Ep - Ef - Esi = 0 \quad \cdots\cdots 9)$$

바. 멀칭층

토양피복재에 의한 저열을 무시하면 피복의 상하부의 열수지식은 다음과 같이 쓸 수 있다.

- 상부

$$RBmo + RDmo + REmo - RLmo + CM - Emo - Hmo = 0 \quad \cdots\cdots 10)$$

- 하부

$$RBmi + RDmi + REmi - RLmi - CM + Emi + Hmi = 0 \quad \cdots\cdots 11)$$

사. 멀칭과 토양표면사이의 공기층

공기에 의한 열과 수증기의 저장을 무시하면 여기서의 잠열의 수지는 다음과 같이 쓸 수 있다.

- 현열수지

$$Hg - Hmi = 0 \quad \cdots\cdots 12)$$

- 잠열수지

$$Eg - Emi = 0 \quad \cdots\cdots 13)$$

아. 토양층

토양층은 토양표층토와 그 하부의 10개층으로 구성하였으며 표층에서 에너지 수지식은 다음과 같이 쓸 수 있다.

$$(RBg + RDg) + REg - RLg - Eg - Hg - Go = 0 \quad \text{..... 14)}$$

표층과 각 토양 층위에서는 다른 층위와 달리 불균질(깊이에 따라서 토양수분 및 온도가 다르다.)한 것으로 가정하였으며 일차원 열확산 방정식으로부터 각 층위의 에너지 수지식은 다음과 같이 쓸 수 있다.

$$C_{vgi} \Delta Z_i \frac{\partial T_{gi}}{\partial t} = G_{i-1} - G_i (i = 1, \cdots, n) \quad \text{..... 15)}$$

* C_{vgi} : volumetric heat capacity ($J/m^3/K$)

2. 모델 flux 방정식

모델의 각 flux를 계산하기 위한 방정식들은 다음과 같다.

가. 온실외부의 태양복사

태양복사(RT)의 산란성분(RD)과 직달성분(RB)는 Erbs et al(1982)에 따라 다음과 같이 계산하였다.

- 산란복사(RD)

$$RD = RT * mt$$

$$\begin{aligned} mt &= 1 - 0.09A_{tr}, && 0 \le A_{tr} \le 0.22 \\ &= 0.9511 - 0.1604A_{tr} + 4.388(A_{tr})^2 - \\ &\quad 16.638(A_{tr})^3 + 12.336(A_{tr})^4, && 0.22 \le A_{tr} \le 0.80 \\ &= 0.165, && A_{tr} \ge 0.80 \end{aligned}$$

- 직달복사(RB)

$$RB = RT - RD$$

대기권외 태양복사(R_{ex})는 태양상수(S_c)와 태양의 천정각으로부터 다음과 같이 계산하였으며,

$$R_{ex} = S_c \cos\theta_z = S_c \sin\alpha$$

대기의 투과율은 지상에서 측정한 수평면일사량(RT)과 대기권외 태양복사로부터 다음과 같이 계산하였으며 이들의 계산에 필요한 변수들은 다음과 같이 계산하였다.

$$A_{tr} = RT/R_{ex}$$

RT ; global solar radiation ($J/m^2 \cdot sec$)

태양고도(α)와 천정각(θ_z)은 다음과 같이 계산하였다.

$$\cos\theta_z = \sin\alpha = \sin\delta\sin\phi + \cos\delta\cos\phi\cos\omega$$

θ_z ; zenith, α; solar elevation ϕ; lattitude
δ ; solar declination
$\delta = 23.45\sin[360(d_n+284)/365](\pi/180)$
d_n ; julian day

온실 피복재에 대한 직달태양복사 입사각(θ)은 다음과 같이 계산하였다.

$$\cos\theta = \cos\beta\cos\theta_z + \sin\beta\sin\theta_z\cos(\psi - \gamma)$$

ψ; solar azimuth[east(+), west(−)]
γ; azimuth of the outside wall of greenhouse

$$\cos\psi = (\sin\alpha\sin\phi - \sin\delta)/(\sin\alpha\sin\phi)$$
$$0 \le \psi \le \pi/2, \quad \cos\psi \ge 0$$
$$\pi/2 \le \psi \le \pi, \quad \cos\psi \le 0$$

태양상수(S_c)는 다음과 같이 계산하였다.

$$S_c = S_0[1 + 0.033\cos(2\pi d_n/365)]$$

$$S_0 = 1370 J/(m^2 \cdot sec)$$

나. 온실내의 태양복사

온실외부의 태양복사가 피복재를 투과하여 온실내부로 들어오는 태양복사의 각 피복재 표면에서의 북사 수지를 계산하기 위하여 필요한 사항들을 Takakura et al(1975)에 따라 다음과 같이 계산하였다.

온실내부 보온피복재의 그늘진 부분(Pssl)과 햇빛을 직접 받는 부분(Pcsh)의 외피 전체 면적에 대한 비율은 다음과 같이 계산하였다.

$$P_{csl} = \frac{1}{A_c}\sum_{i=1}^{n} A_{ci}, \qquad \theta > 0$$

$$P_{csh} = \frac{1}{A_c}\sum_{i=1}^{n} A_{ci}, \qquad \theta \le 0$$

A_c ; total cover surface area
A_{ci}; cover surface area with tilt angle, β and azimuth, γ

온실내부 보온피복재의 그늘진 부분(Pssl)고 햇빛을 직접 받는 부분(Pssh)의 전체 면적에 대한 비율은 다음과 같이 계산하였다.

$$P_{ssl} = \frac{1}{A_s}\sum_{i=1}^{n} A_{si}, \qquad \theta > 0$$

$$P_{ssh} = \frac{1}{A_s}\sum_{i=1}^{n} A_{si}, \qquad \theta \leq 0$$

A_s ; total screen surface area
A_{si}; screen surface area with tilt angle, β and azimuth, γ

평균 형태 계수는 다음과 같이 계산하였다.

- 외피복의 하늘을 보는 형태계수

$$CF_c = \frac{1}{A_c}\sum_{i=1}^{n} A_{ci}\frac{1 + \cos\beta_i}{2},$$

β ; tilt angle of the surface

- 보온 피복재의 하늘을 향한 형태계수

$$CF_s = \frac{1}{A_s}\sum_{i=1}^{n} A_{si}\frac{1 + \cos\beta_i}{2},$$

β ; tilt angle of the surface

- 온실의 부지면의 외피에 대한 형태계수

$$CF_{gc} = \frac{1}{A_c}\sum_{i=1}^{n} A_{ci}\frac{1 - \cos\beta_i}{2},$$

β ; tilt angle of the surface

직달 태양복사의 평균 입사각은 다음과 같이 계산하였다.

- 외피복에 대한 입사각

$$\theta_{ca} = \frac{1}{A_c}\sum_{i=1}^{n} A_{ci}\theta_i$$

- 보온피복에 대한 입사각

$$\theta_{sa} = \frac{1}{A_s}\sum_{i=1}^{n} A_{si}\theta_i$$

(1) 온실 내외 피복재에 의한 태양복사의 흡수, 반사, 투과

투명매질에 의한 복사의 반사, 흡수, 투과는 매질의 굴절계수에 의하여 결정되며 굴절계수(FI)는 Snell의 법칙에 따라서 다음과 같이 정의된다.

$$FI = \frac{\sin\theta}{\sin\theta_r}$$

θ ; incident angle
θ_r; refraction angle

$$\theta_r = \sin^{-1}(\sin\theta/FI)$$

Lambert-Beer의 법칙에 의하면 복사가 매질의 일정 두께(fl)을 투과하면서 흡수되고 남은 비율 즉 투과율은 다음과 같이 계산된다.

$$aa = \exp(-k_c \cdot fl/\cos\theta_r)$$

k_c ; extinction coefficient(1/mm)
fl ; film thickness(mm)

Fresnel의 법칙으로부터 반사율은 다음과 같이 계산될 수 있다.

$$fr = \begin{cases} \dfrac{(1-FI)^2}{(1+FI)^2}, & \theta = 0° \\ 0.5*\left[\dfrac{\sin^2(\theta-\theta_r)}{\sin^2(\theta+\theta_r)} + \dfrac{\tan^2(\theta-\theta_r)}{\tan^2(\theta+\theta_r)}\right], & \theta > 0° \end{cases}$$

이상의 관계로부터 투명피복재의 태양복사 투과(tr), 반사(re),흡수(ab) 계수는 다음과 같이 계산된다.

$$tr = \frac{(1-fr)^2 \cdot aa}{(1-fr^2 \cdot aa^2)}$$
$$re = fr + \frac{fr \cdot (1-fr)^2 \cdot aa^2}{(1-fr^2 \cdot aa^2)}$$
$$ab = 1 - tr - re$$

산란 복사에 대한 흡수·반사, 투과계수의 계산에는 Kimball(1973)에 따라 태양복사의 유효입사각이 64℃인 것으로 가정하였으며 외피와 내부 피복재에 의한 태양복사의 평균 투과·흡수, 반사계수는 다음과 같이 계산하였다.

- 외피복

o 직달복사

$$tr_{cb} = \frac{1}{A_c}\sum_{i=1}^{n} A_{ci}tr(\theta_i)$$

$$re_{cb} = \frac{1}{A_c}\sum_{i=1}^{n} A_{ci}re(\theta_i)$$

$$ab_{cb} = \frac{1}{A_c}\sum_{i=1}^{n} A_{ci}ab(\theta_i)$$

o 산란복사

$tr_{cd} = tr(64°)$
$re_{cd} = re(64°)$
$ab_{cd} = ab(64°)$

- 온실내부 피복재

o 직달복사

$$tr_{sb} = \frac{1}{A_s}\sum_{i=1}^{n} A_{si}tr(\theta_i)$$

$$re_{sb} = \frac{1}{A_s}\sum_{i=1}^{n} A_{si}re(\theta_i)$$

$$ab_{sb} = \frac{1}{A_s}\sum_{i=1}^{n} A_{si}ab(\theta_i)$$

o 산란복사

$tr_{sd} = tr(64°)$
$re_{sd} = re(64°)$
$ab_{sd} = ab(64°)$

태양복사의 온실 외피복과 내부피복재를 통한 작물군락 표면으로의 평균투과율은 다음과 같이 계산하였다.

- 외피복

o 직달복사

$$tr_{cgb} = \frac{1}{A_g}\sum_{i=1}^{n} A_{csi}tr(\theta_i)$$

Ag ; ground area of the greenhouse
A_{csi} ; shaded area of cover on the ground

- 내부피복

o 직달복사

$$tr_{sgb} = \frac{1}{A_g}\sum_{i=1}^{n} A_{ssi}tr(\theta_i)$$

A_{ssi} ; shaded area of screen on the ground

온실 외피복과 내부피복재의 상·하부에 의한 태양복사의 흡수는 위에서 기술한 관계로부터 다음과 같이 계산할 수 있다.

(가) 외부피복 상부 표면

$$RT_{co} = RB_{co} + RD_{co}$$
$$= ab_{cb}RB(\cos\theta_{ca}/\cos\theta_z)P_{csi} +$$
$$ab_{cd}RD \cdot CF_{cs} + ab_{cd}re_gRT \cdot CF_{gc}$$

re_g ; reflectivity of the outside ground

(나) 외부피복 하부 표면

$$RT_{ci} = RB_{ci} + RD_{ci}$$
$$= ab_{cb}tr_{cb}(RB/\cos\theta_z)[re_{sb}\cos\theta_{sa}P_{ssi}A_s/A_c +$$
$$(tr_{sb})^2\cos\theta_{ca}P_{csh}] + ab_{cd}tr_{cd}re_{sd}RD \cdot CF_sA_s/A_c$$

(다) 내부피복 상부 표면

$$RT_{so} = RB_{so} + RD_{so}$$
$$= [ab_{sb}tr_{cb}RB(\cos\theta_{sa}/\cos\theta_z)P_{ssi} + ab_{sd}tr_{cd}RD \cdot CF_{cs}](1 + RE_{cd})$$

(라) 내부피복 하부 표면

$$RT_{si} = RB_{si} + RD_{si}$$

$$= ab_{sb}tr_{cb}tr_{sb}RB(\cos\theta_{sa}/\cos\theta_z)P_{ssh} +$$

$$ab_{sd}re_v(A_g/A_s)\cdot(tr_{cb}tr_{sgb}RB + tr_{cd}tr_{sd}RD)$$

re_v ; canopy reflectance

식물체 군락의 반사율(albedo, re_v)은 Gourdrian(1975)에 따라서 다음과 같이 계산하였다.

$$re_v = re_c + (re_s - re_c)\exp(-2K\cdot LAI)$$

$$re_c = \frac{2}{1 + K_{bl.dif}/K_{bl}} \cdot \frac{1 - \sqrt{(1-\sigma)}}{1 + \sqrt{(1-\sigma)}}$$

$K_{bl} = 0.5/\sin\alpha$, for sphearical lleaf distribution
$K_{bl.dif} = 0.8$
$K = K_{bl}\sqrt{1-\sigma}$
σ = scattering coefficient of lleaf
re_s = soil reflection coefficient

(2) 온실 내부의 태양복사

온실 내부의 수평면에 들어오는 일사량(GSOL, W/m^2)은 다음과 같이 계산하였다.

$$GSOL = RB*tr_{cgb}*tr_{sgb} + RD*tr_{cd}*tr_{sd}$$

온실내부에 스크린을 치지 않은 경우는 tr_{sgb}=1, tr_{sd}=1이다.

(3) 작물군락에 의한 태양복사 흡수

작물군락내부에서 태양복사는 투과엽층(엽면적지수 ; LAI)의 증가에 따라 지수적으로 감소하는 것을 가정한 Monsi와 Saeki(1953)의 식을 이용하여 다음과 같이 계산하였다.

$$RT_{va} = RT_{vo} + RT_{vi}$$
$$= tr_c \cdot tr_s(1-r_{ev})RT\{1-\exp(-K\cdot LAI)\}$$

K : canopy light extinction coefficient

r_{ev}: canopy reflection coefficient

(4) 멀칭에 의한 태양복사의 흡수

$$\begin{aligned} RT_{ma} &= RT_{mo} + RT_{mi} \\ &= RT \cdot tr_s \cdot tr_c \cdot \exp(-K \cdot LAI) \cdot ab_m \\ &\quad (1 + re_g \cdot tr_m + re_m \cdot re_v) \end{aligned}$$

(5) 토양표면에서의 태양복사 흡수

$$\begin{aligned} RT_{ga} &= RT \cdot tr_s \cdot tr_c \cdot tr_m \cdot \exp(-K \cdot LAI) ab_g \\ &\quad (1 + re_g \cdot re_m + tr_m^2 \cdot re_v \cdot re_g) \end{aligned}$$

다. 장파복사

온실의 각 층위에서의 장파복사의 흡수는 각 구성요소에 의한 1차 반사만을 고려하여 다음과 같이 계산하였다.

(1) 외피복의 장파복사 흡수

$$\begin{aligned} RL_c &= RL_{co} + RL_{ci} - (RE_{co} + RE_{ci}) \\ &= \varepsilon_c(1 + re_{ls} \cdot tr_{lc} + re_{lv} \cdot tr_{lc} \cdot tr_{ls}^2 \\ &\quad + re_{em} \cdot tr_{lc} \cdot tr_{ls}^2 \cdot tr_{lv}^2 \\ &\quad + re_{lg}^2 \cdot tr_{lc} \cdot tr_{ls}^2 \cdot tr_{lv}^2 \cdot tr_{lm}^2) R_L \\ &\quad + \varepsilon_c(\varepsilon_s \sigma T_s^4 + tr_{ls} \cdot \varepsilon_v \sigma T_v^4 \\ &\quad + tr_{ls} \cdot tr_{lv} \cdot \varepsilon_m \sigma T_m^4 \\ &\quad + tr_{ls} \cdot tr_{lv} \cdot tr_{lm} \varepsilon_g T_{go}^4) \\ &\quad - \varepsilon_c \sigma T_c^4(2 - \varepsilon_c re_{ls} - \varepsilon_c tr_{ls}^2 \cdot tr_{lv}^2 \cdot re_{lm} - \varepsilon_c tr_{ls}^2 \cdot re_{lv} \\ &\quad - \varepsilon_c tr_{ls}^2 \cdot tr_{lv}^2 \cdot tr_{lm}^2 \cdot re_{lg}) \end{aligned}$$

(2) 내부 피복의 장파복사 흡수

$$\begin{aligned} RL_s &= RL_{so} + RL_{si} - (RE_{so} + RE_{si}) \\ &= \varepsilon_s(1 + tr_{ls} \cdot re_{lv} + tr_{ls} \cdot tr_{lv}^2 \cdot re_{lm} \\ &\quad + tr_{ls} \cdot tr_{lv}^2 \cdot tr_{lm}^2 \cdot re_{lg}) tr_{lc} \cdot R_L + \varepsilon_s(\varepsilon_c \sigma T_c^4 \\ &\quad + tr_{ls} \cdot re_{lv} \varepsilon_c \sigma T_c^4 + tr_{ls} \cdot tr_{lv} \varepsilon_m \sigma T_m^4 \\ &\quad + \varepsilon_v \sigma T_v^4 + tr_{ls} \cdot re_{lc} \varepsilon_v \sigma T_v^4 \\ &\quad + tr_{lv} \cdot tr_{lm} \varepsilon_g \sigma T_{go}^4) \\ &\quad - \varepsilon_s \sigma T_s^4(2 - \varepsilon_s re_{lc} - \varepsilon_s re_{lv} \\ &\quad - \varepsilon_s re_{lm} \cdot tr_{lv}^2 - \varepsilon_s re_{lg} \cdot tr_{lv}^2 \cdot tr_{lm}^2) \end{aligned}$$

(3) 작물군락에 의한 장파복사 흡수

$$
\begin{aligned}
RL_v &= RL_{vo} + RL_{vi} - (RE_{vo} + RE_{vi}) \\
&= \varepsilon_v (1 + tr_{lv} \cdot re_{lm} + tr_{lv} \cdot tr_{lm}^2 \cdot re_{lg}) \\
&\quad tr_{lc} \cdot tr_{ls} \cdot R_L + \varepsilon_v(tr_{ls} \cdot \varepsilon_c \sigma T_c^4 + \\
&\quad \varepsilon_s \sigma T_s^4 + \varepsilon_m \sigma T_m^4 + tr_{lm} \varepsilon_g \sigma T_{go}^4) \\
&\quad - \varepsilon_v \sigma T_v^4(2 - \varepsilon_v re_{lc} tr_{ls}^2 - \varepsilon_v re_{ls} \\
&\quad - \varepsilon_v re_{lm} - \varepsilon_v re_{lg} \cdot tr_{lm}^2)
\end{aligned}
$$

(4) 멀칭에 의한 장파복사 흡수

$$
\begin{aligned}
RL_m &= RL_{mo} + RL_{mi} - (RE_{mo} + RE_{mi}) \\
&= \varepsilon_m(1 + re_{lg})\, tr_c \cdot tr_s \cdot tr_v \cdot R_L + \\
&\quad \varepsilon_m(tr_{ls} \cdot tr_{lv} \cdot \varepsilon_c \sigma T_c^4 + tr_{lv} \varepsilon_s \sigma T_s^4 + \\
&\quad \varepsilon_v \sigma T_v^4 + \varepsilon_g \sigma T_{go}^4) - \varepsilon_m \sigma T_m^4(2 - \\
&\quad \varepsilon_m re_{lc} \cdot tr_{ls}^2 \cdot tr_{lv}^2 - \varepsilon_m re_{ls} \cdot tr_{lv}^2 - \\
&\quad \varepsilon_m re_{lv} - \varepsilon_m re_{lg})
\end{aligned}
$$

(5) 토양표면에 의한 장파복사 흡수

$$
\begin{aligned}
RL_g &= RL_{go} - RE_{go} = \varepsilon_g\, tr_c \cdot tr_s \cdot tr_v \cdot tr_m R_L + \\
&\quad \varepsilon_g(tr_{ls} \cdot tr_{lv} \cdot tr_{lm} \varepsilon_c \sigma T_c^4 + tr_{lv} \cdot tr_{lm} \cdot \varepsilon_s \sigma T_s^4 \\
&\quad + tr_{lm} \cdot \varepsilon_v \sigma T_v^4 + \varepsilon_m \sigma T_m^4) - \varepsilon_g \sigma T_g^4(1 - \\
&\quad \varepsilon_g re_{lc} \cdot tr_{ls}^2 \cdot tr_{lv}^2 \cdot tr_{lm}^2 - \varepsilon_g re_{ls} \cdot tr_{lv}^2 \cdot tr_{lm}^2 \\
&\quad - \varepsilon_g re_{lv} \cdot tr_{lm}^2 - \varepsilon_g re_{lm})
\end{aligned}
$$

* $(\varepsilon_c, \varepsilon_s, \varepsilon_v, \varepsilon_m, \varepsilon_g)$, $(re_{lc}, re_{ls}, re_{lv}, re_{lm}, re_{lg})$, $(tr_{lc}, tr_{ls}, tr_{lv}, tr_{lm}, tr_{lg})$, $(T_c, T_s, T_v, T_m, T_{go})$는 각각 외피복, 내부피복, 작물군락, 멀칭, 토양표면의 복사능, 반사율, 투과율, 온도이다.

천공으로부터의 장파복사는 다음과 같은 식을 이용하여 계산하였다.

$$R_L = \varepsilon_a \sigma T_{out}^4$$

여기서 T_{out}는 외기온이며 e는 외기의 수증기압 σ는 Stefan-Boltzman상수이며 맑은 날의 대기투과율은 Brutsaert(1975)에 의하여 대기의 절대습도(AH, g/m^3)로부터 다음과 같이 계산하였다.

$$\varepsilon_a = 0.58\,AH^{\frac{1}{7}}$$

한편 흐린 날의 대기투과율은 운량(c; 0 - 1)과 대기의 투과율(ATR)에 따라서 다음과 같이 계산하였다(Unsworth & Monteith,1975).

$$\varepsilon_{ac} = (1-0.84c)\varepsilon_a + 0.84c$$

$$c = 2.33 - 3.33ATR$$

라. 전도, 대류, 환기

(1). 토양의 열전도

지표층을 제외한 토양 각 층위에서 열전도율은 다음과 같이 계산 하였으며 토양의 열전도도는 토성, 토양수분, 토양 가비중으로부터 McInnes(1981)의 식을 적용하여 계산하였다.

$$G_i = -K_{si}\frac{\delta T_{gi}}{\delta Z_i}$$

K_{si} : thermal conductivity of soil

지표층의 열전도는 다음과 같은 근사식(Chung & Horton, 1987)을 이용하여 계산하였다.

$$G_0 = -K_1[\frac{Tg_0 - Tgo_1}{\Delta Z_1}] - C_{vg1}(Tg_0 - Tgo_0)\Delta Z_1/2\Delta t$$

Tg_0 : surface temperature of present time step
Tgo_0: surface temperature of previous time step
Tgo_1: soil temperature of present time step for the node at vertical position 2

(2) 현열과 잠열 flux

온실 피복재 및 상면과 내외 공기 사이의 현열과(H)과 잠열(E) flux는 다음과 같이 계산 하였

다

$$H = \rho c_p h \Delta T$$

$$E = \rho \lambda h \Delta q$$

ρ : density of air(kg/m^3)
c_p : specific heat of air(J/kg/C)
ΔT : temperature difference(C)
λ : latent heat of vaporization(J/kg)
Δq : specific humidity difference(kg/kg)
h : heat transfer coefficient(m/s)

온실 피복재와 내외 공기와의 현열 및 잠열 전달 계수는 다음과 같이 계산하였다.(Bussinger, 1963; .Kimbal, 1973; Avissar & Mahrer, 1982; Monteith & Unsworth, 1990)

-. 온실 피복재와 외부공기

$h = 0.0029Ws$
Ws: wind speed at 2m(m/s)

-. 온실 내부 공기와 피복재(

$h = 0.001525\Delta T^{1/3}$

-. 온실 내부 공기와 상면

$h = 0.004\Delta T^{1/3}$ for $\Delta T > 0$
$h = 0.004\Delta T^{1/3}$ for $\Delta T < 0$

한편 토마토 군락의 증산잠열과 현열flux는 본 보고서의 제5장의 증산모델에서와 같이 계산하였다.

(3) 환기에 의한 현열과 잠열 교환

환기에 의한 현열(H_f)과 잠열(E_f)의 교환은 다음과 같이 계산하였다.

$$H_f = \rho c_p h_{ven}(T_{in} - T_{out})$$

$$E_f = \rho \lambda h_{ven}(q_{in} - q_{out})$$

h_{ven} : ventilation transfer coefficient
T_{in} : air temperature inside the greenhouse
q_{in} : specific humidity inside the greenhouse
T_{out} : air temperature outside the greenhouse
q_{out} : specific humidity outside the greenhouse

한편 환기교환계수는 측창 환기창의 개방면적(Aw, m^2)과 외부 풍속(Ws, m/s)로부터 다음과 같이 계산하였다(Papapadakis et al, 1996).

$$h_{ven} = (\alpha + \beta Aw \cdot Ws)/A_g$$

여기서 α와 β는 상수로서 Papadakis et al(1996)이 제시한 값을 이용하지 않고 본 연구에서 사용한 온실에 맞게 calibration하여 이용하였다.

3. 모델의 수치 해법

그림 2.2는 모델 방정식들의 수치해법을 나타내는 흐름도이다. 온실의 형태, 크기등 일반적인 특성치, 토양, 식물 및 피복재의 열적 및 광학적 특성, 엽면적 지수, 토양온도 등이 내부 입력상수 또는 초기치 자료로서 필요하며 외부 입력 자료로서는 기온, 습도, 일사량, 풍속 등의 외부기상조건이 필요하다. 수치해법의 순서는 다음과 같다.

① 내.외부 피복, 군락, 멀칭면, 토양표면 에서의 순복사량과 온실 내외부 공기와 온실 구성요소들 간의 현열 및 잠열 전달계수를 계산한다.

② 에너지 및 물질수지 연립 방정식을 Newton-Raphson방법으로 수치 해석하여 토양표면 온도, 멀칭 온도, 작물군락 온도, 피복재 온도, 온실내의 습도 및 CO_2 농도 를 계산한다. 각 방정식의 1차 편도함수는 수치 미분(Rechardson's Improvement Formula)에 의하여 계산하므로서 모델 층위의 추가 삭제를 용이하게 하였다.

③ 적당한 수치해가 얻어질 때까지 ① ~ ②를 반복 계산한다.

④ 토양 층위별 온도는 ① ~ ③에서 구한 토양표면 온도와 1m 깊이의 토양온도를 경계조건으로 하여 implicit finite difference method에 의하여 각 층(5cm 간격)의 토양온도를 계산한다.

⑤ 다음의 time step에 대하여 ① ~ ④를 반복한다.

미기상모델은 Quick Basic으로 program되었으며 source list를 부록에 수록하였다.

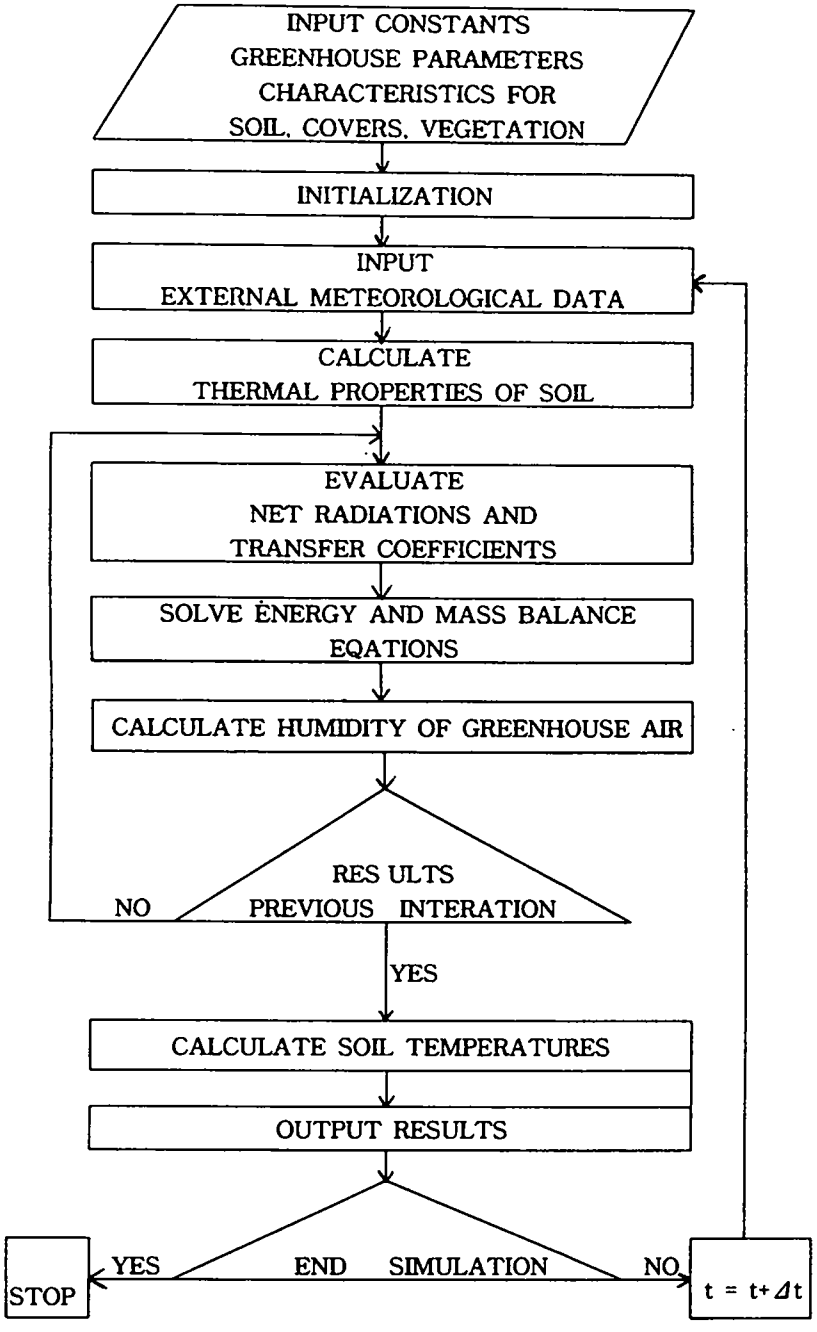


Fig. 3.2 Flow chart for the computer program of the model

제 3 절. 모델 검증

1. 미기상관측

미기상 모델의 파라미터추정과 검증을 위하여 온실 내외부의 풍속, 온도, 습도, 일사량, 순복사량, 지중열류량, 토양온도, 토양수분, 증산, 엽온 등의 관측 실험을 실시하였다. 온실 내외부 기상의 관측은 서울대학교 농업생명과학대학 부속농장에 본 연구를 위하여 설치된 4동의 프라스틱 온실에서(70㎡/동)에서 1995년, 1996년,1997년 봄작기 및 가을 작기에 토마토를 재배하면서 관측하였다. 온실 내외부의 기상관측 요소의 측정은 기상 측정 센서를 data logger(Campbell Scientific Inc.)에 연결하여 10초마다 관측하여 10분간의 평균치를 집록하였다. 한편 토마토의 엽면적, 초장, 건물중 등 생장과 발육도 1주일 간견으로 측정하였다. 관측된 자료를 이용하여 모델의 parameter추정 및 검증을 하였으며 그 일부를 다음에 제시하였다.

2. 모델의 검증

측정한 온실 내외의 미기상 자료와 토마토 생육자료를 이용하여 모델의 parameter 추정 및 검증을 실시하였다. 모델의 simulation에 이용한 특성치들은 표3.1 및 3.2와 같다.

3.1. Photometric properties of cladding film, mulch film and soil

Parameters	Cladding (polyethylene)	Mulch (polyethylene)
Short-wave radiation		
Reflectance	0.120	0.084
Absorptance	0.050	0.036
Transmittance	0.830	0.880
Long-wave radiarion		
Emissivity	0.067	0.030
Reflectance	0.146	0.120
Transmittance	0.791	0.850
Refraction index	1.50	
Ligh extinction coefficient	0.165(1/mm)	
Film thickness	0.05mm	0.01mm

Table 3.2 Values of the parameters used for the simulation

Parameters	Values	Unit
Soil		
Bulk density(Db)	1.10	g/cm^3
Particle density(Dp)	2.65	g/cm^3
Clay fraction	0.20	
Water content(WAT)	0.20	g/cm^3
Bottom(1m) boundary temperature	13.50	℃
Vegetation		
Light extinction coefficient	0.58	
Scattering coefficient	0.20	
Emmissivity	0.95	
Charcteristic length of leaves	0.06	m
Greenhouse		
Length	14.0	m
Width	5.0	m
Wall height	2.3	m
Gabble height	1.1	m
Surface area	169.4	m^2
Greenhouse volume	199.5	m^3
Orientation	north - south	
Ventilation		
Fan ventilation rate	0.36	m^3/s
Side window ventilation rate	0.17 + 0.045Aw*Ws	m^3/s
	Aw: window area	m^2
	Ws: wind speed	m/s

가. 환기율의 추정

1997년 5월18일부터 22일까지 환기방법을 달리하면서 측정한 미기상 자료를 이용하여 model simulation을 통하여 온실의 간극, 측창 및 환기선에 의한 환기효율을 결정하였다. 5월18일은 환기창을 폐쇄하였으며, 19일은 환기창을 폐쇄한 상태에서 환기선만을 가동하여 환기하였으며 20일은 환기창을 14m^2, 21일과 22일은 7m^2 개방한 상태에 온실 내외의 미기상을 측정하였다. 환기율의 추정은 환기와 관련된 parameter를 제외한 parameter는 표3.1과 3.2에 나타난 값으로 고정하고 환기와 관련된 parameter만을 변경시켜 가면서 반복 simulation을 하여 온실 온도와 습도의 추정치와 실측치 사이의 평균평방오차(root mean square error)가 가장 적어지도록 환기관련 parameter를 결정하였다. 표3.2에서 보는 바와 같이 본 연구에 사용한 온실에 설치된 환기선의 환기효율은 0.36m^3/s였으며 측창에 의한 환기효율(Ven, m^3)은 외부 풍속(Ws, m/s)과 측창 개방면적(Aw, m^2)으로부터 다음과 같이 추정되었다.

$$Ven = 0.170 + 0.045Ws*Aw$$

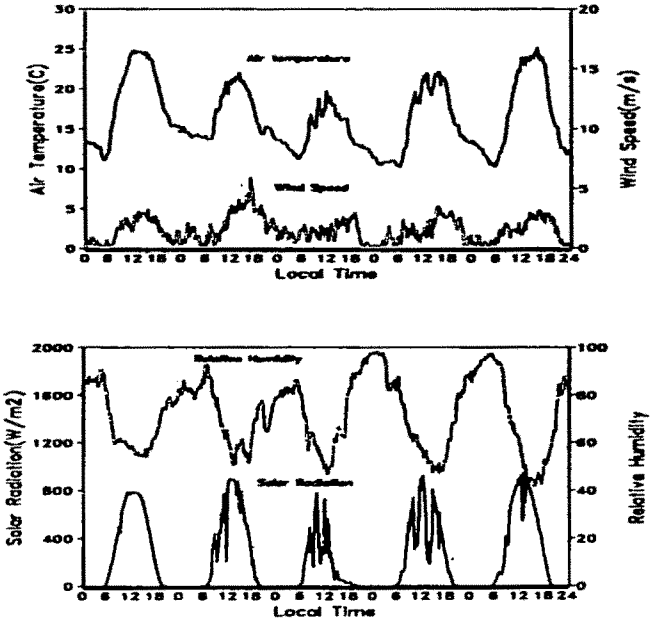


Fig. 3.3 Diurnal marches of air temperature(1.5m), relative humidity(1.5m), solar radiation and wind speed(3m) used as input variables for simulation of greenhouse microclimate from May 18 to May 22, 1997.

이와 같이하여 측정한 환기 parameter값, 표3.1 - 2에 표시한 parameter값과 외부 기상 조건(그림3.3)을 이용하여 1997년 5월 12일부터 24일까지 환기 조건을 달리하여 가며 온실내부 기상을 simulation한 결과는 다음과 같다. 이때 토마토 엽면적지수는 0.3으로 생육 초기단계였다.

온실의 측창과 환기선 등에 의하여 환기 정도를 달리하면서 실측한 온실 내부의 기온, 습도 및 수증기압과 모델 예측치를 비교한 것이 그림 3.4이다. 그림에서 보는 바와 같이 본 모델은 기온, 상대습도 및 수증기압을 매우 정확하게 예측하는 것으로 나타났다.

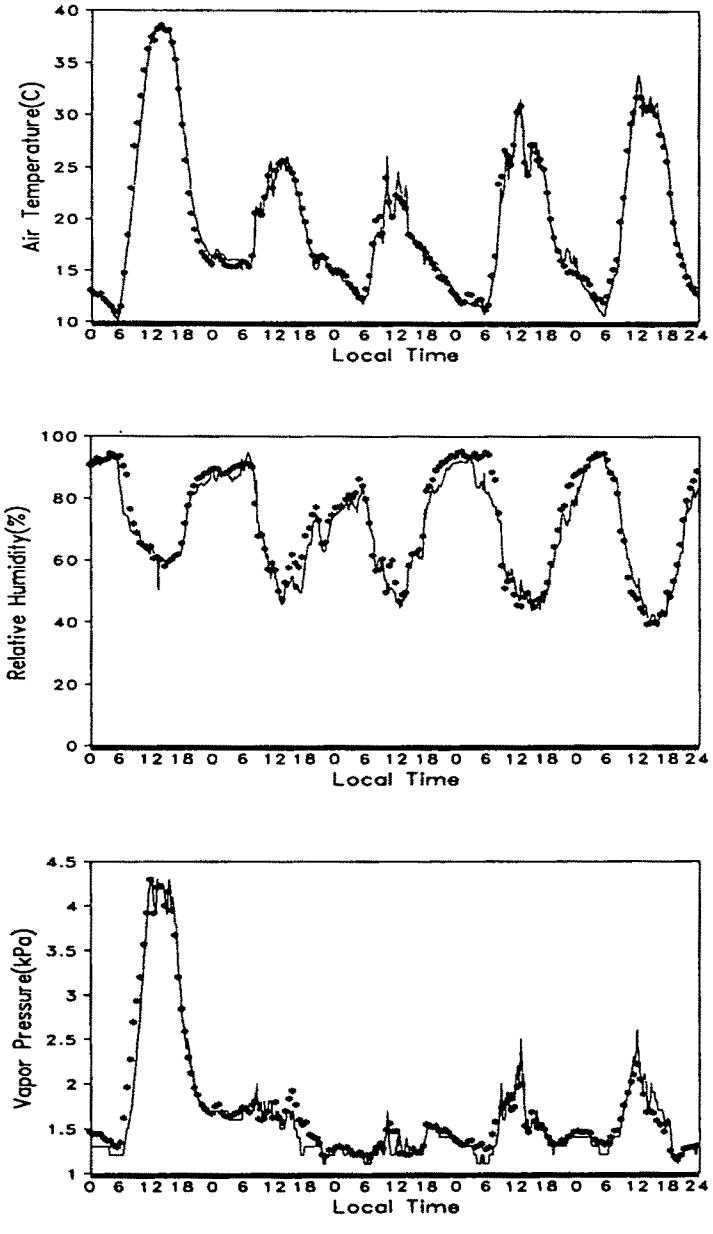


Fig. 3.4 Comparisons between predicted(--) and measured(*) diurnal courses of air temperature, relative humidity and vapor pressure in greenhouse on 18 to 22 May,1997. May 18: window closed, May 19: fan ventilation, May 20: side window opened(14m^2), May 21 and 22: side window opened(7m^2)

한편 미기상 모델로 예측한 온실 내부 토마토 군락의 증산량과 토양 증발량은 그림 3.5와 같다. 그림에서 보는 바와 같이 모델은 증산과 증발의 일변화를 현실적으로 예측하는 것으로 생각되었다. 또한 이 경우 토마토 생육 초기단계(엽면적지수 0.3)이기 때문에 증산량에 비하여 토양 증발량이 많았다.

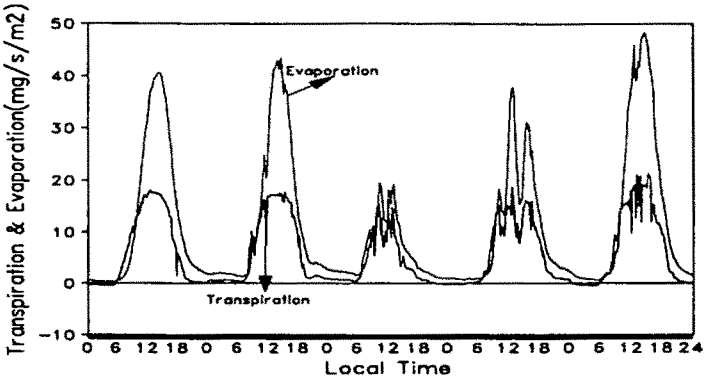


Fig. 3.5 Diurnal courses of simulated transpiration and soil evaporation in greenhouse on 18 to 22 May,1997.

미기상 모델로 예측한 온실내부의 일사량(SolR), 토마토 군락의 순복사량(CNR), 지면의 순복사량(SNR), 지표면에서 지중 열류량(SHF)은 그림 3.8과 같다.

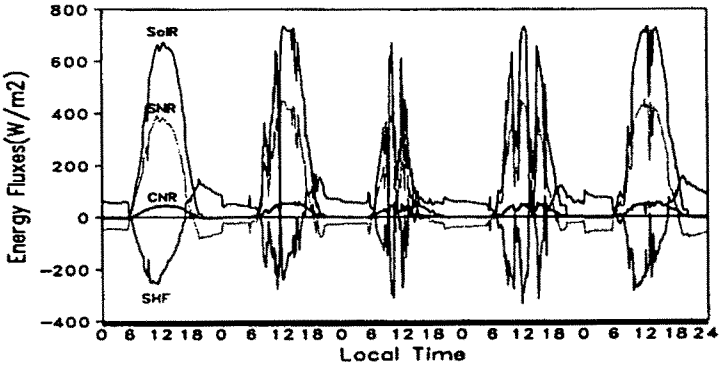


Fig. 3.6 Diurnal courses of simulated solar radiation(SolR), canopy net radiation(CNR) of tomato, soil surface net radiation(SNR) and soil surface heat flux(SHF) on 18 to 22 May,1997.

나. 모델의 검증

모델을 실용적으로 이용할 수 있기 위하여는 모델의 parameter추정 및 calibration에 사용한 자료이외의 독립자료를 이용하여 모델의 적합도가 검증되어야한다. 다양한 외부 환경조건에서 모델의 적합도를 검증한 결과 모델온실의 미기상을 매우 정확하게 예측하였으며 이중 1996년 11월 17일 온실 미기상예측 결과만을 다음에 제시하였다. 이 날의 외부 기상조건은 그림 3.7와 같으며 당시 토마토는 엽면적지수가 2.3이었으며 모델의 parameter값은 표 3.1과 3.2을 이용하였다.

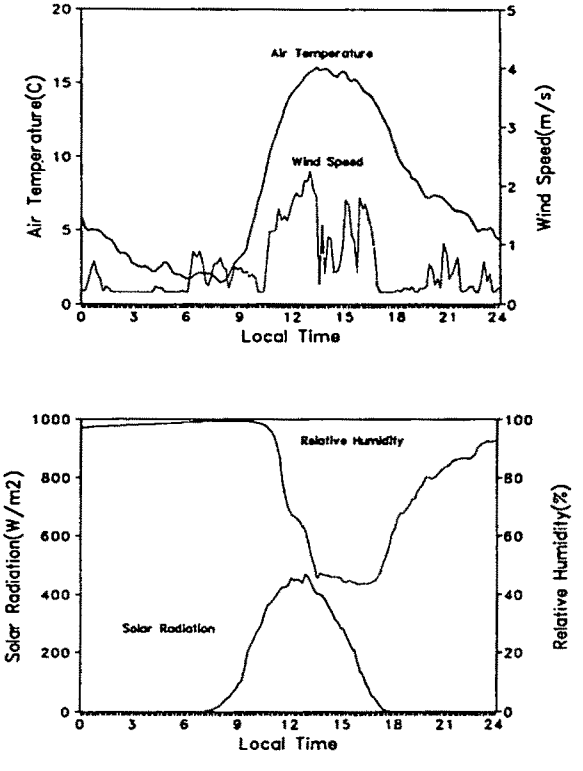


Fig. 3.7 Diurnal marches of air temperature(1.5m), relative humidity(1.5m), solar radiation and wind speed(3m) used as input variables for simulation of greenhouse microclimate on 17 November, 1996.

온실 내부의 일사량과 토마토 군락의 흡수 일사량 및 순복사량의 예측치와 실측치를 비교한 것이 그림 3.8이다. 그림에서 보는 바와 같이 모델은 온실 내부의 일사량, 토마토 군락의 흡수일사량 및 순복사량을 정확하게 simulation하는 것으로 나타났다.

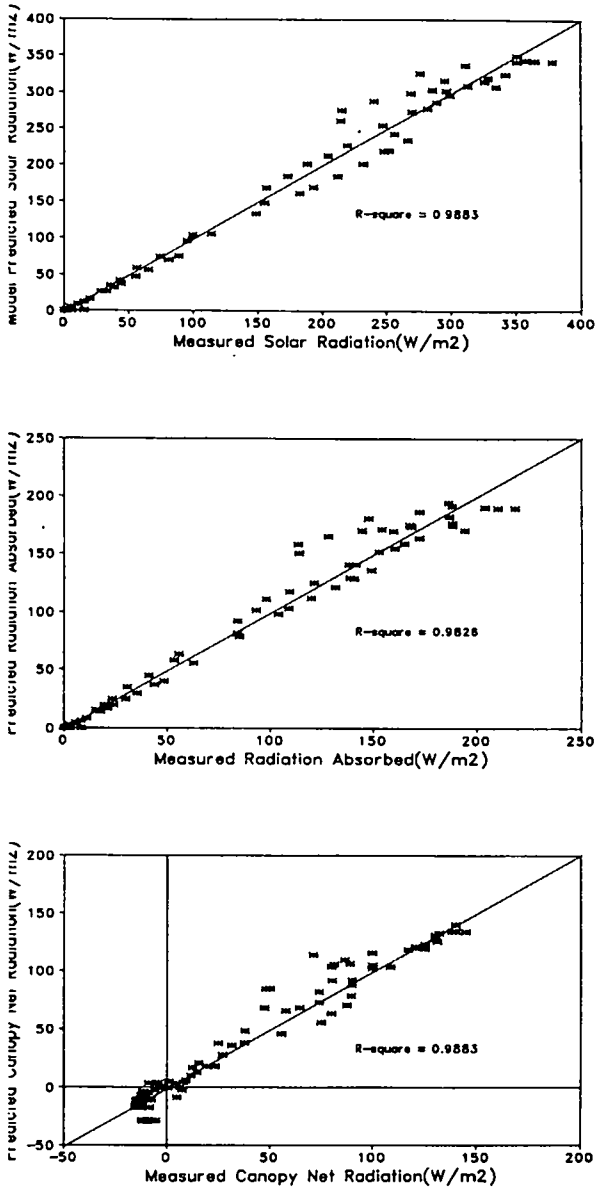


Fig. 3.8 Plots of predictions versus measurements for solar radiation in greenhouse, solar radiation absorbed by and net radiation of tomato canopy on 17 November, 1996.

온실의 기온, 상대습도 및 수증기압의 예측치는 그림 3.9에서 보는 바와 같이 실측치와 잘 일치하였다.

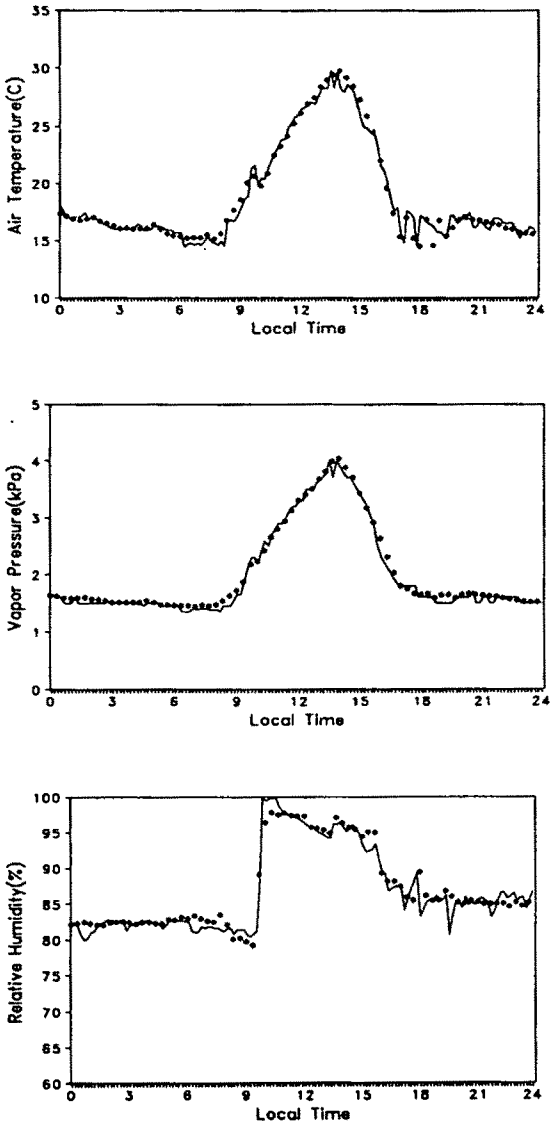


Fig.3. 9 Comparison between predicted(--) and measured(*) diurnal course of air temperature, relative humidity and vapor pressure in greenhouse on 17 November, 1996.

온실 토마토 군락의 온도와 증산 속도의 모델에 의한 예측치는 그림 3.10에서 보는 바와 같이 실측치와 잘 일치하였다.

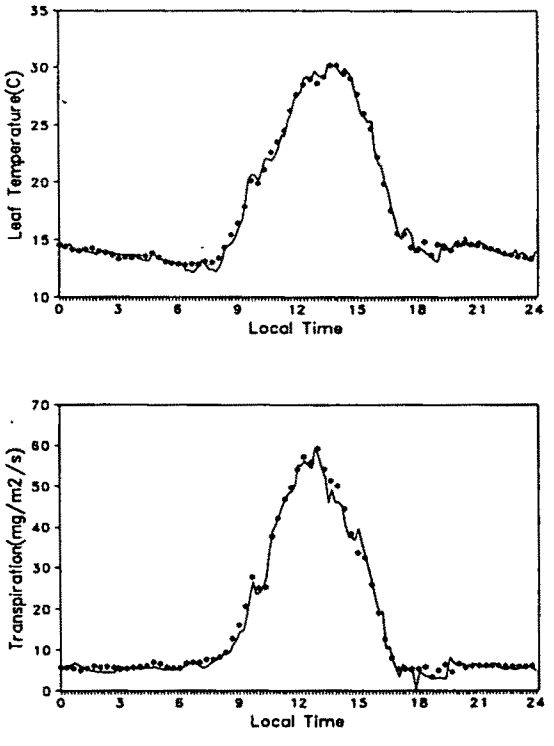


Fig.3.10 Comparison between predicted(--) and measured(*) diurnal course of leaf temperature and transpiration in greenhouse on 17 November, 1996.

제 4절 결 론

온실 미기상 모델의 설계 및 program을 완성하여 calibration 및 다양한 외부 환경조건에서 검증을 수행하였다. 모델의 검증 결과 온실 외부의 일사, 기온, 습도 및 풍속과 피복재, 토마토, 토양 등의 온실 구성 요소들의 특성치를 이용하여 온실 내부의 일사, 기온, 습도, 엽온, 증산량 등을 정확하게 예측할 수 있는 것으로 판단되어 이 모델을 온실복합환경 제어시스템 및 최적관개 제어시스템의 submodel로 구성하였다. 본 연구에서 개발한 미기상 모델은 기존의 모델들과

는 달리 에너지 및 물질수지 방정식들을 수치미분하는 방법을 적용함으로서 모델의 구성방정식을 일일이 편미분하여 모델 program에 삽입하는 복잡성을 탈피하였다. 따라서 모델에 새로운 층위를 추가하거나 삭제하는 것이 용이하도록 설계되었다.

제4장 시설내 복합환경 최적제어모델 개발

제1절 서언

시설내 작물 주변의 환경을 최적으로 유지시키는 것은 매우 중요하고도 어려운 기술이다.

시판되고 있는 온실환경 자동제어기는 대부분 온도를 중심으로 한 단 요인제어기들이며, 또한 제어목표점을 사용자인 농민이 스스로 알아서 정하도록 만들어져 있다. 제어시점도 타이머방식의 단순제어를 지향하고 있어서 사계절동안 일출과 일몰의 변화에 적응하지 못하는 등의 단점이 있다. 특히 재배과정에서 수시로 발생하는 생육불균형과 생리적인 비정상상태에 적극적으로 대처할 수 없었다.

해외에서 도입된 온실환경 자동제어기도 이런 상황은 비슷하여 대부분이 온도중심의 비례제어방식을 채택하고 있는 실정이다. 현재 국내에 도입되어 사용중인 외국상품으로는 네델란드의 PRIVA사의 제품과 미국 Q-COM사의 GEM Ⅲ 등이 대표적인 경우인데 이들 제품은 다년간 축척된 개발 Know-How로 인해 제어기의 신뢰도가 국내제품에 비해 비교적 앞서고, 온도제어이외에 습도제어, CO_2제어 등을 지원하며 제어인자간의 상호관련성을 고려한 제어를 구현하였지만, 제어목표점과 제어시점의 설정은 역시 사용자의 몫에 맡기는 형편이다. 이는 다양하게 변하는 온실 내외의 기상환경으로 인하여 흔히 '제어실패'의 상황을 만들기도 한다.

따라서 진정한 의미의 복합환경제어란 재배초기(정식시점)의 사용자의 기초적인 요구만을 입력받고, 이후의 제어과정은 컴퓨터에 전적으로 맡기고 사용자는 수시로 그 상태를 점검하는 것이라 할 수 있다. 이 경우 제어목표치의 설정에 관여하는 요인으로는 첫째 온실 외부와 내부의 기상상태를 실시간으로 파악하여 이로 인한 온실내부의 미기상상태를 면밀히 해석하고 있는 것, 둘째 생산량을 최대로 높히는 것, 셋째 생산물의 품질을 극대화하는 것, 넷째 환경제어에 소요되는 제어비용을 최소화하는 것 등을 들 수 있다.

그러나 이 같은 제어방식의 완성은 작물별, 품종별, 온실의 시설조건, 작물의 재배조건에 따른 세밀한 연구가 선행되고, 이를 통해 획득한 정보를 데이터베이스화하여 시스템내부에 이식할 때만이 가능해진다고 판단된다.

본 연구는 토마토 생장과 수익성을 최대화하기 위해 온실내부의 온도, 습도, CO_2 등의 기상환경을 적극적으로 제어하는 '최적복합환경'의 구현을 목표로 하였다.

이를 위해 온실 내부와 외부의 기상환경의 변화를 실시간으로 탐지하여 이에 따른 온실내부의 미기상상태을 예측하고, 또한 온실내부의 기상값을 인위적으로 변화시켰을 때 토마토의 가상생장량을 Simulation하고 최적생장량을 얻을 수 있는 방향으로 온실내부의 기상조건을 유도해 가는 과정을 개인용 컴퓨터의 프로그램을 이용하여 구현하였다. 또한 제어에 소요된 비용정보을 누적시켜 가면서 다음단계의 제어비용을 최소화할 수 있는 제어방법을 택해가는 인공지능형 프로그램을 구현하였다.

제2절 복합환경 최적제어시스템의 구성

1. 온실 복합환경 제어시스템

온실제어용 컨트롤시스템은 설계자의 기술력과 대상작물의 종류와 품종특성, 그리고 지형특성과 온실의 내부시설 등 주변 여건에 따라 여러형태로 구성될 수 있다.

본 개발에서 설계한 시스템의 개요는 아래 Fig4.1에서 보는 바와 같다.

창문개폐기, 온풍기, 환기팬, CO_2발생기 등 온실내외에 설치된 제어기에 대한 제어가 온도, 습도, CO_2 등 기상센서 값을 근거로 이루어 지며 이 같은 제어에는 풍향, 풍속, 일사, 강우, 외부온도, 외부습도 등 외부 기상 값이 유기적으로 관여 하여 제어방향을 결정하도록 제작되었다.

본 개발에 사용된 시스템은 크게 개인용컴퓨터(PC)에 탑재된 종합관리시스템, 컨트롤러(Controller), 계전기 등 3요소로 구성되었다. 계전기는 컨트롤러의 지휘를 받아 전기적인 동작기제어를 담당하며, 컨트롤러는 PC상의 종합관리시스템의 일별명령을 받아 온실내외 기상환경을 고려하여 최적의 온실재배환경을 만들기 위한 필요한 모든조치를 취하는 일을 담당하며 또한 센서값 및 제어명령에 대한 모든 상황을 수시로 PC상의 종합관리시스템에게 전달하는 역할을 수행한다.

개인용컴퓨터와 컨트롤러는 멀티포트(Multi-port) 를 통해 다수의 온실에서 오는 각종 기상자료와 제어신호, 그리고 Limit신호 등을 주고 받는다. 컨트롤러는 컴퓨터의 제어명령을 받아 컨트롤러측의 출력보드(Output-Board)를 통해 신호를 보내고, 계장측은 릴레이보드(Relay-Board)를 통해 제어명령을 수신하여 동작기를 움직인다. 또한 온실내부에 설치된 온도센서, 습도센서, CO_2측정기에서 송신된 기상자료를 A/D Converter를 통해 Digital로 변환하여 컴퓨터에 전달한다. 외부기상 데이터(온도, 풍향, 풍속)는 Weather station을 통해 수집되며 멀티포트를 통해 컴퓨터에 자료를 송신한다. 원격지의 컴퓨터(Remote Computer)는 온실에 설치된 컴퓨터내부의 모뎀을 통해 온실상태를 수신받고, 필요한 명령을 보낼 수 있다. 경보기는 컴퓨터가 온실의 비정상 상태를 감지한 경우 시스템 운영자를 호출하는 기능을 수행한다.

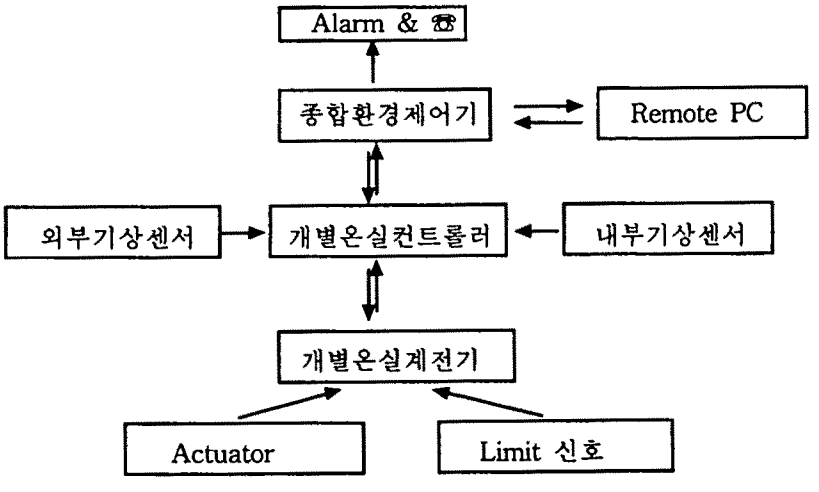


Fig. 4.1 Brief Diagram of greenhouse control system.

2. 복합환경제어용 Software

온실제어용 소트프웨어는 크게 2종류로 나누어 진다. 하나는 PC용 종합환경관리에 사용하는 프로그램이고 다른 하나는 컨트롤러용 MICROPROCESSOR에 탑재할 제어용 프로그램이다.

먼저 종합관리용 프로그램의 경우에 Windows 95용 Visual Basic(ver7.0)을 이용하여 주 시스템이 제작되었으며, PC용 TURBO-C와 Quick Basic을 이용하여 일부 모듈이 코딩되었다. 한편 사용자의 편이성을 고려하여 명령체계를 단순화하였고, 대부분의 명령이 Icon을 넣은 버튼으로 만들어져 사용자 인터페이스를 최대한 강화하였다.

프로그램의 기능적 측면에서의 개요는 다음과 같다.

제어항목은 온도, 습도, CO_2 등 3개 환경을 대상으로 하며 한 제어항목에 대해 30분 단위의 제어목표점을 설정할 수 있게하여 자동제어나 수동제어 모두 하루에 총48개 단계의 제어논리 편집이 가능하며 이는 습도 및 CO_2에 대해서도 동일하다. 또한 모든 제어항목은 제어편차범위를 설정할 수 있게 하였다.

제어기의 동작은 가장 먼저 예상되는 작물재배기간에 대해 입력을 받아야 이루어진다. 즉 재배하고자 하는 작물, 재배시작 일시, 재배종료 일시, 그리고 센서자료, 제어자료, 에러자료 등 자료저장 화일명 등을 입력하여야 기본적인 제어에 들어갈 수 있도록 되어 있다.

Table 4.1 Input parameters to initialize the greenhouse control system.

구 분	결 정 사 항	내 용
작 기 설 정	재배작물	토마토
	재배시작시점	YY/MM/DD
	재배끝 시점	YY/MM/DD
	자료저장화일명	센서자료, 제어자료, 에러자료, 제어비용자료

제어기를 통해 입력받을 센서값에 각각의 번지(ADDRESS)를 지정하기 위해 센서의 등록, 제어기의 등록 등을 사용자가 입력하는 기능을 넣었다. 사용자가 입력할 수 있는 센서의 최대 갯수는 온도 및 습도의 경우 컨트롤러 1대당 각 2개씩, 그리고 CO_2는 1개, 그리고 기타 추가 센서는 총 3개까지 입력이 가능하도록 설계하였다.

컨트롤러와 컴퓨터간의 원활한 통신과 작기단위 관리를 위해 시계 및 날짜 맞추기 기능을 삽입하였다.

컴퓨터를이용한 수동제어를 위해 수동제어용 버튼쉬트를 만들어 컨트롤러의 제어인터럽트를 빼았는 기능을 추가하였다.

제어환경설정에 가장 중요한 요소는 일출과 일몰시간인데 이를 위해 시스템이 설치되는 지점의 경도와 위도를 입력받아 자동으로 해당일자의 일출·일몰을 계산하고 디스플레이 시킨다.

불량한 온실외부환경에 대해 효과적으로 대처하기 위해 외부풍속 및 강우에 대한 창문개폐 제약조건 설정기능을 추가하였다.

저장된 센서값과 제어명령수행상황 및 에러상황 등을 작기단위 화일로 저장하여 화면이나 프린터를 통해 디스플레이 시키는 기능을 넣어 효과적인 관리체계를 구축하였다.

컨트롤러용 제어프로그램은 일차적으로 PC상의 종합관리프로그램의 명령을 받아 제어를 수행하나 별도로 몇가지 기능을 독자적으로 갖고 있다. 즉 창문 open delay time 조정을 가능하게 하는 Control Time 수정기능과 환기팬 동작을 설정온도에 의한 직접제어방식과 온도제어 Sequence에 의한 작동방식을 선택할 수 있게한 알고리즘 변경 기능, 커튼 동작을 외부광량에 의해 제어하는 Shading Control방식과 Timer에 의한 제어방식을 선택할 수 있게한 알고리즘 변경기능, 그리고 ID번호에 의한 자체 데이타 구별 신호 부가기능, 그리고 내부 감시타이머에 의한 시스템 down 방지기능 등 별도의 기능을 갖고 있는 프로그램을 개발하였다.

3. 컨트롤러(Local controller)

본 제어기는 온실제어전용기판(Printed Circuit Board,PCB)를 자체 제작하여 개발하였다. 중앙처리장치는 인텔사의 8bit형의 CPU인 i8051 모델을 사·용하였으며, 기억장치의 크기는 ROM이 64K Byte, RAM이 1M Byte로 구성하였다. 특히 RAM은 SRAM을 채용하여 전원공급 중단후에도 저장된 자료가 소멸되지 않는 특징을 갖도록 하였다. 제어기의 Display화면은 LCD를 이용하였는데 한개 글자는 5×7 Dot Matrix의 영문전용으로 구성되었으며, 한 화면에 20字를 2줄로 표현할 수 있는 크기로 만들었다.

센서신호처리를 위한 A/D converter는 12Bit방식으로 처리 하였으며 센서 입력 Channel은 Main Board상에 최대 8개까지, Limit신호등 Digital 입력 Channel은 최대 24개까지 가능하도록 설계되었다. PC와의 통신을 위해서 RS232방식의 통신Port를 채용하였으나 센서데이타 수신시 잡음(noise)가 다소 발생하여 RS422전환 Kit를 추가로 설치하였다. 키보드는 기능키(Function Key)가 F1, F2, F3, F4, F5, F6, F7, F8 등 8개, 숫자키(Number Key)가 1, 2, 3, 4, 5, 6, 7, 8, 9, 0 등 10개로 구성되었다.

제어기의 외형은 방수·방습을 위해 두껑에 고무 팩킹처리가 된 연질 Poly Carbonate 재질의 기성품케이스를 이용해 제작하였다.

Table 4.2 Specification of greenhouse controller.

구 분	품 명	규 격
내부MICOM	CPU	i8051(8bit)
	ROM	64K BYTES
	RAM	1M BYTES
	REAL TIME CLOCK	년,월,일,시각,72hr 정전보상기능
	DISPLAY	LCD채용, 20자x2행, 5x7도트, 영문전용
	A/D CONVERTER	12BIT
	입력접점	Analog : 8Channel, Digital : 24Channel
	출력접점	24Channel
	통신 PORT	RS232/RS422
	키보드	Function키 10개, 숫자키 10개
외 형	FIBOX	재질:PC, 뚜껑:투명PC, L380xW280x180xT2

4. 계전기(Relay)

본 연구개발에서도 일반농가의 제어기와 유사한 형태의 제어를 제작하였으나, 앞서 설명한 환경제어기의 신호를 계전기에 원활히 전달하고 동작기(Actuator)에서 발생하는 Limit신호를 제어기에 전달하기 위하여 Digital Relay Board를 추가로 설치하여 종합적인 제어 시스템이 되도록 하였다. 그외 천·측창 모터구동을 위해 24Volt Transformer와 개별 Magnetic Relay의 안전을 위해 Overload Relay를 설치하는 등의 구성은 일반적인 계전기와 유사하다고 할 수 있다.

5. 센서 및 Limit switch

온실내 제어요소 및 센서의 구성은 Table 4.3 에서 보는 바와 같다. 센서는 먼저 온실동별로 내부에 온도, 습도, CO_2 센서를 한개씩 설치하였고, 온실외부에 공통센서로 풍향, 풍속, 강우, 온도, 습도 센서를 각각 1개씩 설치하여 온실내외 기상자료 수집에 이용하였다.

Table 4.3 List of sensors for monitoring greenhouse climate.

구분	위치	종류	비고
입력센서	외부	풍향, 풍속, 강우, 온도, 습도(각1개)	공통센서
	내부	온도, 습도, CO_2(각1개)	온실별 개별설치

6. 경고(Alarm) 기능 및 기구

경고는 컴퓨터에서 판단한 제어목표치에 온실 내부환경이 도달치 못한 경우에 사용자에게 그 내용을 통보하여 후속조치를 기다리는 기능으로서 본 개발에서는 경고의 내용을 발생시각과 함께 모니터에 표시하고 디스크에 기록해 두며 또한 컨트롤러로 1bit의 경고신호를 보내 경광등(또는 부저)을 동작할 수 있게 하였다.

7. 동작기(Actuator)

제어요소는 일반농가의 제어요소와 유사하게 천·측창, 온풍기, 환풍기, 커튼, CO_2발생기, 점적관수 등을 설치하여 이들요소의 제어를 통해 온실내 환경을 최적화하려고 시도하였다.

Table 4.4 List of actuators to control greenhouse climate.

구분	위치	종류	비고
제어출력	내부	창문개폐기(2개), 온풍기(1대), CO_2발생기(1대), 환기팬(1개)	온실별 개별설치

제3절 복합환경제어 논리

1. 온실 재배환경제어의 개념

먼저 온실외부의 기상조건을 입력받아 온실내부의 미기상을 파악하고 이를 근거로 최적 제어조건을 계산한다. 이를 위해 온실외부의 실재 기상조건과 예상되는 기상조건, 제어에 소요되는 비용, 재배상의 제약조건 등을 고려한 최적제어조건의 계산을 하여 이를 근거로 제어기를 가동한다. 온실환경 제어를 통한 일차적인 목표는 광합성 또는 증산의 증진에 있으며, 이는 작물의 생장모델을 거쳐 최종적인 수량으로 계산된다.

광합성의 증진은 작물체에 대한 생체정보 취득단계를 거쳐 차후의 최적 제어조건 계산에 동원될 수 있다.

이 같은 온실재배환경제어의 개념도는 Fig. 4.2에서 보는 바와 같다.

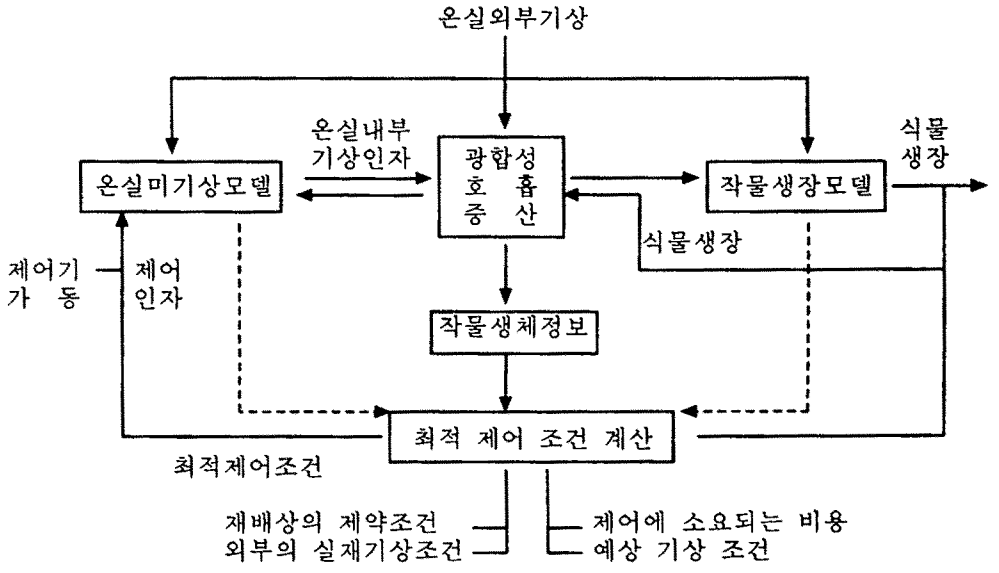


Fig. 4.2 The concept diagram for greenhouse control system.

2. 제어의 최종목표

조수익을 최대화하는 표준 모델(Seasonal Performance Criterion Formulation)은 다음과 같다. 즉 생산량을 최대화 하는 것이 일차적인 목표이고 이와함께 환경제어시에 발생한 제어비용, 특히 난방비용과 환기비용을 최소화함으로서 조수익 증대는 달성될 수 있다.

$$J = \int_0^{t_f} k(t)f(t)dt - c_h\int_0^{t_f} F(t)dt - c_v\int_0^{t_f} Q(t)dt$$

생산물 판매수입 | 난방비용 | 환기비용

t : 시간, tf : 종료시점, f : 판매가능한 생산물 생산 비율, F : heating flux
Q : ventilation flux, k, ch, cv : 생산물, 난방, 환기의 각 단위 가격

3. 환경제어용 Simulation model

온실 환경제어를 위한 논리흐름도는 Fig. 4.3에서 보는 바와 같다.

최초 시스템을 가동하면 먼저 외부기온, 외부 풍속, 외부상대습도를 입력값으로 온실 미기상 예측모형을 가동하여 온실내부의 온도와 습도, 그리고 이산화탄소의 농도를 계산한다. 이를 이용해 환경제어 목표치를 설정하는데 온도와 습도를 두 축으로 한 2차원의 가상조건 배열을 생성한 후 이를 이용하여 토마토 생육모형을 가동하게 된다. 이를 통해 얻어진 광합성값중 최대치를 얻는 온도와 습도조건을 제어목표치로 정하여 제어기를 가동하게 된다.

제어가 완료된 후 예상된 목표치에 도달하였는지 여부를 판단하여 도달치 못한 경우 다시 목표치를 변경하는 경로를 거치게 된다. 이런 과정을 되풀이하면서 제어비용이 계산되며 이는 인공지능형 모듈에 교육정보로 활용하게 된다.

최초 시스템 가동시는 제어목표치에 근접하지 못하고 불규칙적으로 진동을 거듭하는 부정형의 파형을 그리게 되나 시간이 지날수록 제어목표치에 근접하게 된다.

이 같은 과정을 통해 광합성 또는 증산량 최대의 조건과 제어비용 최소화의 두 목적을 달성할 수가 있다.

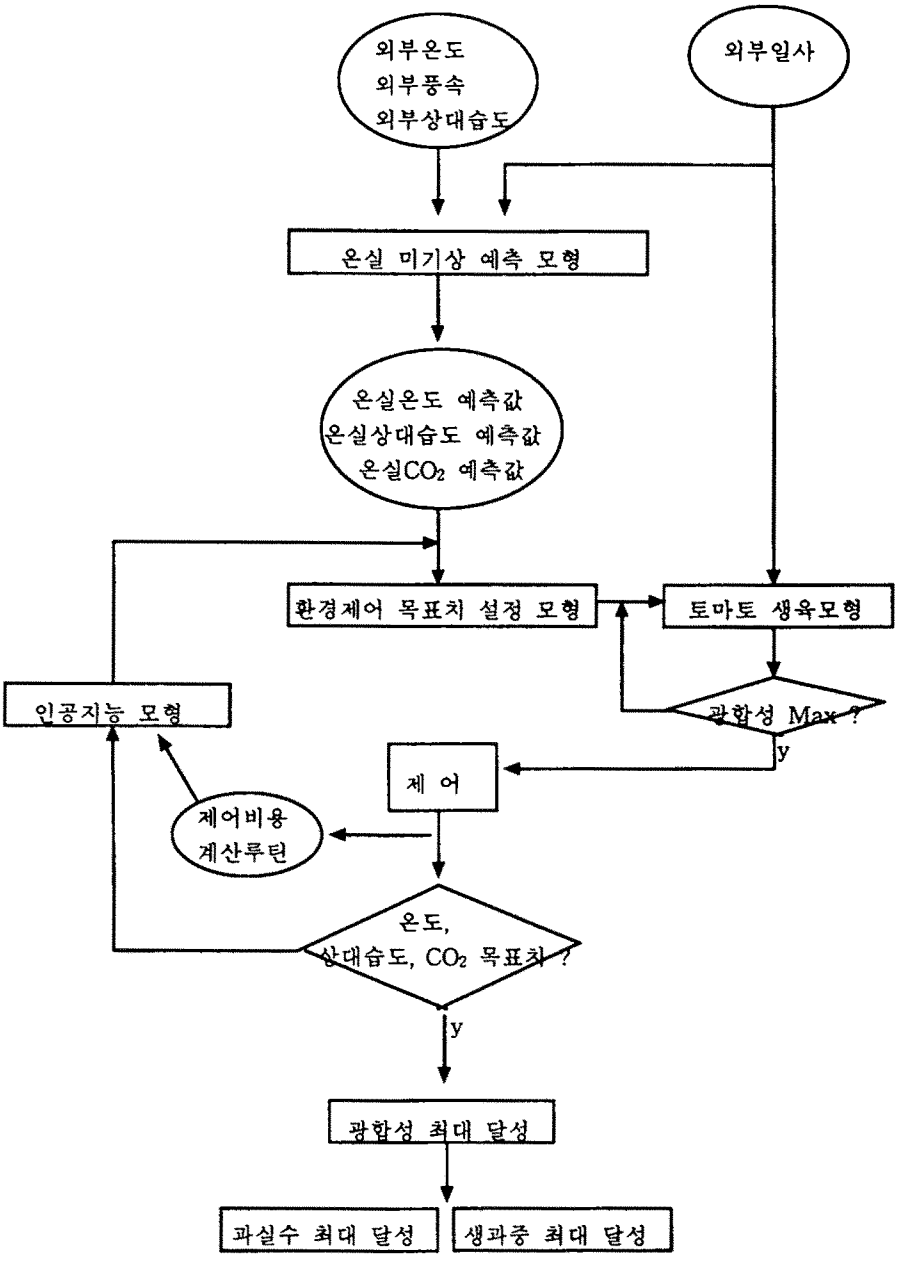


Fig. 4.3 Logical chart diagram of greenhouse control system.

가. 온실 미기상 모형

온실 복합환경 제어시스템의 가동시 최초로 동원되는 모형인 온실 미기상 모형의 입출력 변수의 종류는 Fig. 4.4에서 보는 바와 같다.

즉 날짜, 시각과 함께 온실외부의 온도, 풍속, 일사량, 상대습도 등을 입력받아 온실내부의 각종 온도값과 습도, 증산량, 증발량, 일사량 등을 계산해 준다

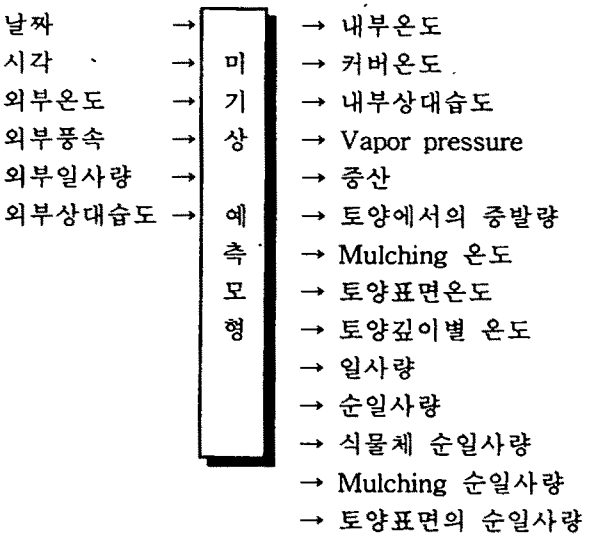


Fig. 4.4 Input and output variables of microclimate simulation model in greenhouse.

나. 토마토 생육 모형

토마토 생육모형에 동원되는 입력변수로는 먼저 최초 정식시에 토마토묘의 마디수, 엽중, 엽면적, 경중, 그리고 재식밀도 등 생육초기 값을 받으면서 모형이 시작된다. 그다음은 앞서 사용한 온실미기상모형에서 얻은 온실내부의 온도와 상대습도, 그리고 온실외부의 일사량값을 이용하여 토마토 생육모형이 가동된다. 이때 얻어지는 생장량 지표는 마디수, 경중, 엽면적, 경중 등의 생육형질값과 과실수, 생과중 등 수량형질을 모두 얻을 수 있다. 한편 simulation의 interval은 사용자가 정할 수 있는데 해당 interval마다 기상값을 입력받아 생장량을 계산해 낼 수 있다.

본 토마토 생육모형은 서광, Momotaro, Arletter품종에 한정된 것이며 다른 토마토 품종에의 적용은 해당 변수의 조정을 통해서만이 가능하다.

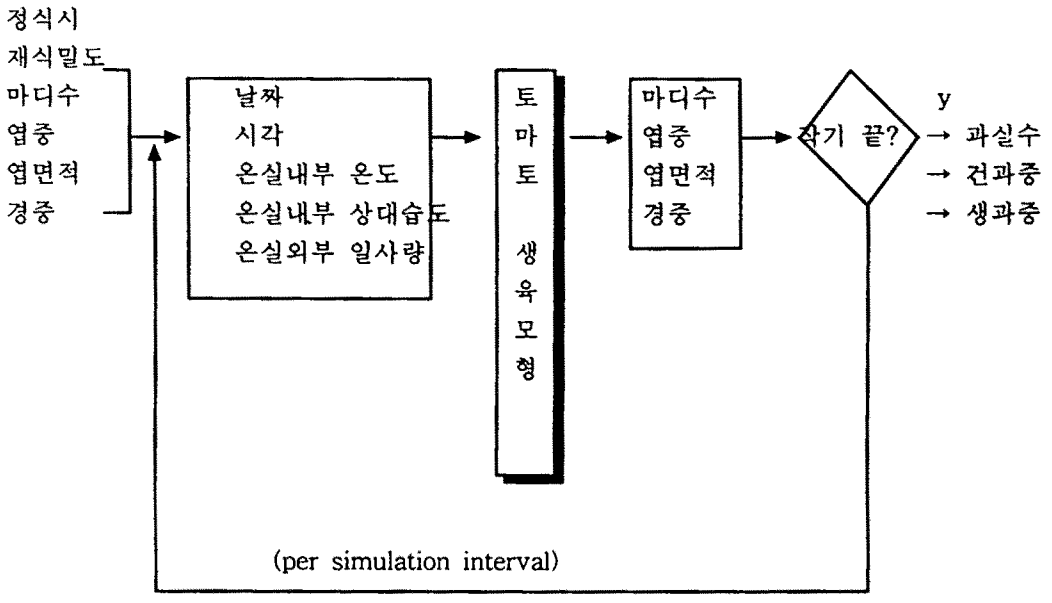


Fig. 4.5 Flow chart of tomato growth model in greenhouse.

다. 환경제어 목표치 설정모형

온실의 환경제어를 위한 제어목표치 설정에 대한 논리도는 Fig. 4.6에서 보는 바와 같다.

먼저 온실 미기상모형을 거쳐 얻은 온실내부의 온도 및 습도 값을 중심축으로, 온도의 경우 ±5℃범위에서 0.5℃간격의 계급을 정하고, 습도의 경우 ±10%범위에서 1%간격의 계급을 설정하여 20×20 matrix를 생성한다. 이들 온습도 조건을 토마토생육모형의 입력변수로 이용하여 그 결과물로서 20×20 matrix의 예상 광합성값을 얻을 수 있다.

이 2차원 matrix를 반응표면분석(Response Surface Method, RSM)을 통해 광합성을 최대화하는 온도 및 습도조건을 탐색할 수가 있다. 이를 이용해 제어를 수행하게 된다.

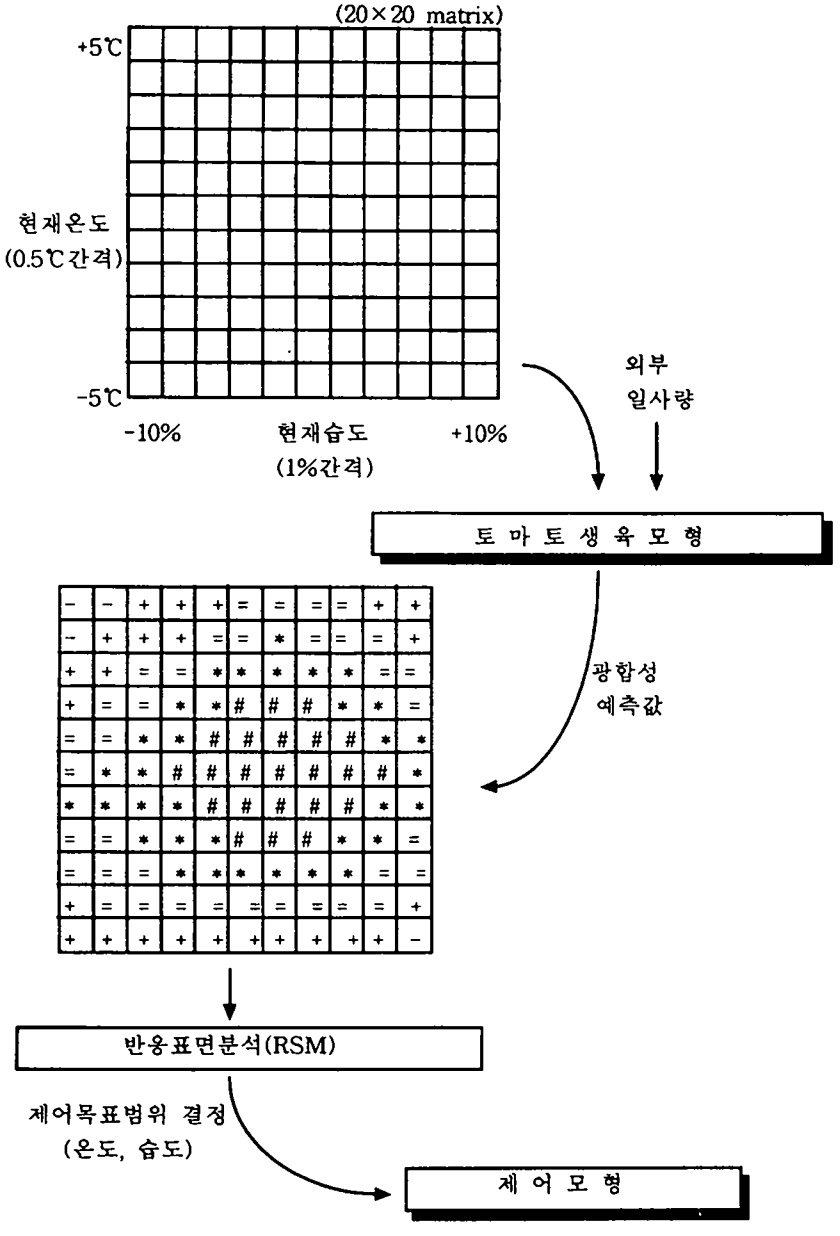


Fig. 4.6 Determination of control target point for optimization of greenhouse climate

라. 제어모형

1) 온도제어 Sequence

제어목표치가 설정되면 다음은 제어논리 모형이 가동되는데 이는 크게 온도에 대한 제어bit 생성, 습도에 대한 제어bit 생성, 주야간 결정bit 생성 등을 거쳐 전체적인 제어 sequence를 거치게 된다.

Fig. 4.7은 온도에 대한 제어bit의 생성과정을 그림으로 표시한 것이다.

먼저 제어목표치가 들어오면 현재의 온실내부의 온도와의 편차가 오차범위내에서 제어가 필요할 만큼 높은지 낮은지를 판단하여 높은 경우, 먼저 가온기를 off시키고, 환기, 창문개폐의 과정을 순차적으로 진행하게 된다.

이 경우 한가지 제어방향이 결정되면 다른 제어기의 가동은 무시되며 상위의 제어만을 수행하게 되며, 향후 제어도달치가 만족스럽지 못한 경우에 다른 제어기를 선택하거나 제어강도를 조절하게 된다.

또한 이 과정은 직접적인 제어가 이루어 지는 것은 아니며 설치된 제어기에 대한 1차원의 제어용 matrix의 해당 bit의 setting의 과정이다.

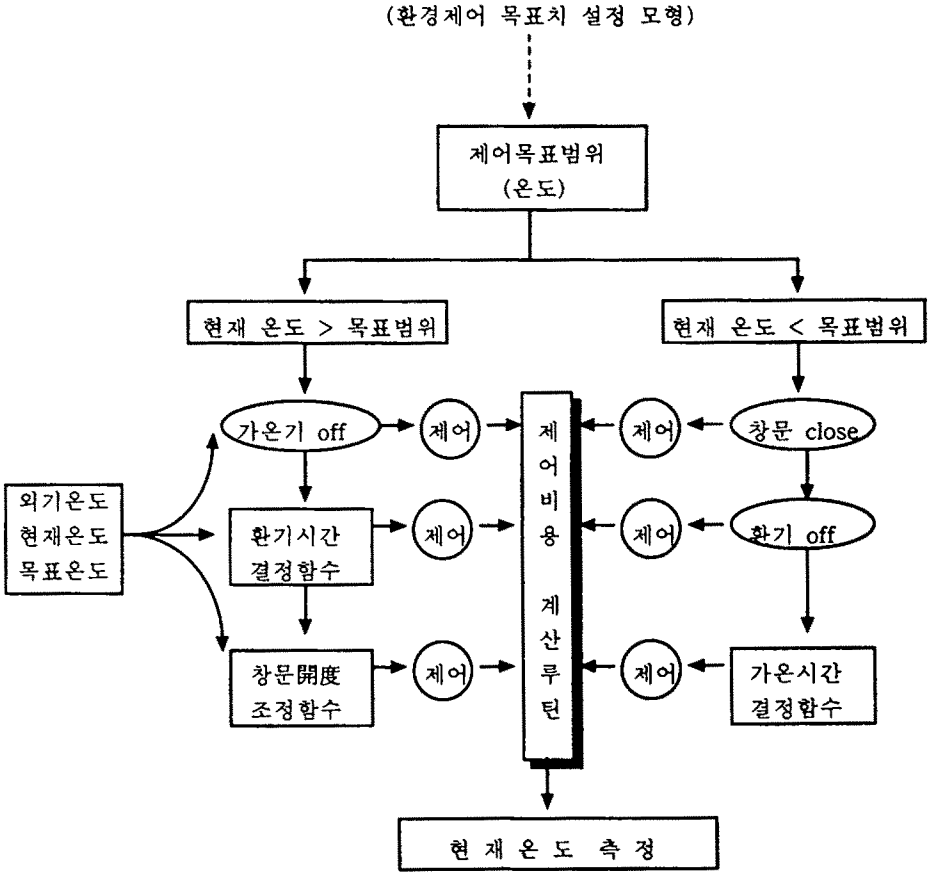


Fig. 4.7 Flow diagram of bit setting for the control of temperature in greenhouse.

2) 습도제어 Sequence

Fig. 4.8은 습도에 대한 제어bit의 생성과정을 그림으로 표시한 것이다.

온도의 경우와 마찬가지로서 먼저 제어목표치가 들어오면 현재의 온실내부의 습도와의 편차가 오차범위내에서 제어가 필요할 만큼 높은지 낮은지를 판단하여 높은 경우 환기, 가온의 과정을 순차적으로 진행하게 된다.

이 경우도 한가지 제어방향이 결정되면 다른 제어기의 가동은 무시되며 상위의 제어만을 수행하게 되며, 향후 제어도달치가 만족스럽지 못한 경우에 다른 제어기를 선택하거나 제어강도를 조절하게 된다.

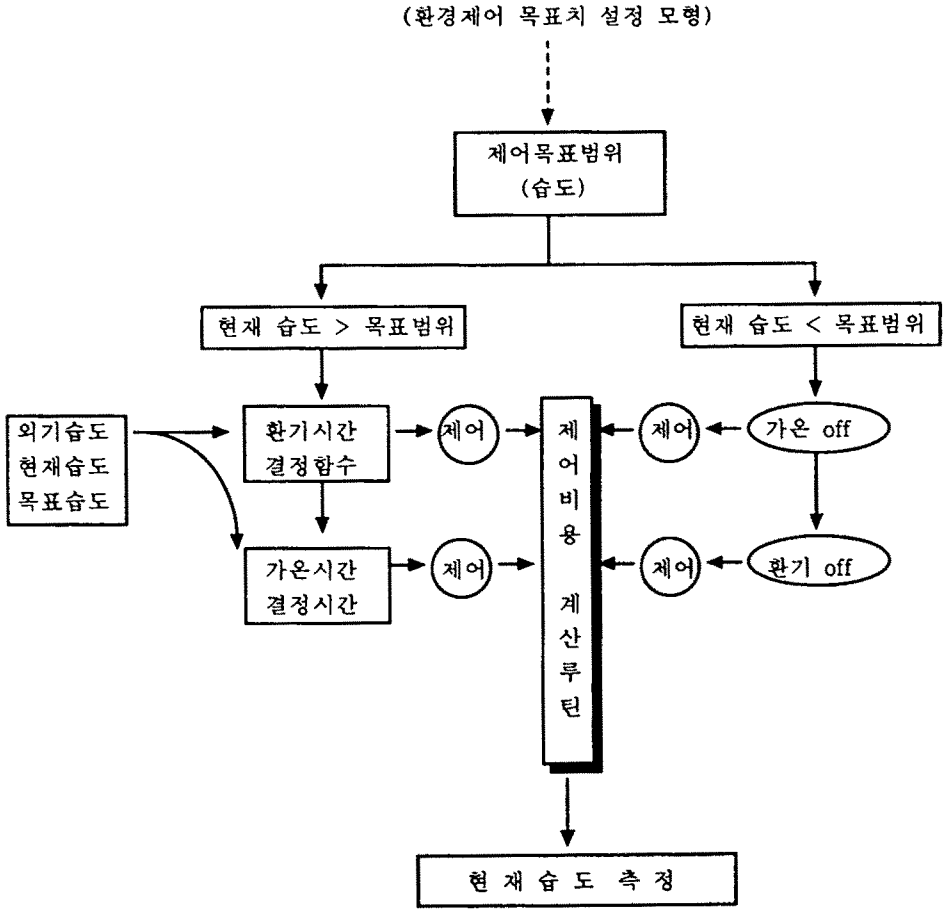


Fig. 4.8 Flow diagram of bit setting for the control of relative humidity in greenhouse.

3) 주간, 야간 제어 결정 함수

보다 효과적인 환경제어를 위해 하루를 주간제어, 야간제어, 그리고 변화기간의 제어 등 3단계로 나누어 제어가 이루어 지도록 설계하였다. 주간, 야간제어 결정함수의 논리적 모형은 Fig. 4.9에서 보는 바와 같다.

즉 일출시각과 일몰시각을 계산하여 일출후 1시간이 경과한 후부터 일몰까지를 주간 제어기간으로 간주하여 광합성 최대와 제어비용 최소의 목표로 제어를 진행하고, 일몰후 1시간후부터 다음날 일출때까지는 야간제어기간으로 간주하여 사용자가 원하는 변온조건을 입력받아 제어를 실시하게 된다. 그러나 일출후 1시간동안과 일몰후 1시간동안은 모든제어기의 동작을 중단하고 제어대기상태를 유지함으로써 제어비용을 최소화할 수 있도록 설계·제작 하였다.

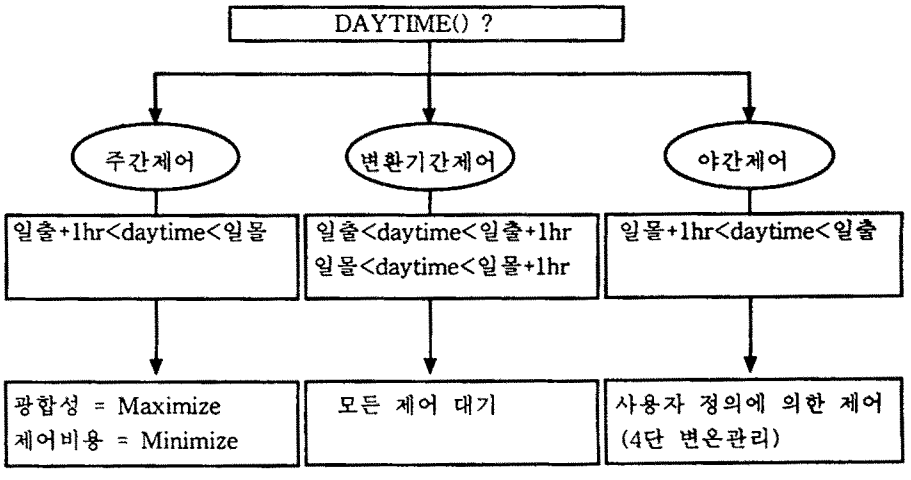


Fig. 4.9 Loical model of daytime determination for optimal control in greenhouse.

4) 전체 제어 Sequence

앞서 설명한 온도 sequence와 습도 sequence는 실제로 전체 제어 sequence의 한 부분이라고 할 수 있다. 여기서 온도, 습도, CO_2 제어를 위한 최종적인 제어방향을 결정하게 된다. Fig. 4.10은 전체 제어sequence를 나타낸 것이다.

먼저 사용자의 수동mode요구가 있었는지를 확인한다. 수동mode요구가 있으면 모든 제어는 중지되며 사용자의 명령에 따르게 된다. 자동mode인 경우에는 먼저 습도제어 sequence를 거쳐 습도제어bit를 set하고, 온도제어 sequence를 거치게 된다. 그 다음은 강우처리함수를 거치게 되는데 강우시는 위의 제어명령을 무시하고 조건없이 창문닫기를 시도하게 된다. 그다음은 수동처리함수로서 컴퓨터상에서의 수동명령 수행을 담당하게 된다.

이런 과정을 거쳐 전체 제어bit가 결정되면 비로소 일회의 제어명령이 controller로 전달되게 되는데 제어명령후에도 제어확인과정과 처리대기과정을 거치게 된다.

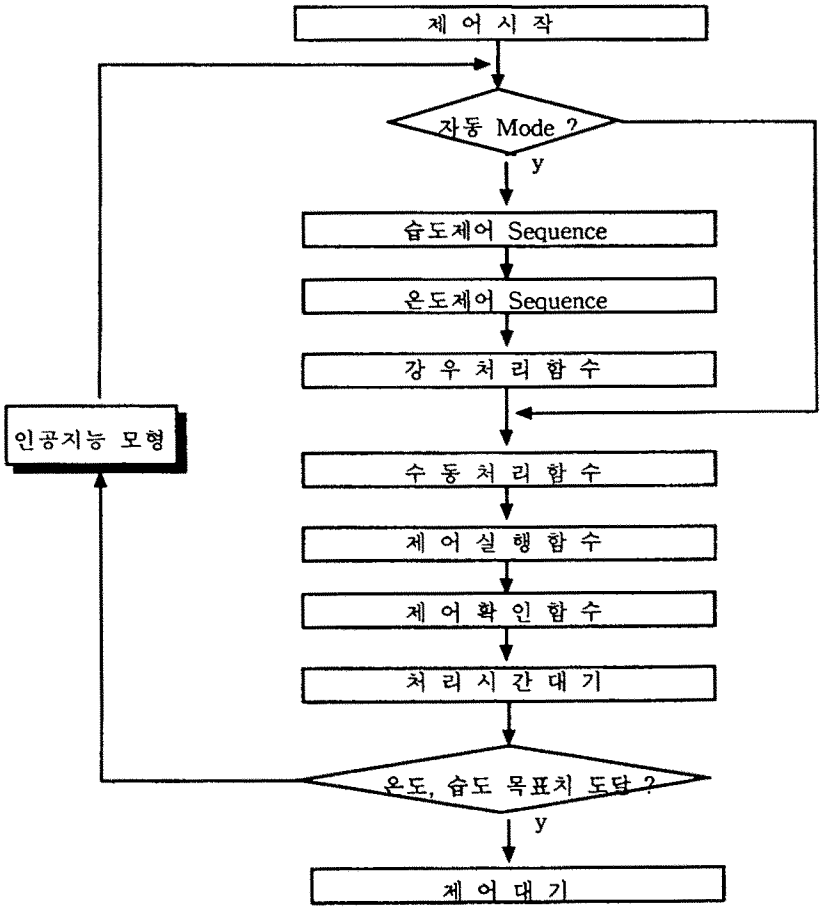


Fig. 4.10 Flow diagram of total bit setting sequence.

마. 피드백형 인공지능 모형

인공지능형 프로그램은 Lisp 프로그램과 Turbo-C언어를 이용하여 코딩하였으며 sequential control search(Garsia and others, 1989)모형에 의해 얻은 최적결과를 이용해 neural network 훈련에 이용하였다. 이를 차후의 control rule로 이용하여 제어효율을 점검하고 및 보완하였다.

제 5 장 생체정보를 이용한 토마토 최적 관개 모델의 개발

제 1절 서 언

온실은 피복재에 의하여 강우가 차단되기 때문에 노지에서와는 달리 관개는 필수 불가결하며, 관개 방법의 양부는 온실재배작물의 수량 뿐만 아니라 품질에도 지대한 영향을 미친다. 그러나 우리나라의 경우 온실에서 재배되는 작물의 관개 시기 및 관개량 결정에 대한 기준이 제대로 마련되어 있지 않다. 작물의 관개 개시 시기를 결정하는 데에는 작물 체내 수분 상태를 기준으로 하고 관개량은 증발산량에 근거하여 결정하는 것이 가장 타당하다고 한다.(Moriya et al.,1992)

식물체의 수분 상태에 근거하여 관개시기를 결정하기 위해서 식물체 수분 상태를 비 파괴적으로 monitoring할 수 있어야 하는데, 줄기 직경의 변화(Moriya et al,1992; Namken et al,1969; Schoch et al,1987,1989; Simonean et al,1993; Huck et al,1977), 줄기 정전용량의 변화(Moriya etal,1992) 등을 측정하여 이를 식물체 수분 상태 변화의 간접적 지표로 삼아 관개시기의 결정에 이용하고자하는 많은 연구가 이루어 지고 있다. 특히 경직경의 변화는 식물체 수분상태를 잘 반영하여(Simoneau et al,1993) 관개시기 결정에 이를 지표로 이용하기 위한 연구가 많으며 Moriya et al(1992)은 경직경 측정 장치를 이용하여 포도원의 관개 자동화 시스템을 실용화 하였다.

본 연구의 최종 목적은 경직경의 모니터링과 모델에 의한 온실내의 증발산량의 예측에 근거하여 온실에서 재배되는 토마토의 관개시기 및 관개량 결정 논리를 개발하고 이를 관개시스템으로 구성하여 실용화하는 것을 목표로하였다.

제 2절 온실재배 토마토의 증산모델 개발

1. 연구 목적

溫室은 露地에 비하여 日射量과 풍속이 낮고 濕度가 높아 蒸散이 감소되기 쉽다. 따라서 온실재배 토마토의 경우 칼슘 부족이 야기되거나 어린잎의 發育이 저해되어 收量이 감소하기 쉬우므로(Aikman and Houter, 1990; Bakker, 1990) 蒸散을 촉진할 수 있도록 온실 微氣候 環境을 적절히 조절하는 것이 필요하다. 이를 위해서는 온실의 微氣候 환경과 蒸散과의 관계를 정확하게 파악할 수 있는 증산 모델의 개발이 필요하며 또한 온실재배 작물의 정확한 灌漑量을 결정하는데 있어서도 정확한 蒸散量 豫測 모델은 필수적이다.

이와 같은 이유에서 온실 재배 작물의 蒸散量 豫測 및 이를 이용한 온실 미기후 조절과 관개 제어에 관한 연구가 활발하게 이루어지고 있으나 우리 나라에서는 이 분야의 연구가 거의 이루어지지 않고 있는 실정이다. 蒸散 모델은 蒸散에 크게 영향하는 要因인 日射量, 대기의 飽差, 風速 등과 蒸散量과의 回歸式을 誘導한 經驗的 모델(de Graaf and van den Ende, 1981; FAO, 1977; Okuya and Okuya, 1988)과 作物 群落의 熱 및 物質 收支에 근거한 理論的 모델(Aikman and Houter, 1990; Boulard et al, 1991; Jolliet and Bailey, 1992; Jolliet, 1994; Papadakis et al, 1994; Stanghellini, 1987)로 대별할 수 있다. 經驗 모델은 氣候條件, 作物의 發育 段階, 作物의 栽植方法 등이 달라지면 이용할 수 없고 각 조건에 따라서 係數를 다시 추정하여야 하는 문제점이 있다(Jolliet and Bailey, 1992). 熱收支式에 근거한 모델들은 기본적으로 Penman-Monteith식과 類似하며, 각 모델들은 純輻射量(Rn), 葉面境界層擴散抵抗(r_a), 氣孔擴散抵抗(r_s)을 계산하는 방법이 다른데 이중 r_s가 가장 민감한 變數이다. r_s의 계산 방법과 모델의 계수 추정에 이용한 環境 範圍에 따라서 全體 모델의 適用 範圍가 달라진다. Stanghellini(1987)모델을 Jolliet 과 Bailey(1992)가 영국의 Silsoe에서 檢證한 결과 蒸散量 推定의 정확도가 매우 높다고 하였으나, Papadakis 등(1994)과 Pieters(1995)는 Stanghellini(1987) 모델은 증산량 추정의 정확도가 매우 낮다고 하였는데 그 이유가 r_s 計算 모델의 係數 추정이 이들의 實驗條件보다 낮은 日射 條件에서 이루어졌기 때문이라고 하였다. 한편 위에서 언급한 6개의 이론 모델을 Lee(1996)가 우리 나라 여름의 온실 기후 조건에서 검증한 결과 우리 나라에서 그대로 적용하기에는 모든 모델이 문제점이 있다고 하였다. 예를 들어 Stanghellini의 모델은 흐린 날에는 증산량을 정확하게 추정하였으나 맑은 날의 경우는 실측치보다 58% 나 증산이 낮게 추정된다고 하였다..

따라서 본 연구에서는 온실재배 토마토 군락의 열수지 및 물질 수지에 근거한 증산모델을 구축하고 실험을 통하여 모델에 필요한 계수의 추정과 모델의 검증을 수행하고자 하였다.

2. 모델의 설계

가. 모델의 개요

토마토의 단위 엽면적당 에너지 수지는 잎에 의한 貯熱(heat storage)을 무시하면 다음과 같이 나타낼 수 있다. Papadakis 등(1994)은 토마토 잎의 저열은 낮의 경우 $\pm 10 W/m^2$이내이고 야간에는 0에 가깝다고 하였으며 또한 Stanghellini(1987)도 토마토 군락에 의한 저열은 다른 에너지 수지항에 비하여 매우 적다고 하였다.

$$Rn = H + lE \quad (1)$$

여기서, Rn: 순복사(W/m^2), H: 현열교환(W/m^2), lE: 잠열교환(W/m^2)을 의미한다. 온실 공기와 토마토잎 사이의 현열교환(H)은 다음의 식으로 표현할 수 있다(Monteith and Unsworth,1990).

$$H = \frac{\rho_a C_p (T_l - T_a)}{r_{ah}} \quad (2)$$

여기서 ρ_a: 공기의 밀도(kg/m^3), C_p: 공기의 정압비열(J/kg/K), T_l: 엽온(K), T_a: 기온(K), r_{ah}: 열의 엽면경계층확산저항(s/m) 이다.

온실 공기와 토마토 잎 사이의 잠열교환(lE)은 수증기의 엽면경계층 및 기공 확산저항이 직열관계에 있는 것으로 가정하여 다음의 식으로 표현할 수 있다(Monteith and Unsworth,1990)..

$$lE = \frac{\rho_a C_p}{\gamma} \cdot \frac{e_{ls} - e_a}{r_{av} + r_s} \quad (3)$$

여기서, γ: 건습계상수(Pa/K), e_{ls}: 엽온에서의 포화증기압(Pa), e_a: 대기의 수증기압(Pa), r_{av}: 수증기의 엽면경계층확산저항(s/m), r_s: 기공확산저항(s/m)이다.

한편 엽-대기수증기압차(leaf-air vapor pressure deficit, LVPD)는 포화 수증기압 곡선으로부터 포차 및 엽온과 기온의 차로 다음과 같이 나타낼 수 있다.

$$e_{ls} - e_a = (e_{as} - e_a) + \delta(T_l - T_a) \quad (4)$$

여기서, e_{as}: 온실 공기의 포화수증기압(Pa), δ: 포화수증기압곡선의 기울기(Pa/K)이다.

Monteith와 Unsworth(1990)에 의하면 $r_{av}/r_{ah} = (\kappa/D_v)^{0.67} = 0.93$(여기서 κ, Dv는 각각 대기중 열 및 수증기 확산계수)이고, 위의 (1) - (4)식으로부터 증산잠열교환과 엽온 계산식을 유도하면 다음과 같다.

$$lE = \frac{\frac{\delta}{\gamma} Rn + \frac{\rho_a C_p}{r_{ah}} (e_{as} - e_a)}{0.93 + \frac{\delta}{\gamma} + \frac{r_s}{r_{ah}}} \quad (5)$$

$$T_l - T_a = \frac{\frac{0.93r_{ah} + r_s}{\rho_a C_p} Rn + \frac{1}{\gamma}(e_{as} - e_a)}{0.93 + \frac{\delta}{\gamma} + \frac{r_s}{r_{ah}}} \quad (6)$$

증산을 계산할 때 (3)식 대신에 (5)식을 이용하면 엽온을 측정할 필요가 없기 때문에 유리하며 또한 (6)식으로부터 엽온을 동시에 추정할 수 있다.

토마토 군락내의 모든 잎의 확산 저항은 병렬 관계에 있으며 또한 잎 표면과 이면의 저항이 같고 병렬 관계가 있는 것으로 가정하여(Boulard et al, 1991; Papadakis et al, 1994; Stanghellini, 1987) 엽면적지수(LAI), 엽면경계층확산저항(r_{ah}) 및 기공확산저항(r_s)으로부터 군락의 엽면경계층 확산저항($r_{ah,c}$)과 기공확산저항($r_{s,c}$)을 다음과 같이 계산하였으며,

$$r_{ah,c} = \frac{r_{ah}}{2LAI} \quad (7)$$

$$r_{s,c} = \frac{r_s}{2LAI} \quad (8)$$

군락의 증산과 온도는 위의 관계와 (5), (6)식으로부터 유도하면 다음과 같다

$$lE = \frac{\frac{\delta}{\gamma} Rn + \frac{2LAI\rho_a C_p}{r_{ah}}(e_{as} - e_a)}{0.93 + \frac{\delta}{\gamma} + \frac{r_s}{r_{ah}}} \quad (9)$$

$$T_c - T_a = \frac{\frac{0.93r_{ah} + r_s}{2LAI\rho_a C_p} Rn + \frac{1}{\gamma}(e_{as} - e_a)}{0.93 + \frac{\delta}{\gamma} + \frac{r_s}{r_{ah}}} \quad (10)$$

위의 (9), (10)식에서 Rn은 (5), (6)식과는 달리 군락의 순복사(W/m^2)이며, T_c는 군락의 표면 온도(K)이다. 모델에 의한 군락 증산 예측에는 군락 온도 실측치를 이용하지 않고 (9), (10)식을 반복 계산하여 증산과 엽온을 동시에 추정한다..

나. 군락 순복사

군락의 증산량 예측에서 가장 중요한 것 중의 하나는 군락의 순복사량(Rn)을 정확하게 추정하는 것이다. 본 모델에서 군락의 순복사는 반투명 매질의 복사교환 이론식(Stanghellini, 1987; Stanghellini and de Jong, 1995)을 이용하여 다음과 같이 계산하였다.

$$Rn = F\{(1 + \tau_s\rho_g)[(1-\tau_s) - (1-\tau_l)\rho_c]R_s + (1 - \tau_l)\sigma(\varepsilon_u T_u^4 + \varepsilon_d T_d^4 - 2\varepsilon_c T_c^4)\} \quad (11)$$

여기서 R_s: 온실내의 일사량(W/m^2), ρ_u, ρ_c: 각각 멀칭한 프라스틱필름 및 토마토 폐쇄 군락의 반사율, ε_u, ε_d, ε_c: 각각 군락 상부 및 하부 표면으로 장파복사를 사출하는 복사체의 복사능과 군락의 복사능, T_u, T_d,: 각각 군락 상부 및 하부 표면으로 장파를 사출하는 복사체의 온도: τ_s, τ_l: 각각 단파복사와 장파복사의 투과율이다. 모델에서 T_u와 T_d는 각각 온실 피복재 내부 표면온도 및 지표면의 온도로 하여야 하나 이들을 측정하거나 예측하는 어려움을 피하기 위하여 온실의 기온으로 대치하고 실험을 통하여 ε_u와 ε_d를 결정하여 이용하였다. ρ_g와 ρ_c는 각각 0.09와 0.12를 이용하였다. 모델에 이용한 ε_u와 ε_d는 각각 0.94와 0.93이었으며 군락의 복사능 ε_c는 0.95를 이용하였다. 단파복사와 장파복사의 군락 투과율은 엽면적지수(LAI), 단파복사의 흡광계수(k_s) 및 장파복사의 흡광계수(k_l)로부터 다음과 같이 계산된다.

$$\tau_s = e^{-k_s LAI} \quad (12)$$

$$\tau_l = e^{-k_l LAI} \quad (13)$$

군락의 흡광계수 k_s와 k_l은 Stanghellini(1987)가 토마토 군락에 대하여 측정한 값인 0.48과 0.64를 각각 사용하였다. 한편 (10)식에서 F는 條植을 하는 작물에서 이랑의 형태에 따른 군락의 흡광 감소를 보정하기 위한 변수로서 다음과 같이 계산된다(Stanghellini, 1987; Stoffers, 1975).

$$F = [1 - 0.07(1 - b)(4.7 - a)]\frac{1 - 0.64\tau_l^{1.18b} - 0.36\tau_l^{2.75b}}{1 - 0.64\tau_l^{1.18} - 10.36\tau_l^{2.75}} \quad (14)$$

여기서, a와 b는 각각 군락의 높이와 이랑폭의 인접이랑 중앙부간의 거리에 대한 비이다.

다. 확산저항

위의 모델로 증산을 예측하기 위하여는 엽면경계층확산저항(r_{ah})과 기공확산저항(r_s)를 정확히 평가하여야 한다. 엽면경계층확산저항은 Nusselt number(Nu)와 다음과 같은 관계가 있다(Monteith and Unsworth,1990)..

$$r_{ah} = \frac{d}{\lambda N_u} \quad (15)$$

여기서, κ는 공기의 열확산계수(m^2/s), d는 토마토 잎의 특성길이(m)이다.

강제 대류의 경우 Nu는 Reynold's number(Re) 및 Prandtle number(Pr)로 다음과 같이 나타내었다(Monteith and Unsworth,1990)..

$$N_u = 0.66Re^{0.5}Pr^{0.33} \quad (16)$$

자유 대류의 경우 Nu는 Grashof number(Gr)에 의하여 다음과 같이 나타내었다(Monteith and Unsworth,1990; Parkhurst et al, 1968).

$$N_u = 0.24Gr^{0.25}, \quad \text{for } T_l < T_a$$
$$N_u = 0.37Gr^{0.25}, \quad \text{for } T_l > T_a \quad (17)$$

온실 작물의 엽면 경계층내에서는 혼합 대류가 주류를 이루므로 엽면 경계층저항은 자유대류에서의 저항($r_{a,fr}$)과 강제대류에서의 저항($r_{a,fd}$)을 다음과 같이 병렬관계로 계산하였다.

$$r_{ah} = \frac{r_{a,fd}r_{a,fr}}{r_{a,fd} + r_{a,fr}} \quad (18)$$

기공확산저항은 일사량, LVPD, 엽온, CO_2농도, 엽수분포텐셜 등에 의하여 좌우되는 것으로 알려져 있다(Boulard et al, 1991; Jolliet and Bailey, 1992; Jolliet, 1994; Papadakis et al, 1994; Stanghellini, 1987). 그러나 본 모델에서는 이들 중 가장 영향도가 큰 일사량과 LVPD만을 이용하여 다음과 같이 표현하였으며 실측한 기공확산저항, 일사량, LVPD 등의 자료를 이용하여 상수들을 추정하였다.

$$r_s = r_{sm}[1 + e^{-a_1(Rs - a_2)}][1 + e^{b_1(LVPD - b_2)}] \quad (19)$$

여기서, r_{sm}: 최소기공확산저항(s/m), Rs: 온실내의 일사량(W/m^2), LVPD: 엽-대기 수증기압차(hPa), a1, a2, b1, b2: 상수이다.

3. 증산 및 온실 미기상 측정 실험

이 실험은 서울대학교 부속 실험농장에 위치한 南北棟의 Venlo型 유리온실에서 1996년에 실시되었다. 토마토 品種 瑞光을 1월 17일 播種 育苗하여 3월 19일 플라스틱 箱子[(27cm(W) x 44.5cm(L) x 30cm(D)]에 正式하여 栽植密度가 8株/m^2가 되도록 配置하였고, 토양 증발을 막기 위하여 투명 플라스틱 필름을 멀칭하였다. 물관리는 点滴灌漑 시스템으로 土壤水分張力이 0.2bar 이하가 되도록 관개하였다. 온실의 환경 관리는 야간온도의 설정치를 15°C로 하였으며 晝間에는 25°C 이상이 되면 天窓을 열어 換氣가 되도록 하였다.

증산량은 群落의 중앙에 위치한 2 個體에 대하여 測定하였으며, 줄기의 熱收支(stem heat balance)를 이용하는 sapflow meter(model SGB10 & SGB13, Dynamax Inc, Houston, TX)를 줄기의 基部 근처에 설치하여 측정하였다. 추가로 sensor의 heater 중앙부에 위치한 줄기의 表皮 약1mm 깊이에 thermocouple을 설치하여 줄기의 온도를 동시에 측정하하였으며, 이를 줄기의 heat storage를 補正하여 증산량을 계산하는데 이용하였다(Grime et al,1995). 10초마다 증산을 측정하여 30분간 평균치를 증산 모델의 검정에 이용하였다. 측정 전에 물을 충분히 관개하였으며, 측정 개체 및 인접 개체들의 葉長(LL, cm)과 葉幅(LW, cm)을 측정하여 葉面積(LA)을 다음 식(이변우, 미발표)에 의하여 계산하였다.

$$LA(cm^2) = 5.09 + 0.287LL*LW + 5.347*(LL*LW/1000)^3 \quad (n=300,\ R^2 = 0.9037) \quad (20)$$

2 개체에 대하여 측정한 증산량과 葉面積을 이용하여 床面 1m^2당 群落 蒸散量을 계산하였다.

蒸散量 측정 이외에 온실 내부의 日射量(RSu), 群落 下部의 日射量(RSd), 群落의 반사율(ρ_c), 群落 上部의 純輻射量(Rnu), 下部의 純輻射量(Rnd), 온실 중앙의 군락 하부, 중앙부, 상부 세 곳에서의 乾濕球 溫度, 葉溫(infrared thermometer), 風速(hot wire anemometer) 등을 10초 간격으로 측정 10분간 평균을 하여 이용하였다. 군락의 純輻射量(Rn)은 다음과 같이 계산하였다.

$$Rn = (Rnu - Rnd) + (\rho_c - \rho_s)Rsu \quad (21)$$

여기서 ρ_c, ρ_s는 각각 군락과 床面의 反射率이다.

한편 기공확산저항은 증산량, 엽온, 대기의 증기압 등 실측치를 이용하여 다음과 같이 계산하였다.

$$r_s = -0.93r_{ah} + \frac{\rho_a C_p(e_{ls} - e_a)}{\gamma \cdot lE} \quad (22)$$

4. 결 과

가. 기공확산저항

그림 1은 1996년 7월18일부터 20일까지 3일 동안 10분 간격으로 측정한 증산량, 엽온, 수증기압 등의 자료로부터 식 (22)를 이용하여 계산한 기공확산저항과 온실 내부 일사량과의 관계를 나타낸 것이다. 기공확산저항은 기존의 보고들(Boulard et al, 1991; Papadakis et al, 1994; Stanghellini, 1987)과 마찬가지로 일사량 증가에 따라서 급격하게 감소하였으며, 일사량 100W/m^{2}이 되면 일사량에 따른 기공확산저항의 차이는 크지 않고 일사량 이외의 환경요인 들에 의해서 지배되는 것으로 판단된다.

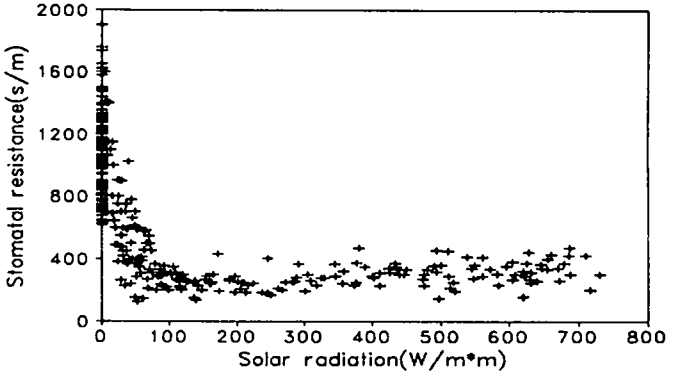


Fig. 1. Stomatal resistance of tomato plotted against solar radiation in greenhouse

자연 상태에서는 일사량, 기온, 수증기압 등은 서로 상관을 가지고 변하므로 이들 요인을 구별하여 기공확산저항에 미치는 영향을 파악하기는 매우 어렵다. 따라서 그림 1에서 보는 바와 같이 일사의 영향이 적을 것으로 판단되는 일사량 200W/m^{2}이상일 때의 자료만을 이용하여 LVPD와 기공확산저항과의 관계를 나타낸 것이 그림 2이다. 자료가 분산되어 있기는 하나 LVPD가 증가함에 따라서 기공확산저항이 증가하는 분명한 경향을 보여 기존의 보고들(Boulard et al, 1991; Papadakis et al, 1994; Stanghellini, 1987)과 같은 결과였다.

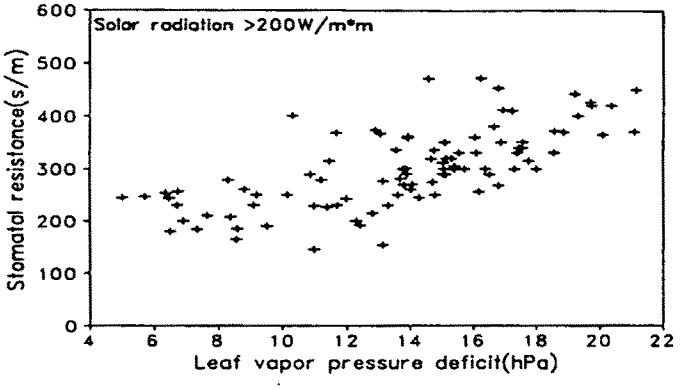


Fig. 2. Stomatal resistance of tomato plotted against leaf vapor pressure deficit.

모델에 의하여 증산을 정확하게 계산하기 위하여는 환경조건의 변화에 따른 기공확산저항의 변화를 정확하게 예측할 수 있어야 한다. 지금까지 증산량 예측 모델이 다수 개발되었으나(Aikman and Houter, 1990; Boulard et al, 1991; Jolliet and Bailey, 1992; Jolliet, 1994; Papadakis et al, 1994; Stanghellini, 1987) 이 모델들을 개발된 지역이 아니라 다른 지역에서 적용하는 경우 증산을 정확하게 예측하지 못하는 경우가 대부분인데, 그 이유는 기공의 확산저항 모델의 계수 추정 기후 조건이 적용지의 기후조건과 상이하기 때문이다(Lee, 1996; Papadkis et al, 1994; Peters, 1995). 일사, 엽온, LVPD, CO_2 농도 등이 기공확산저항에 영향을 하나 본 모델에서는 일사와 LVPD만을 고려하여 식(19)의 계수를 추정하였으며 그 결과는 표 1과 같다. 자연상태에서는 LVPD와 엽온간의 colinearity가 매우 강하여 이들 효과를 분리하여 고려하기가 매우 어려우며 또한 LVPD를 통하여 기공확산저항에 대한 엽온의 영향이 반영되고 탄산가스를 시용 하지 않는 경우 자연적인 CO_2 농도 변화 범위에서는 기공확산저항에 대한 영향이 크지 않기 때문에 모델에서 엽온과 CO_2농도를 제외시켰다. Stanghellini(1987)에 의하면 CO_2농도가 200vpm에서 500ppmv으로 증가하는 경우 기공확산저항은 약5% 증대한다. 표 1에서 보는 바와 같이 모든 계수가 통계적으로 유의하며 또한 일사량과 LVPD를 이용한 본 모델은 기공저항 변화의 80%이상을 설명할 수 있는 것으로 나타났다.

Table 1. Parameter estimates and statistics of stomatal resistance model(Eqn. 19).

Climatic variable	Parameter	Estimate	Standard error	t-value
	r_{sm}	79.4	5.2	15.3**
Solar radiation(W/m^2)	a_1	2.34×10^{-2}	1.48×10^{-3}	15.8**
	a_2	76.6	6.9	11.1**
Leaf vapor pressure	b_1	7.90×10^{-2}	5.30×10^{-3}	14.9**
deficit(hPa)	b_2	2.8	2.70×10^{-1}	10.4**
R-square = 0.8020,	Standard error of estimated stomatal resistance = 182s/m			

기공확산저항의 실측치와 모델에 의하여 계산한 값의 경시적 변화를 10분 간격으로 나타낸 것이 그림 3이다. 낮 동안에는 실측치와 계산치간에 잘 일치하나 밤에는 계산치가 실측치보다 낮게 추정되는 경향이다.

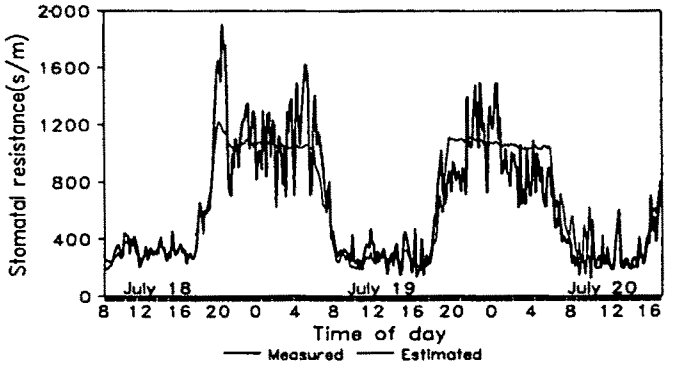


Fig. 3. Diurnal courses of measured and estimated stomatal resistance of tomato in glasshouse on three consecutive days in 1996

그림 4와 5는 (9), (10)식, 기공확산저항 모델[식 (19)]의 계수 추정치(표1) 및 실측한 군락 순복사를 이용하여 계산한 증산속도 및 군락 온도의 경시적 변화를 실측치와 대비시켜 10분 간격으로 나타낸 것이다. 증산을 측정한 7월 18일은 매우 맑은 날이었으며 19일은 다소 구름이 낀 날이었고 20일은 매우 흐린 날이었다. 그림에서 보는 바와 같이 18일과 19일은 모델로 계산한 증산속도와 실측치간에 잘 일치하였으나 19일에는 한낮의 경우 실측치 보 다소 높게 추정되는 경향을 보이고 있다. 한편 군락온도의 경우는 18일 야간에 다소 높게 추정되었으나 그 이외의 시간에는 실측치와 잘 일치하였다.

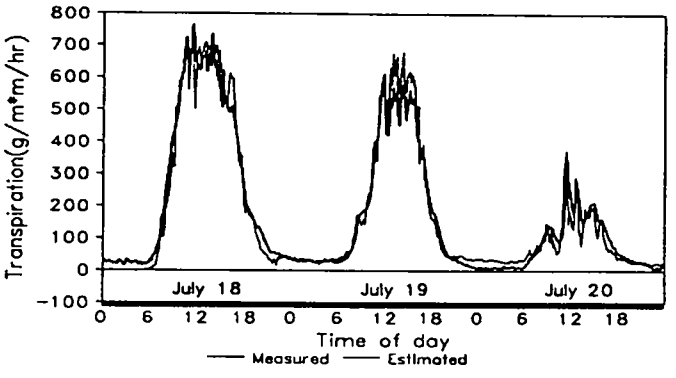


Fig. 4. Diurnal courses of transpiration fluxes of tomato measured and estimated by the present model in glasshouse on three consecutive days in 1996. Measured net radiation of canopy were used for the calculation.

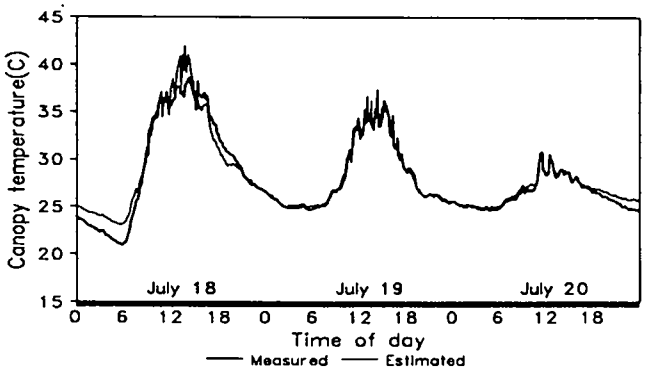


Fig. 5. Diurnal marches of measured and estimated canopy temperature of tomato in glasshouse on three consecutive days in 1996. Measured net radiation of canopy were used for the calculation.

나. 기공확산저항모델의 검증

기공확산저항 모델의 계수 추정에 이용하지 않았던 자료와 실측한 군락의 순복사량과 온도를 이용하여 계산한 10분간격의 증산 속도 및 일 총증산량을 실측치와 대비한 것이 각각 그림 6과 그림 7이다. 증산속도의 실측치와 추정치간의 상관은 r=0.962로서 양자간에 잘 일치하였으며, 또한 일 총 증산량의 경우도 추정 정확도가 매우 높았다. 따라서 본 연구에서 개발된 기공확산저항모델은 실시간 및 일 총증산량을 예측하는 증산모델의 구성식으로 이용이 가능한 것으로 판단된다.

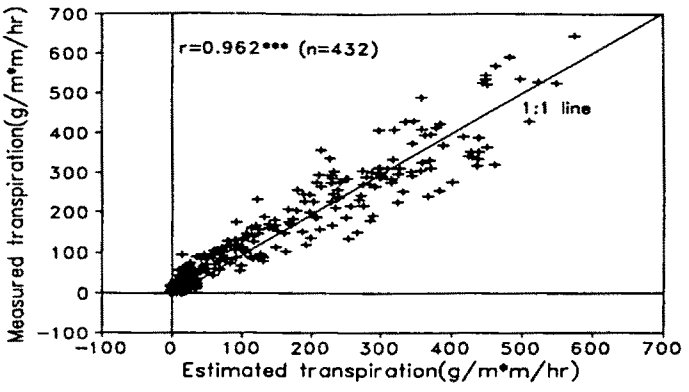


Fig. 6. Comparison of hourly transpiration fluxes measured and estimated on 10-minute intervals on three days. Data are independent of those used in the parameter estimation of stomatal resistance model, and the measured net radiation and canopy temperature were used for model calculation.

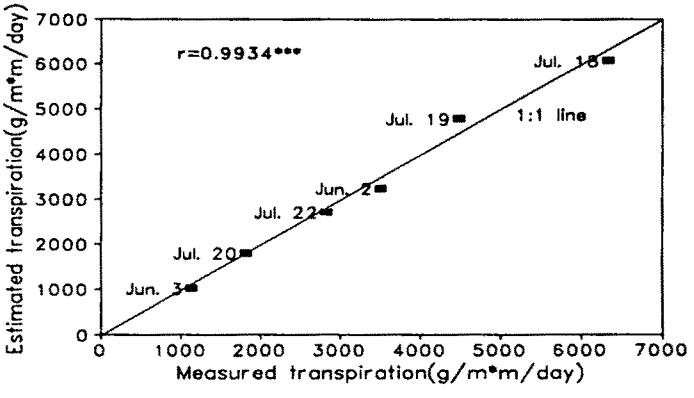


Fig. 7. Comparison of daily total transpiration measured and estimated by the model. Data are independent of those used in the parameter estimation of stomatal resistance model, and the measured net radiation and canopy temperature were used for model calculation.

다. 증산 모델의 검증

증산 모델을 실용적으로 이용할 수 있기 위해서는 모델에 필요한 입력 자료를 최소한으로 줄여야 하며 또한 모델의 변수 추정에 이용되지 않은 자료를 가지고 다양한 조건에서 검증되어야 한다. 증산량을 예측하는데 있어서 가장 중요한 것 중의 하나는 군락의 순복사량을 정확하게 평가하는 것이다. 그러나 군락의 순복사량을 관측하여 모델의 입력 변수로 이용하는 것은 관측의 어려움 등으로 인하여 현실성이 없다. 따라서 본 모델에서는 일반적으로 관측을 하고 있는 일사량 및 기온을 입력변수로 하여 (11)식에 의하여 순복사량을 계산하였으며 또한 군락 온도 역시 실측치 대신에 (10)식에 의한 추정치를 이용하였다. 본 모델에서 순복사량[(11)식], 기공확산저항[(19)식] 및 엽면경계층확산저항[(18)식] 등은 군락 온도의 함수이고 또한 군락 온도[(10)식]은 이들의 함수이기 때문에 해석적으로 모델을 풀 수 없고 수치 해석을 하여야 한다. 본 연구에서는 반복 계산에 의하여 군락 온도를 먼저 계산하고 이를 이용하여 증산량을 계산하였다. 즉 군락 온도의 초기치를 가지고 (11), (18), (19)식에 의하여 각각 순복사량, 엽면경계층확산저항, 기공확산저항을 계산하고 이들을 (10)식에 대입하여 군락 온도를 계산하는 과정을 반복하여 전 단계와 현 단계에서 계산된 엽온의 차가 0.001K 이하가 되면 반복 계산을 멈추고 현 단계에서 계산된 군락 온도, 순복사량, 기공 및 엽면경계층확산저항을 (9)식에 대입하여 증산속도를 계산하였다. 이와 같이 하여 계산된 군락의 순복사량, 온도, 증산속도 및 일 총증산량을 실측치와 대비한 것이 각각 그림 8, 9, 10, 11이다. 한편 본 모델의 검증에는 기공확산저항모델의 계수 추정에 이용한 날의 자료는 제외하였다. 그림 8에서 보는 바와 같이 군락 순복사의 실측치와 모델 추정치는 매우 잘 일치하였으며, 또한 그림 9, 10, 11에서 보는 바와 같이 군락 온도, 증산속도 및 일 총증산량 또한 실측치와 매우 잘 일치할 뿐만 아니라 실측한 군락의 순복사량 및 군락을 모델의 입력 변수로 하여 증산을 예측한 경우(그림 6, 7)와 비교하여도 추정의 정확도가 떨어지지 않았다. 따라서 본 연구에서 개발된 증산 예측모델은 온실 환경의 제어나 관개 제어를 위한 구성 모델로 충분히 이용될 수 있을 것으로 판단되었다.

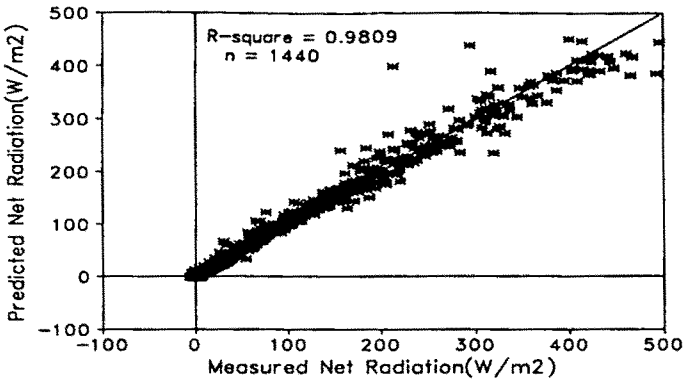


Fig. 8. Comparison of canopy net radiations measured and estimated at 10 minutes interval for 10 days during 10 days in June and July, 1996.

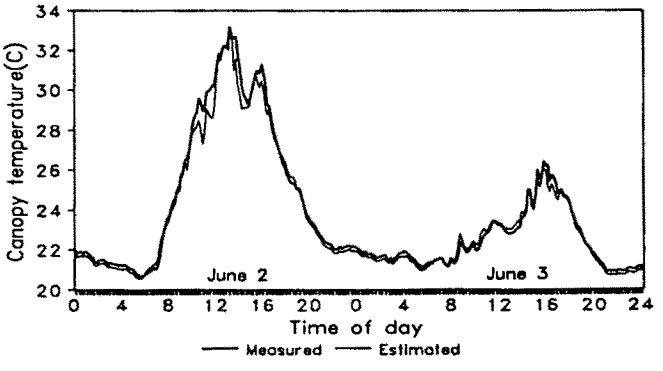


Fig. 9. Diurnal marches of canopy temperature measured and estimated by the present transpiration model in glasshouse on two consecutive days in 1996.

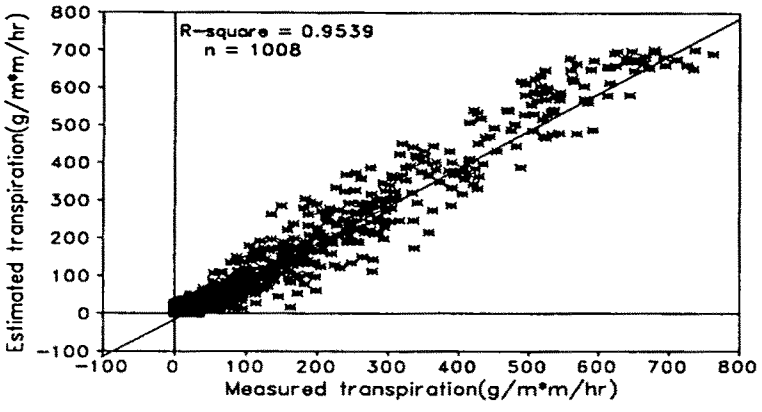


Fig. 10. Comparison of hourly transpiration fluxes measured and estimated by the present transpiration model at 10-minute intervals on seven days during June and July, 1996.

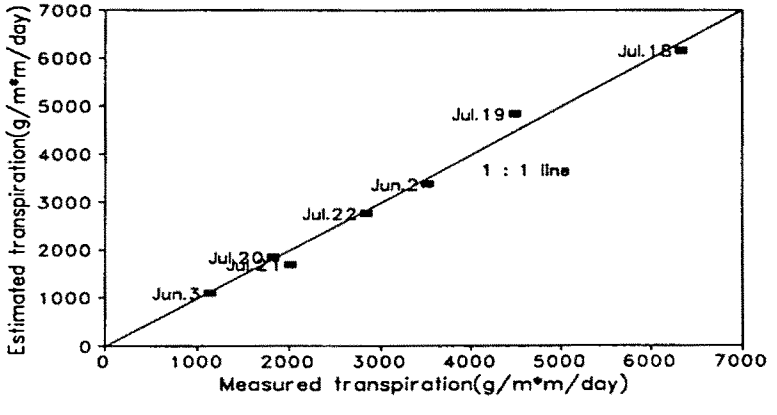


Fig. 11. Comparison of daily total transpiration measured and estimated by the present transpiration model for 7 days during June and July, 1996.

6. 결 론

본 연구에서는 온실 재배 토마토 군락의 열수지에 근거한 증산모델을 구성하고 실험을 통하여 모델에 필요한 계수의 추정과 모델의 검증을 수행하였다. 온실의 일사량과 엽-대기수증기압차(LVPD)를 매개변수로 하는 기공확산저항 추정식을 구성하여 기공확산저항 실측 자료를 이용하여 추정식의 계수를 추정하였다. 이 추정식으로 기공확산저항 변이의 80%이상을 설명할 수 있었으며 추정식에 이용하지 않았던 독립 자료를 이용하여 검정한 결과 정확도가 매우 높아 증산예측모

델의 구성식으로 이용될 수 있는 것으로 판단되었다. 반투과성 매질의 복사 흡수이론을 적용한 Stanghellini(1987)식과 Stanghellini and de Jong(1995)식[3]을 다소 변형하여 모델의 군락 순복사 추정식으로 사용하였으며 이 추정식에 의하여 계산된 순복사량은 실측치와 잘 일치하였다. 계수 추정에 사용하지 않았던 독립 자료를 이용하여 순복사 및 기공확산저항 추정식으로 구성된 증산 예측 모델의 군락 온도 및 증산 예측 정확도를 검증하였다. 모델에 의하여 계산된 군락 온도, 순간 증산속도 및 일 총 증산량은 실측치와 잘 일치하여 본 연구에서 작성된 증산 예측 모델은 온실 재배 토마토의 환경제어, 관개제어 등에 실용적으로 활용될 수 있을 것으로 판단되었다.

제3절 온실 재배 토마토의 경직경 변화에 의한 관계시기 결정

1. 연구 목적

작물의 관개 개시 시기를 결정하는 데에는 작물의 체내 수분 상태를 기준으로 하는 것이 토양수분 상태를 기준으로 하는 것보다 유리하다고 한다(Li an Huguet, 1990; Moriya et al, 1992). 작물의 체내 수분상태는 엽수분포텐셜, 잎의 상대수분함량 등을 측정하여 판단하는 것이 일반적이나 비파괴적으로 연속관측하는 것이 어렵기 때문에 관개시기의 판정에 이용하기 어렵다(Namken et al, 1969). 그런데 경직경의 변화는 작물의 수분상태를 잘 반영하며 또한 비파괴적으로 연속측정이 용이하여 이를 식물체 수분상태 변화의 간접적 지표로 삼아 관개시기의 결정에 이용하고자 하는 연구가 활발히 이루어지고 있다(Baille et al, 1992; Garnier and Berger, 1986; Katerji et al, 1990; Schoch et al, 1989; Simoneau et al, 1993). 경직경은 수분의 흡수와 증산의 불균형 및 생장에 의하여 수축과 팽창을 하는데(Kramer and Boyer,1995; Simoneau et al, 1993), 증산이 최대로 이루어지는 시각보다 다소 지체되어 최대로 수축이 되고 이후 증가하여 해뜨기 직전에 최대로 팽창이 되는 일변화를 보인다(Katerji et al, 1990; Klepper et al, 1971; Simoneau et al, 1993). 경직경의 최대 수축량은 일중 최소 엽수분포텐셜 및 토양수분함량과 높은 상관을 보인다는 보고(Garnier and Berger, 1986)와 상관성이 낮다는 상반된 보고(Katerji et al, 1994)도 있는데 이는 대상 작물의 차이에서 기인되는 것으로 보인다. 한편 해뜨기 직전에 나타나는 최대 경직경은 해뜨기 직전의 엽수분포텐셜 및 토양중 유효수분함량과 높은 정의 상관을 보인다고 한다(Katerji et al, 1994). 경직경 변화에 근거한 관개 시기의 판단 지표에 대해서는 Higgs와 Jones(1984) 및 Huguet(1985)는 경직경의 수축 정도를 이용하였으나 Katerji 등(1990, 1994)은 최대 경직경의 일 증가량을 기준으로 하는 것이 줄기의 수축 정도를 기준으로 하는 것 보다 유리하다고 하였다.

본 연구에서는 온실재배 토마토에서 경직경의 경시적 변화와 식물체내 수분, 토양수분, 증산, 일사량 등과의 관계를 정량적으로 파악하여 토마토 경직경의 monitoring에 의한 온실 재배 토마토의 관개시기 진단 방법을 모색하고자 하였다.

2. 재료 및 방법

Venlo형 유리온실에서 토마토 품종 서광을 1996년 1월 17일 파종하여 3월 19일에 플라스틱폿트[27cm(W) × 44.5cm (L) x 30cm(D)]에 정식하여 재식밀도가 8주/m^2가 되도록 배치하였다. 토양증발을 막기 위하여 투명 플라스틱 필름을 멀칭하였으며 물관리는 점적관수 시스템으로 토양수분장력이 0.2bar이하가 되도록 관개하였으며 본 실험기간 중인 7월 1일부터 7월 중순까지는 실험

목적에 따라서 한낮에 토마토에 위조 증상이 나타날 때까지 건조를 시켰다가 관개를 하였다. 온실의 환경 관리는 야간 기온을 15℃로 설정하였으며 주간에는 25℃이상이 되면 천창을 열어 환기가 되도록 하였다.

토마토 경직경의 변화는 strain gauge(120Ω) 4개로 full bridge를 구성하여 제작한 경직경변화 측정기(Moriya et al, 1992)를 군락 중앙부에 위치한 2개체에 대하여 측정하였는데 경직경 측정기는 제1화방이 붙어 있는 마디 바로 위 부분에 설치하였다. 증산은 경직경 측정 개체의 줄기 기부 근처에 sapflow meter(model SG10 & SG13, Dynamax Inc, Houston, TX)를 설치하여 측정하였으며, 토마토 체내 수분함량의 변화는 위와 동일 개체를 Load cell(suspension type)에 매달아(de Koning and Bakker, 1992) 지상부 생체중의 변화로 측정하였다. 한편 온실 내부의 일사량, 순복사량, 온도, 풍속, 토중 10cm와 20cm의 토양수분포텐샬, 엽온 등을 측정하였다. 모든 관측은 10초 간격으로 측정하여 10분간 평균을 집록하여 이용하였다.

3. 결 과

그림 1은 7월3일의 온실내 일사량, 증산량, 생체중 및 경직경의 일변화를 나타낸 것이다. 증산속도의 일변화는 일사량의 일변화 패턴과 매우 유사하였으며 식물체의 수분상태를 나타내는 생체중은 일출 직후인 오전 6시경에 최대에 달한 후 증산의 증가와 함께 감소하기 시작하여 증산이 최대에 달한 시각보다는 다소 지체되어 15시경에 최소를 나타냈으며 이후 증산의 감소와 함께 증가하였는데, 이는 일출 이후 수동적 수분 흡수의 증가가 증산 증가보다 지체되기 때문이다 (Kramer and Boyer, 1995). 오전 6시에 비하여 15시의 생체중은 약22g/개체 감소하였는데, 이 기간 중의 광합성에 의한 생체중 증가를 감안하면 실제 수분함량의 감소는 이 보다 큼을 알 수 있다. 한편 경직경의 일변화 패턴 역시 생체중의 변화와 매우 유사하여 6시경에 줄기가 최대로 팽창하였다가 이후 수축하기 시작하여 15시경에 최대로 수축이 되고 이후 다시 팽창하였다. 따라서 경직경 변화는 식물체의 수분 흡수와 증산의 불균형에 의한 식물체의 수분함량 변화를 잘 반영하는 것으로 판단이 되었는데, Simonneau 등(1993)도 복숭아나무에서 같은 결과를 보고하였다.

그림 2는 7월1일부터 16일까지 토양수분포텐셜, 생체중 및 경직경의 경시적 변화를 나타낸 것이다. 물관리는 실험 시작 날인 7월1에 충분히 관개를 하였으며 이 후는 외관상으로 위조 증상이 나타나기 시작한 다음날 오전 8시경에 관개를 하였다. 실험기간중 토양수분포텐셜의 경과를 보면 토중 10cm는 -0.2bar, 20cm는 -0.15bar로 유지되다가 7일부터 낮아지기 시작하여 각각 -0.7bar와 -0.3bar까지 저하하였으며 8일 오전에 관개 후 다시 각각 -0.2bar 및 -0.18bar까지 높아 졌고, 10cm의 경우는 11일부터 다시 낮아지기 시작하여 12일 오전 관개하가 직전에는 -0.35bar까지 낮아졌으며 20cm의 경우는 이 기간중 서서히 낮아져 -0.21bar를 나타내었다. 12일 이후는 10cm와 20cm 모두 약 -0.2bar를 유지하였다. 지상부 생체중과 경직경은 매우 유사한 변화 패턴을 보였는데, 앞에서 언급한 바와 같이 일출 경에 최저에 이르고 낮에는 감소하는 일변화를 보이는 동시에 또한 식물체가 생장함에 따라서 경시적으로 증가하는 변화를 보였다. 그러나 낮에 외관적으로 위조 증상이 나타났고 토양수분포텐샬이 현저하게 낮아졌던 7일과 11일의 경우 생체중 및 경직경

의 증가가 둔화되거나 정지하였고 관개 이후 다시 정상적인 증가 경향을 나타내었다. 10일의 경우는 외관적인 위조 증상이 나타나지 않았고 또한 토중 10cm와 20cm의 토양 수분 포텐셜이 -0.2bar 이상으로 유지되었음에도 불구하고 생체중 및 경직경의 증대가 이루어지지 않았다.

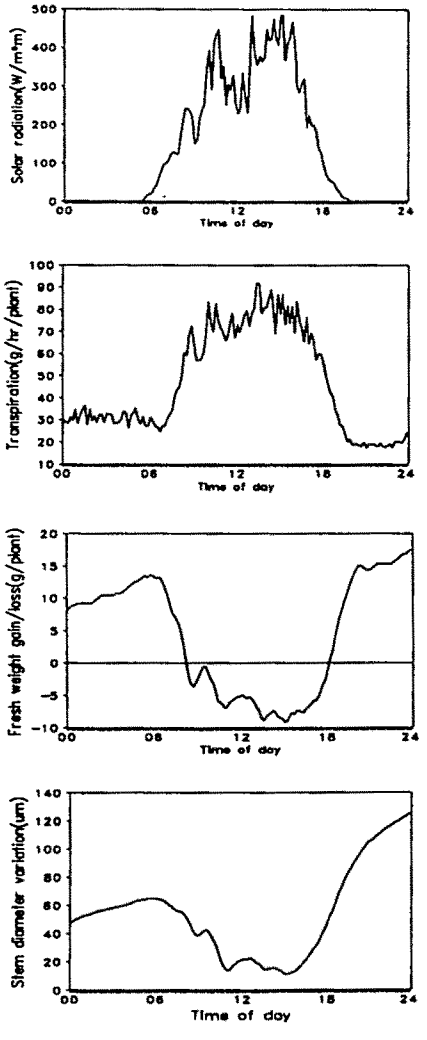


Fig. 1. Diurnal courses of solar radiation, transpiration, fresh weight and stem diameter of greenhouse tomato on 3 July, 1996.

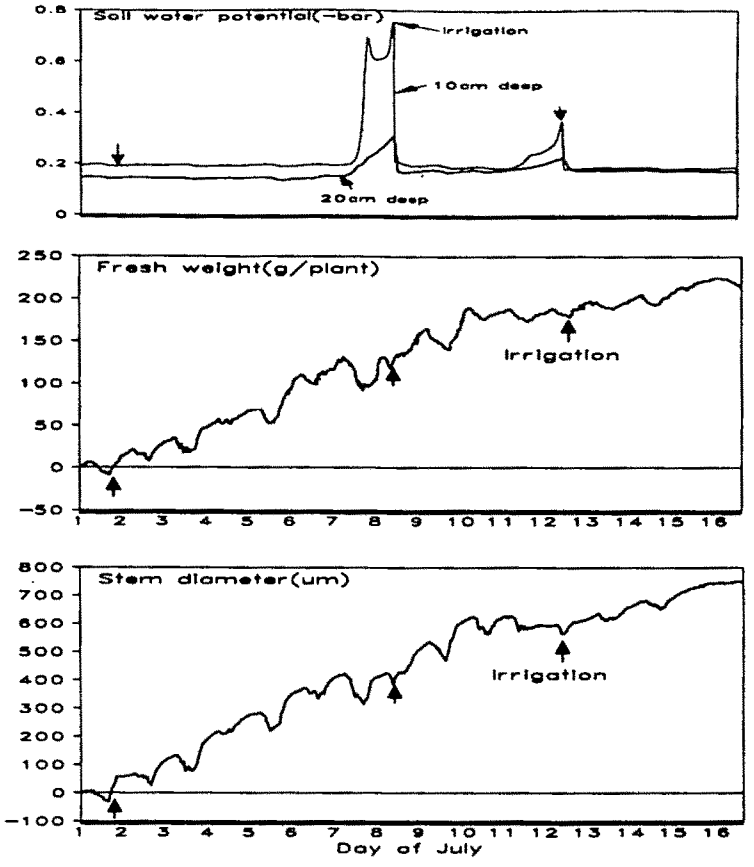


Fig. 2 Time course changes of soil water potential, shoot fresh weight and stem diameter of tomato plant during July 1 to 16, 1996. Stem diameter and shoot fresh weight were described on the basis of 0 at 100:00h on July 1.

실험기간 중 식물체의 수분장애 여부를 판단하기 위하여 증산량의 측정치와 증산모델에 의하여 예측한 증산량을 대비하여 나타낸 것이 그림 3이다. 이 증산 모델은 일사량, 군락 순복사량, 엽-대기수증기압차 등을 입력변수로 하여 토양수분이 부족하지 않은 상태에서 증산을 예측하는데, 그림에서 보는 바와 같이 7, 10, 11일을 제외하면 실측치와 예측치가 매우 잘 일치한다. 그러나 7, 10, 11일은 예측치 보다 실측치의 증산량이 현저하게 낮아 수분장애가 있었던 것으로 판단된다.

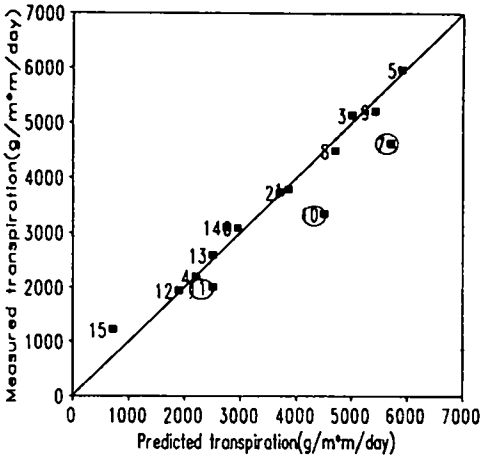


Fig. 3. Comparison of daily transpirations measured and predicted by the model(Lee, 1997). Numbers in the graph mean the dates of July.

식물체의 경직경 변화와 수분 상태 및 생장 변화의 관계를 정량적으로 파악하기 위하여 생체중과 경직경의 일변화(그림1) 및 경시적 변화(그림2) 특성을 그림 4와 같이 정량화 하였다. 즉 J일과 J+1일의 일출 무렵의 생체중 및 경직경의 차는 J일의 주간 광합성에 의한 생장뿐만 아니라 식물체 수분함량의 변화를 동시에 의미하는 것으로서 각각 일당 생체중 증가량(g/일, DG) 및 일당 경직경 증가량(μm/일, DI)으로 나타내었으며, 또한 일출 무렵의 최대 생체중 및 경직경과 낮 동안의 최저 생체중 및 경직경의 차이는 증산과 수분 흡수의 불균형에 의한 체내 수분 감소를 반영하는 것으로서(Garnier and Berger, 1986; Simoneau et al, 1993) 각각 최대 수분 손실량(g/일, ML) 및 최대 경직경 수축량(μm/일, MC)으로 나타내었다.

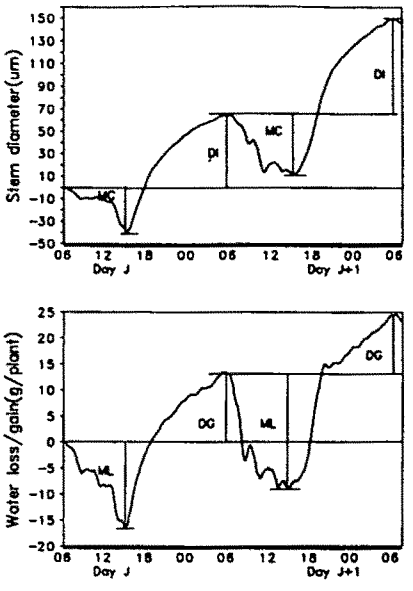


Fig. 4. Schematic representation of time course change in stem diameter and fresh weight of tomato: MC, maximum contraction of stem diameter; DI, daily increase in stem diameter; ML, maximum loss of water; DG, daily gain of fresh weight.

그림 5에서 보면 DG와 DI 및 ML과 MC는 고도로 유의한 정의 상관을 나타내어 경직경의 증가는 생체중의 증가를 잘 반영하며 또한 낮 동안의 경직경의 수축은 체내 수분함량의 감소를 잘 반영하여 경직경의 변화는 생장과 식물체의 수분변화에 대한 유용한 정보를 제공하는 것으로 판단되는데, Schoch 등(1989, 1990)[1]은 DI와 순광합성량 간에는 정의 상관이 있다고 하였으며 또한 Garnier 와 Berger(1986)는 MC와 일 중 최소 엽수분포텐셜 간에, Simonneau 등(1993)은 주간의 경 수축과 수분함량간에 밀접한 관련이 있다고 하여 본 연구와 유사한 결과를 보고한 바 있다

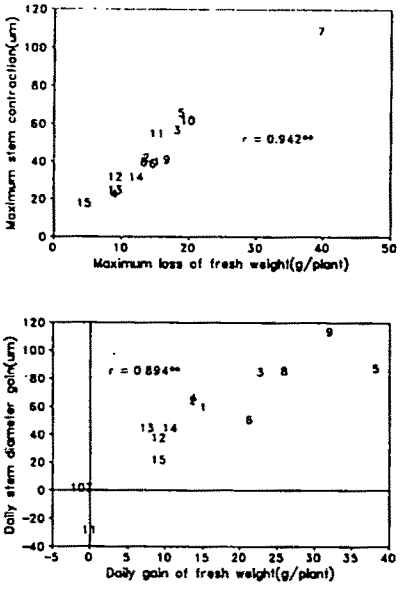


Fig. 5. Relationship between daily increases of shoot fresh weight(DG) and stem diameter(DI), and daily maximum contraction of stem(MC) and daily maximum loss of shoot fresh weight(ML).

그림 6은 일사량과 DG, DI, ML, MC와의 관계를 나타낸 것이다. 일사량과 DG 및 DI와는 고도로 유의한 정의 상관이 있어서 DG 및 DI는 일사량의 증대에 비례하여 증대되는 경향으로 물이 부족하지 않은 상태에서 DG와 DI는 광합성 증대에 따른 생장을 잘 반영하는 것으로 생각되며 Schoch 등(1987,1990)[1]도 적산일사량과 최대 경지경, 순광합성과 경직경 증가량(DI)은 정비례 관계가 있다고 하였다. 그러나 수분장애가 나타난 것으로 판단되는 7, 10, 11일의 경우는 각 날의 일사량에 의하여 기대되는 DG 및 DI보다 현저하게 낮았다. 따라서 DG 및 DI는 작물의 수분장애 여부를 판단하는데 유용하게 이용될 수 있을 것으로 보이며 또한 10일의 경우 외관상으로나 토양수분 상태로부터는 작물의 수분 부족을 파악 하기 어려웠으나 DG 및 DI에 의하여 분명하게 수분 부족이 파악될 수 있었던 것을 감안하면 토양수분에 근거하여 관개 시기를 결정하는 것 보다 DG 또는 DI로 판단하는 것이 유리한 것으로 판단된다. 한편 일사량과 ML 및 MC와는 유의한 정상관을 보여 일사량이 많아 증산이 증대될수록 낮동안의 식물체 수분함량 감소 및 줄기의 수축이 비례하여 증대되며 수분장애가 나타났던 날은 경향치 보다 높은 경향이었다. 그러나 수분장애가 미약하게 나타나기 시작한 10일의 경우는 확연하게 구분이 되지 않았는데 이는 식물체가 수분부족상태에 들어가면 기공을 닫아 증산이 줄어들기(Katerji et al, 1994) 때문인 것으로 생각된다. 따라서 경직경 및 생체중의 변화로 관개 시기를 판정하고자 하는 경우 ML 또는 MC를 기준으로 하는 것보다는 DG 또는

DI를 기준으로 판단하는 것이 유리할 것으로 판단되었는데 Katerji 등(1990, 1994)도 가지와 옥수수에 대한 실험에서 MC를 기준으로 하는 것보다 DI에 근거하여 관개 시기를 진단하는 것이 타당하다고 하였다.

이상의 결과들로부터 볼 때 DG 및 DI는 작물의 수분상태 변화를 잘 반영하며 이들의 연속적인 monitoring을 통하여 관개 시기의 정확한 진단이 가능하며 이를 토마토의 관개 자동화에 이용할 수 있을 것이다.

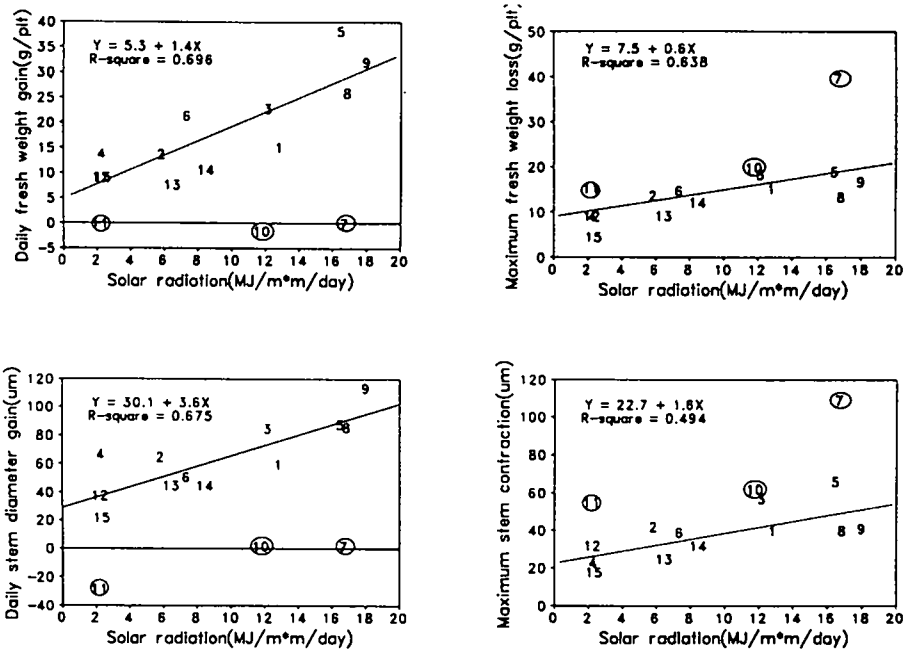


Fig. 6. Relationship between the daily solar radiation and the daily increases of shoot fresh weight(DG) and stem diameter(DI), and the maximum loss of shoot fresh weight(ML) and the maximum contraction of stem diameter(MC). The encircled dates were water-stressed and excluded in the calculation of regressions.

4. 결 론

본 연구의 목적은 온실재배 토마토에서 경직경의 경시적 변화와 식물체내 수분, 토양수분, 증산, 일사량 등과의 관계를 정량적으로 파악하여 토마토 경직경의 monitoring에 의한 관개시기 진단 방법을 모색하는 것이다. 스트레인 게이지로 제작한 경직경 미소변화 측정장치와 load cell을 이용하여 각각 경직경 변화 및 지상부 생체중의 변화를 1996년 7월 1이부터 16일까지 10분 간격으로 계측하였으며 또한 토양 수분함량, 증산, 일사량

동도 동시에 관측하였다. 실험을 시작하기 전에 충분한 관개를 하였으며 이후는 외관상으로 위조 증상이 나타나기 시작할 때 관계를 하였다. 생체중과 경직경은 매우 유사한 일변화를 보였는데 해뜰 무렵인 오전 6시경에 최대치에 이르고 이후 일사량과 증산의 증대에 따라서 감소하기 시작하여 증산이 최대에 이르는 시각보다 다소 늦은 오후 3시경에 최저에 달하였다가 일사 및 증산의 감소에 따라서 다시 증대하여 경직경의 변화와 체내 수분함량의 변화와는 매우 밀접한 관계가 있었다. 수분 부족 증상이 없었던 날들의 일당 경직경 증가량(DI) 및 생체중 증가량(DG)은 일사량과 높은 정의 상관을 보였으나 수분 부족장애를 받은 날들은 일사량에 따른 이들의 기대치보다 현저하게 낮아 생체중 및 경직경 증대가 매우 둔화되거나 오히려 감소하여 이들을 기준으로 관개 시기를 정확하게 판단하는 것이 가능하였다. 실험 기간 중 7월 10일의 경우 외관상으로는 위조증상이 없었으나 증산량은 현저히 감소하여 식물체가 수분부족 상태였던 것으로 판단되었는데 이날 토중 10cm 및 20cm에서의 토양수분 포테샬은 -0.2bar 이상으로 전일과 큰 차이가 없었으나 경직경 및 생체중의 증가는 현저히 둔화되어 수분 부족을 잘 반영하였다. 한편 낮 동안의 생체중 최대 감소량(ML)과 경직경 최대수축량(MC) 역시 일사량과 유의한 정의 상관을 보여 일사량 및 증산량 증대에 따라 체내 수분 손실량이 증대하고 이에 따라서 경직경이 감소하는 관계를 보였다. 그리고 식물의 수분장애가 심한 경우에는 일사량에 따른 이들의 기대치보다 현저하게 높았으나 수분장애가 미약한 경우에는 기대치와 구별하기가 어려워 이들을 기준으로 관개 시기를 정확하게 판단하기는 어려운 것으로 판단되었다. 따라서 관개 시기의 판단에는 토양수분, MC 및 ML을 기준으로 하는 것보다 DG 또는 DI를 기준으로 하는 것이 효율적인 것으로 판단되었다..

제 4절 최적관계시스템 제작 및 실증실험

1. 최적관개 제어 개념

작물의 관개 개시 시기를 결정하는 데에는 작물 체내 수분 상태를 기준으로 하고 관개량은 증발산량에 근거하여 결정하는 것이 가장 타당하다고 한다.(Moriya et al.,1992) 본 연구에서는 경직경 변화의 연속적 모니터링에 의하여 식물체내 수분상태를 진단하여 관개 시기를 결정하는 한편 증산 모델을 이용하여 관개량을 계산하여 관개를 하는 관개제어 시스템을 구축하였다. 최적 관개 제어 시스템의 개념은 다음과 같다.

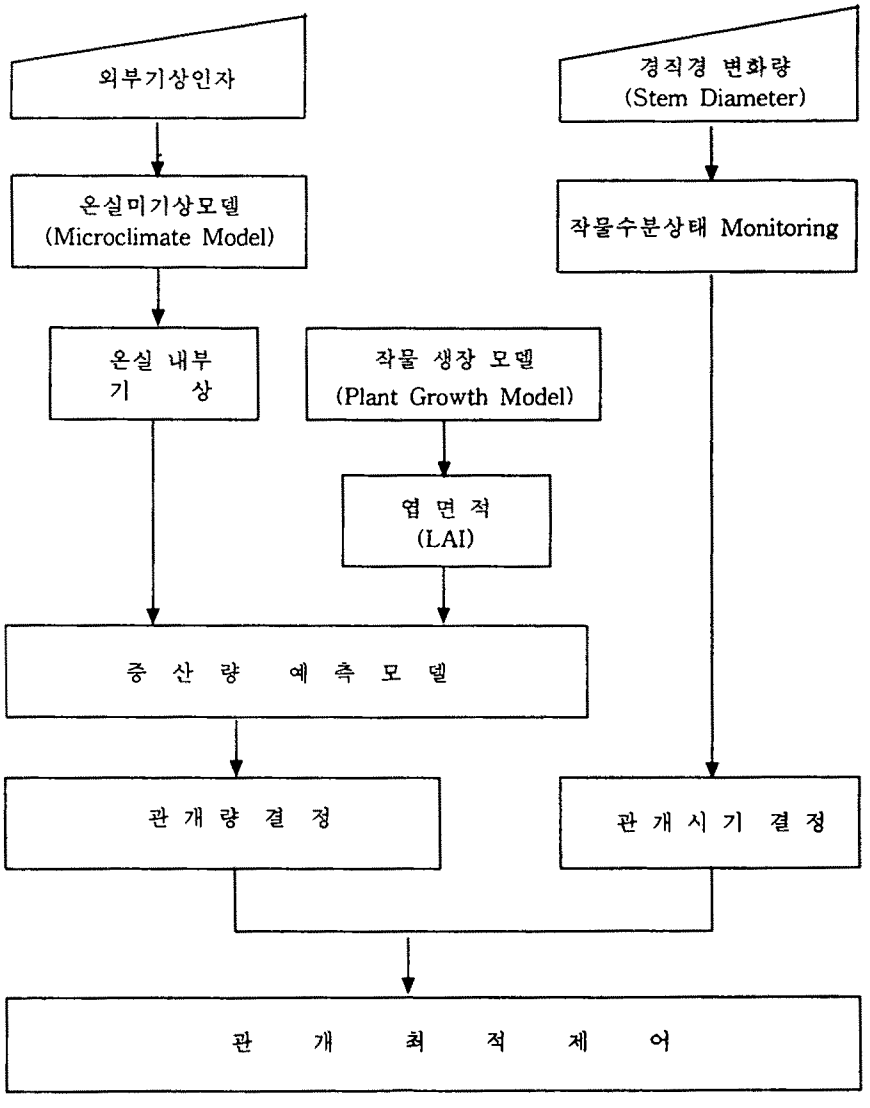


Fig. 1. Conceptual diagram of optimum irrigation control based on stem diameter monitoring and transpiration prediction.

2. 최적관개제어 시스템

가. 최적관개제어 시스템 구성

제어시스템은 센서로부터 자료를 수집, 저장하고 관개조건이 만족되었을 때 릴레이, SSR을 거쳐 전자밸브 및 펌프를 동작시키는 구조로 되어있다. 제어시스템은 10초간격으로 계측한 10분간의 평균값을 기록하고 PC와 연결하여 전송받을 수 있도록 그림 2와 같이 시스템을 구성하였다.

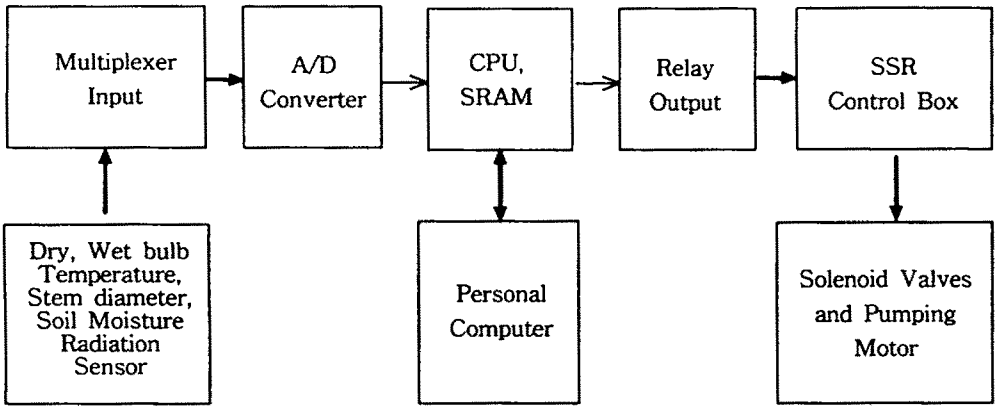


Fig. 2. Structure of optimal irrigation control system

나. 관개제어 시스템 제원

- CPU : Z80180 (16 bit)
- ROM : 32 kbytes
- RAM : 512 kbytes
- RTC :
- DISPLAY : LCD 4×20
- AD Converter : 12 bit
- 입력접점 : Analog 16 channel
- 출력접점 : 8 channel
- 통　　신 : RS232
- 입력장치 : 4×6 Key panel
- 외　　형 : FIBOX
- 프로그램 : 전용 C compiler로 작성하여 RAM에 Download

- 기　　능 : 계측, 저장, 비교제어, 통신기능을 수행하며 제어기에서 수집된 자료는 PC 프로그램에서 DownLoad 받아 File 로 저장할 수 있다
- 계 전 함 : 전자Valve 및 Motor를 On/Off 하기 위하여 SSR로 별도의 계전함을 구성하여 제어기에 연결하였다.

다. 경직경 측정기

식물체내 수분상태 변화에 따른 줄기 직경 변화를 측정하기 위하여 strain gauge를 이용한 변위(displacement) 변환기를 제작하였다. 변위변환기에 이용한 회로는 미소저항변화에 정확도가 높은 Wheatstone bridge회로를 구성하여 사용하였다. 그림 3에서 보는 바와 같이 Strain guage(120Ω) 4개로 Full bridge를 구성하고 입력 전압 V_i가 DC 5V일 때 브릿지 출력전압 V_o를 제작한 Controller의 ADC 채널에 입력하였다. 제작한 경직경 측정기의 測度檢定 결과는 표2 및 그림 4와 같다.

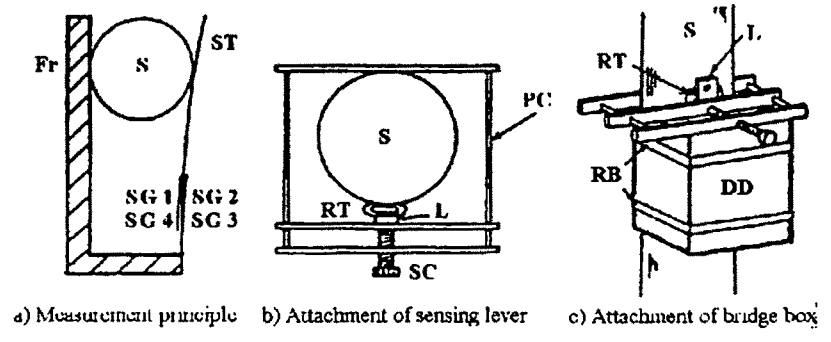


Fig. 3. Device for measuring stem diameter microvariation(Moriya et al, 1992)
S: stem; Fr: rigid frame; ST: displacement sensing lever; SG: strain gauge
PC: pinch cork; RB: rubber tube; DD: bridge box; L: sensing lever

Table 2. Equations to regress the output voltage(Y, μV) to lever displacement(X, μm) of stem diameter gauges.

	Equation	R^2
gauge 1:	Y = 1.988X	0.9987
gauge 2:	Y = 2.146X	0.9986
gauge 3:	Y = 1.970X	0.9988
gauge 4:	Y = 1.996X	0.9984
gauge 5:	Y = 2.082X	0.9983

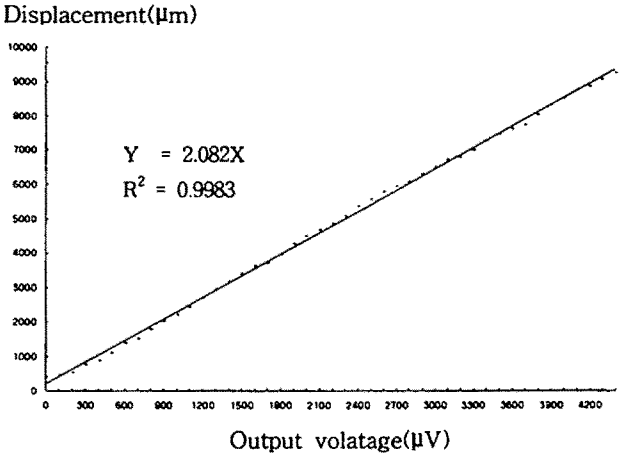


Fig. 4 Relationship between output voltage and displacement of stem diameter gauge lever

라. 관개 제어

제어시스템은 그림 5에서와 같이 센서입력, 자료저장, 제어조건비교, 관개제어의 과정을 반복수행하면서 제어를 수행한다 토양수분장력, 일사량, 경직경 증가량의 값을 사용여 앞절의 연구 결과에 따라서 관개시기를 결정하며 관개량은 직전의 관개시기로부터 현재 관개시까지 증산에 의한 토양수분소모를 증산량 예측모델로 계산하여 결정한다.

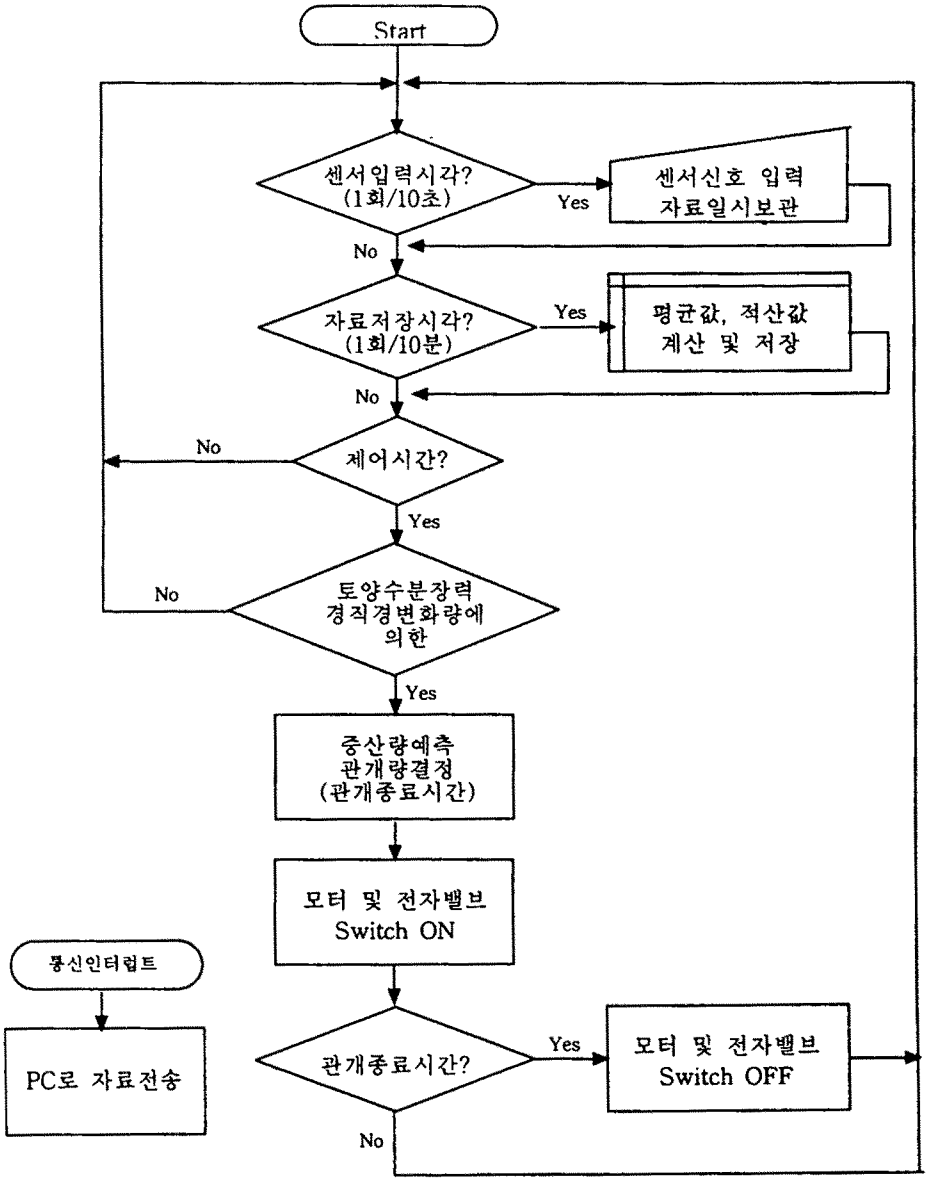


Fig. 5. Flow diagram of optimal irrigation control system

* 관개제어 시스템의 전체 프로그램은 부록에 수록하였음.

3. 관개시스템 적합도 검증 예비실험

가. 목 적

관개제어 시스템의 관개제어상의 오류 및 증산량 모델의 정상동작 여부등 시스템의 적합도를 검증하기 위하여 관개제어 예비실험을 실시하였다.

나. 재료 및 방법

토마토 품종 서광을 공시하여 1997년에 Venlo형 유리온실에서 실험을 실시하였다. 4월19일에 플라스틱폿트[27cm(W)×44.5cm(L)×30cm(D)]에 정식하여 재식밀도가 8주/m^2가 되도록 배치하였다. 시비는 기비로 N-P-K = 4.9-3.8-3.8 g/pot 시용하였으며 질소 및 가리는 관개 시스템을 통하여 15일 마다 N-K = 1 - 1 g/pot를 관비하였다. 환경관리는 주간 25℃, 야간 10℃로 설정하였으며 투명 플라스틱 필름으로 멀칭하여 토양 증발을 억제하였다.

관개 시점 결정 방법만을 달리하여 다음과 같이 세 처리를 두었으며, 관개는 관개제어 시스템에 연결된 점적관개장치를 이용하여 세 처리 모두 오전 6시부터 1시간 30분간(2,700cm^3/pot) 관개하였다.

처리 번호	처 리 내 용(관개시기 결정)
처리 1	토양(지하 10cm)수분 장력이 0.2bar 이상이 되면 관개
처리 2	적산일사량(X,MJ/m^2/day)과 경직경변화량(Y, μm/day)과의 관계식 Y=30.1+3.6X에 의하여 예측한 경직경 증가량에 비하여 실측한 경직경 증가량이 90%수준 이하일 때 관개
처리 3	일 경직경 증가량이 0 μm 미만일 때 관개

* 경직경센서는 제1화방 마디의 상단에 고정하였다가 생장이 어느정도 진전된 후 제2화방 상단으로 이동하여 고정하였다.

온실 내부의 기온, 습도, 일사량기상을 측정하였으며, 각 처리별로 군락 중앙부의 2개체에 대하여 경직경변화, 토양수분장력을 측정하였으며 7월 9일부터 8월1일까지 성숙된 과실수 및 과실중을 조사하였다.

나. 결 과

표3에서 보는 바와 같이 관개시기 결정방법에 따라 수량은 유의하게 차이가 있었으나 과실수는 유의한 차이가 없었다. 전일의 경직경 증가가 0 μm이하인 때 관개를 한 경우가 토양 수분 장력 0.2bar를 기준으로하여 관개한 경우 보다 수량이 높았으며 실측 경직경 증가량이 일사량을 근거로하여 예측한 예상 경직경 증가량의 90%이하인 때 관개를 한 경우는 과습되는 경향이 있어 수량이 가장 낮았다.

Table 3. Fresh fruit yield of greenhouse tomato irrigated with different methods of irrigation timing

Treatment	Yield(kg/10a)	Fruit number(/10a)
1	3,371	17,480
2	3,178	16,560
3	3,887	16,100
F-value	12.37 *	< 1
LSD(5%)	412	

* significant at probability level of 5%

라. 결 론

토양 수분을 근거로 하여 관개 시기를 결정하는 것 보다 경직경 변화를 근거로 하여 관개시기를 결정하는 것이 수량 향상에 유리함을 확인할 수 있었다. 그러나 관개시의 경직경 증가 임계점에 대한 검토가 더 필요한 것으로 판단되었다. 관개시스템 시제품의 관개제어 오류를 교정하고 증산 모델을 검증, 보완하여 관개시기 뿐만 아니라 관개량을 자동 결정하여 관개할 수 완전한 시스템을 구축하였다.

4. 관개시스템 적합도 검증

가. 목 적

토마토 생장 및 증산 예측모델을 추가하여 완성한 관개제어시스템의 적합도를 검증 보완하여 경직경 변화 monitoring 및 증산량 예측에 근거한 관개제어시스템의 실용화 가능성을 평가하고자 하였다,

나. 재료 및 방법

토마토 품종 서광을 공시하여 1997년에 Venlo형 유리온실에서 실험을 실시하였다. 8월8일 파종.육묘하여 9월6일에 플라스틱폿트[27cm(W)×44.5cm(L)×30cm(D)]에 정식하여

재식밀도가 8주/m^2가 되도록 배치하였다. 시비는 기비로 N-P-K = 4.9-3.8-3.8 g/pot 시용하였으며 질소 및 가리는 관개 시스템을 통하여 15일 마다 N-K = 1 - 1 g/pot를 관비하였다. 환경관리는 주간 25℃, 야간 10℃로 설정하였으며 투명 플라스틱 필름으로 멀칭하여 토양 증발을 억제하였다.

관개 시점 결정 및 관개량 결정 방법을 달리하여 다음과 같이 세 처리를 두었으며, 관개제어 시스템에 연결된 점적관개장치를 이용하여 세 처리 모두 오전 6시부터 관개하였다.

처리 번호	처 리 내 용(관개시기 결정)
처리 1	토양(지중 10cm)수분 장력이 0.2bar 이상이 되면 직전 시점으로부터 현 관개 시점까지의 예측 증산량 많금을 관개
처리 2	적산일사량(X, MJ/m^2/day)과 경직경변화량(Y, μm/day)과의 관계식 Y=30.1+3.6X에 의하여 예측한 경직경 증가량에 비하여 실측한 경직경 증가량이 90%수준 이하일 때 직전의 시점으로부터 현 관개 시점까지의 예측 증산량 많금을 관개
처리 3	일 경직경 증가량이 0 μm 미만일 때 직전의 시점으로부터 현 관개 시점까지의 예측 증산량 많금을 관개

* 관개 처리는 15엽기 부터 시작하였으며 최종에는 제1화방 마디의 상단에 경직경 측정 센서를 고정하였으며 2주일 마다 센서 고정 위치를 1화방씩 위로 이동하였슴.

온실 내부의 기온, 습도, 일사량기상을 측정하였으며, 각 처리별로 군락 중앙부의 2개체에 대하여 경직경변화, 토양수분장력을 측정하였으며 또한 토마토의 생장량, 엽면적, 과실중, 등을 조사하였다.

다. 결 과

관개시기 결정방법에 따른 10월 23일부터 12월 1일까지 40일간의 관개 상황 및 관개량을 나타낸 것이 표 4이다. 지표로부터 10cm의 토양수분 장력이 0.2bar 이상으로 높아지는 경우 관개를 한 처리(T1)는 이 기간 중 14회 관개가 되었으며, 일 경직경 증가량이 적산일사량(X, MJ/m^2/day)과 경직경변화량(Y, μm/day)과의 관계식 Y=30.1+3.6X에 의하여 예측한 경직경 증가량에 비하여 90%수준 이하일 때 관개를 하는 처리 2(T2)의 경우 25회 관개가 되었고, 경직경이 증가되지 않을 겨우 관개를 하는 처리 3(T3)의 경우는 15회 관개가 되었다. 한편 이 기간 중의 총 관개량은 처리에 관계없이 일정하였는데 이는 관개량을 직전의 관개 이후부터 현재의 관개 시기까지의 적산 증산량으로 하였기 때문이다.

Table 4. Time and amount(g/pot) of irrigation under different irrigation management practices

Date	Irrigation treatment*			Date	Irrigation treatment		
	T1	T2	T3		T1	T2	T3
Oct. 23	1,000**	1,000	1,000	Nov. 12	1,041	1,041	1,041
24	0	0	0	13	0	0	0
25	0	0	0	14	0	0	0
26	0	0	0	15	0	0	0
27	320	320	0	16	1,225	1,225	1,225
28	0	0	0	17	0	310	0
29	700	700	1,020	18	0	368	682
30	0	0	0	19	1,044	369	369
31	1,345	1,345	1,345	20	0	348	0
Nov. 01	0	0	0	21	746	397	0
02	0	0	0	22	0	147	0
03	1,300	1,300	1,300	23	0	371	0
04	0	0	0	24	0	316	838
05	830	830	830	25	0	342	0
06	320	320	320	26	0	111	0
07	0	0	0	27	1,525	238	675
08	0	779	779	28	0	348	0
09	1,165	386	386	29	0	218	0
10	0	0	0	30	0	196	768
11	0	0	0	Dec. 01	1,152	384	0

* T1: Water was applied when soil water potential dropped below -0.2bar
  T2: Water was applied when daily increment of maximum stem diameter around dawn was smaller than the diameter increment(Y, μm/day) estimated by the equation, Y=30.1+3.6X, where X=daily total solar radiation(MJ/m^2/day)
  T3: Water was applied when daily increment of maximum stem diameter around dawn was smaller than 0μm.

** Irrigation amounts were determined by the transpiration amounts from the previous irrigation time to present irrigation time, which were estimated by the transpiration model.

한편 각 처리의 지표로부터 10cm의 토양수분 장력의 경시적 변화를 나타낸 것이 그림 6이다. 그림에서 보는 바와 같이 관개 횟수가 비슷하였던 처리 1과 처리 3의 경우는 토양 수분 장력이 매우 유사하게 진행되었으나 처리 2의 경우는 관개량이 비슷하였음에도 불구하고 타 처리에 비하여 토양 수분 장력이 높게 유지되었다. 처리 3의 경우 관개가 자주 되어 1회의 관개량이 작기 때문인 것으로 판단되었다. 그림 7은 각 처리별 경직경의 경시적 변화를 나타낸 것이다. 처리 2와 3에 비하여 처리 1이 경직경의 증가가 뚜렷하였는데 이는 처리에 의한 차이라기보다는 측정 개체간의 변이에 기인하는 것이라고 판단된다. 그 예로 처리 1과 처리 3은 관개의 횟수와 1회의 관개량이 유사하였고 또한 토양 수분 장력의 변화가 유사하였음에도 불구하고 경직경의 증가 양상이 매우 상이 하였다.

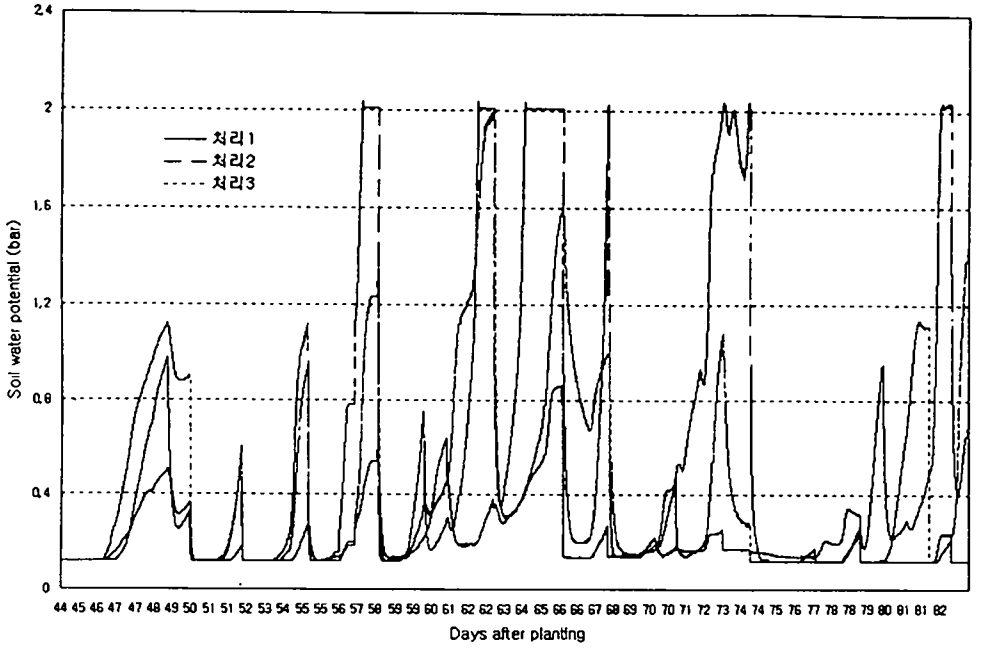


Fig. 6. Temporal changes of soil water potentials when exposed to different irrigation managements. Treatments were explained in Table 3.

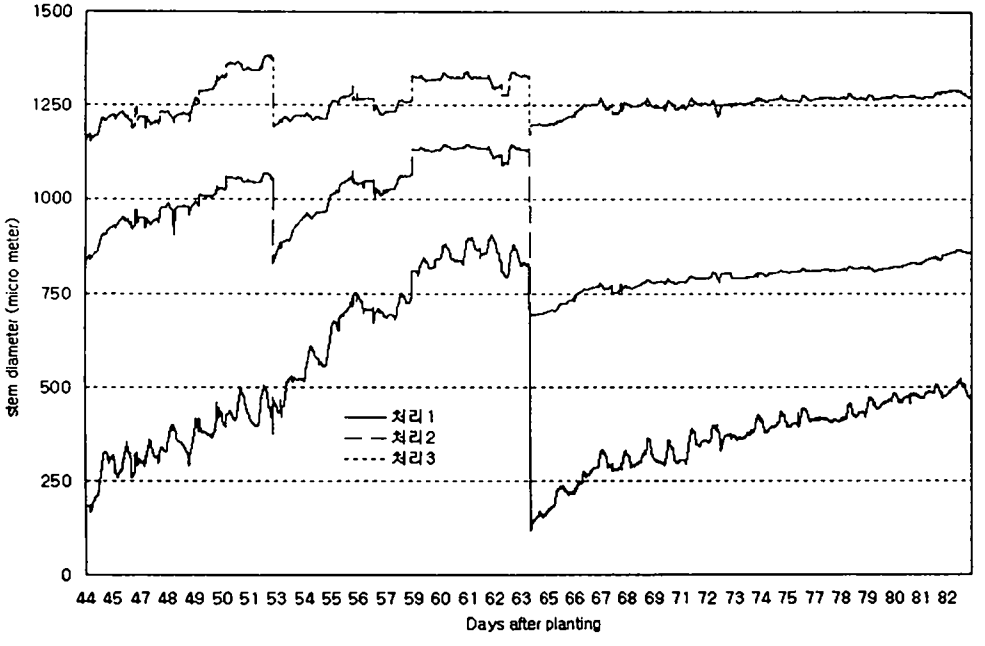


Fig. 7. Temporal changes of stem diameters when exposed to different irrigation managements. Treatments were explained in Table 3.

표 4는 12월 5일에 각 처리당 6개체를 채취하여 각 기관별 건물중을 조사한 결과 이다. 지상부 전체 건물중은 처리 1과 처리 3은 비슷하였으나 처리3의 경우는 처리 1과 3에 비하여 낮았다.

Table 4. Growth of tomato under different irrigation management practices

Treatment	Dry weight(g/plant)					LAI
	Leaf	Petiole	Stem	Fruit	Total	
T1	28.0	14.4	21.4	9.2	73.0	6.7
T2	26.0	13.4	19.6	6.1	65.1	7.7
T3	29.2	15.4	22.5	7.0	74.1	8.4

* Treatments were explained in Table 3.

라. 결 론

본 연구에서 개발한 관개 시스템을 이용하여 이식 후 44일째부터 40일간 관개 실험을 한 결과 시스템이 정상적으로 작동이 되는 것을 확인할 수 있었다. 관개시기를 결정함에 있어서 토양수분 장력이 0.2bar이상으로 될 때에 관개를 하는 경우(처리 1)와 경직경의 증가가 이루어지지 않을 경우(처리 3)에 관개를 하는 경우는 관개의 횟수와 생장량은 차이가 없었다. 그러나 경직경의 증가가 일사량으로부터 예측한 경직경 증가량의 90%이하가 될 때 관개를 하는 경우(처리 2) 앞의 두 경우보다 관개의 횟수가 증가하여 1회당 관개량이 작았고 토양수분 장력도 처리1과 처리 2보다 높았으며 지상부의 생장은 작았다. 경직경의 증가 정도는 개체 및 생육 단계에 따라서 다르고 또한 측정위치에 따라서도 다르기 때문에 일사량으로 경직경의 증가 정도를 예측하여 이를 기준으로 관개시기를 결정하는 데에는 어려움이 있는 것으로 판단되었다. 그러나 경직경의 증가 여부만을 기준으로 하는 경우는 충분히 실용적 가치가 있는 것으로 판단되었다. 그러나 이와 같은 결론은 수량과 품질이 평가되지 않은 상태에서의 잠정적인 결론이며 최종적인 결론은 현재 진행되고 있는 실험이 완료된 이후에 재평가될 것이다.

참 고 문 헌

제 1장

1. 김인수, 강진구. 1993. 시설원예의 경영규모와 경제성. 국내시설산업 발전을 위한 심포지엄. p35-61, 서울대학교 농업개발연구소

제2장

1. Aubinet M, J. Deltour and Dehalleux. 1989. Stonatal regulation in greenhouse crops : Analysis and simulation. Agric. and For. Meteor. 48 ; 21-44
2. Ben-Asher J, C. W. Meek. R. B. Hutmacher, and C. J. Phene. Computational approach to assess actual transpiration from aerodynamic and canopy resistance. Agron. J. 81 ; 776-782
3. Bertin, N and C. Gary. 1993. Tomato fruit-set : A case study for validation of the model TOMGRO.. Acta Hort. 328 ; 185-193
4. Bertin N and E. Heuvelink. 1993. Dry-matter production in a tomato crop : Comparison of Two simulation models. J. of Hort. Sci. 68(6) ; 995-1011
5. Challa, H. and E. Heuvenlink, 1993. Economic evaluation of crop photosynthesis. Acta Hort. 328 ; 219-228
6. Ehler N and P. Karlsen. 1993. Optico-a model based real-tome expert system for dynamic optimization of CO_2 enrichment of greenhouse vegetable crops. J. of Hort. Sci. 68 ; 485-494
7. Ehler Niels. 1991. An autocalibrating model for simulating and measuring net canopy photosynthesis using a standard greenhouse climate computer. Conputers and Elcctronics in Agriculture, 6 ; 1-20
8. Giaglaras Panagiotis, Maria Baille, Alain Baille. 1995. Net photosynthesis response to light and air CO2 concentration of Begonia × hiemalis : Whole plant measurements and modelling. Scientia Horticulturae, 63 ; 83-100
9. Gijzen, H. 1992. Simulation of photosynthesis and dry matter production of greenhouse crops. CABO-DLO/TPE Simulation Report CABO-TT, 28, CABO-DLO Wageningen, The Netherland

10. Gijzen H and J. Goudriaan. 1989. A flexible and explanatory model of light distribution and photosynthesis in row crops. Agricultural and Forest Meteorology, 48;1-20
11. Goudrian, J.. 1977. Crop micrometeorology: a simulation study: 249pp, Centre for Agricultural Publishing and Documentation, Wageningen, The Netherland.
12. Goudrian, J., H. H. Van Laar, H. Van Keulen and W. Louwerse, 1985. Photosynthesis, CO_2 and plant production. In: Wheat growth and modelling(Day, D. and R.K, Atkin Eds). NATO ASI Series, Series A: Life Science, 86 ; 107-122
13. Grange R. I and J. Andrews. 1994. Expansion rate of young tomato fruit growing on plants at positive water potential. Plant, Cell and Environment, 17 ; 181-187
14. Hand D. W, G. Clark, M. A. Hannah, J. H. Thornley and J. Warren Wilson. 1992. Measuring the canopy net photosynthesis of Glasshouse crops. Journal of Experimental Botany, 43 ; 375-381
15. Heuvelink E. 1995. Dry matter production in an tomato crop : Meastrememts and simulation. Annals of Botany 75 ; 369-379
16. Heuvelink E. 1995. Growth, development and yield of a tonmato crop periodic destructive measurements in a greenhouse. Scientia Horticulturae, 61 ; 77-99
17. Jay M. H., J. L. Heilman and R. J. Lascano. Determination of soil water evaporation and transpiration from enertgy bgalance and sten flow measurements. Agricultural and Forest Meteorology, 52 ; 287-301
18. Jolliet O. 1994. Hortitrans, a model for predicting and optimizing humidity and transpiration in greenhouses. J. agric. Engng Res., 57 ; 23-37
19. Jolliet O and B. J. Bailey. 1992. The effect of climate on tomato transpiration in greenhouses : Measurements and models conparison. Agricultural and Forest Meteorology, 58 ; 43-62
20. Jones J. W, E. Dayan, L. H. Allen, H. Y. Van Keulen, H. Challa. 1991. A dynamic tomato growth and yield model(TOMGRO). American Society of Agriculturla Engineers, 34(2) ; 663-672
21. Kano A. and Cornelius H.M. van Bavel. 1988. Design and test of a simulation model of tomato growth and yield in a greenhouse. J. Japan. Soc. Hort. Sci. 56(4) ; 408-416
22. Katerji N and A. Perrier. 1985. Determination de la resistance globale d'un couvert vegetal a la diffusion de vapeur d'eau et de ses differentes conposantes

Approche the'orique et ve'rification experinentale sur une culture de luzerne. Agricultural and Forest Meteorology, 34 ; 105-120

23. Kitano M and H. Eguchi. 1991. Dynamics of plant water relations as affected by evaporative demand. IFAC Mathematical and Control Applications in Agriculture and Horticulture, 367-372
24. Logendra S. and H. W. Janes. 1992. Light duration effects on carbon partitioning and translocation in tomato. Scientia Horticulturae, 52 ; 19-25
25. Myster Jardar, Roar Moe. 1995 Effect of diurnal temperature alternations on plant morphology in some greenhouse crops - a mini review. Scientia Horticulturae, 62 ; 205-215
26. Papadakis G, A. Frangoudakis, and S. Kyritsis. 1994. Experinental investigation and modelling of heat and mass transfer between a tomato crop and the greenhouse environment. J. Agric. Engng Res., 57;217-227
27. Rijsdijk A. A and G. Houter. 1993. Validation of a model for energy consumption, CO_2 consumption and crop production(ECP-model). Acta Horticulturae, 328 ; 125-131
28. Romero-Aranda, R and J. J. Longuenesse. 1995. Modelling the effect ef air vapour pressure deficit on leaf photosynthesis of greenhouse tomatoes : The importance of leaf conductance to CO_2. Journal of Horticultural Science. 70(3) ; 423-432
29. Seginer Ido. 1994. Transpirational cooling of a greenhouse crop with partial ground cover. Agricultural and Forest Meteorology, 71 : 265-281
30. Sinoquet H et R. Bonhmme. 1989. Modélisation de linterception des rayonnements solaires dans une culture en ranfs. II. Structure géométrique du couvert et validation du modèle. Agronomie, 9 ; 619-628
31. Spitters C. J. T, H. A. J. M. Toussaint and J. Goudriaqan, 1986. Separating the diffuse and direct component of global radiation and its implications for modeling canopy photosynthesis. Agricultural and Forest Meteorology, 38 ; 217-229
32. Stanghellini C, W. Th. M. van Meurs. 1992. Environmental control of greenhouse crop transpiration. J. agric. Engng Res., 51 ; 297-311
33. Tchamitchian M and J. J. Longuenesse. 1992. A photosynthesis model for greenhouse row crops. Acta Horticulturae, 328 ; 137-149
34. Thornley J. H. M, D. W. Hand and J. Warren wilson, 1992. Modelling light absorption and canopy net photosynthesis of glasshouse row crops and application

to cucumber. Journal of Experimental Botany, 43 ; 383-391

35. Yang X., Ted H. Short, robert D. Fox and William L. Barerle. 1990. Dynamic modeling of the microclimate of a greenhouse cucumber row-crop. Part I. Theoretical model. American Society of Agricultural Engineers, 33(5) ; 1701-1716

제 3장

1. Avissar R. qnd Y. Mahrer, 1982. Verification study of a Numerical Greenhouse Microclimate Model. Transactions of the ASAE; 1711-1982
2. Brutsaert W., 1975. On derivable formula for long-wave radiation from clear skies. Water Rsour. Res. 11; 742-744
3. Chandra Pitam, L. D. Albright, N. R. Scott. 1981. A time dependent analysis of greenhouse thermal environment. American Society of Agricultural Engineers, 81; 442-448.
4. Chung, S. O. and R. Horton. 1987. Soil heat and moisture flow with a partial surface mulch. Water Resour. Res. 23; 2175-2186
5. Froehlich. D. P, L. D. Albright, N. R. Scott, p. Chandra. 1979. Steady-Periodic analysis of glasshouse thermal environment. American Society of Agricultural Engineers, 79 ; 387-399.
6. Jaxob. C. 1982. Modelisation de la vitesse du vent et de la temperature de l'air a l'interieurd'une parcelle bocagere. Agricultural Meteorology, 27: 89-104
7. Kimball, B. A. 1973. Simulation of the energy balance of a greenhouse. Agric. Meteor., 11(1973) ; 243-260
8. Kindelan, M. 1980. Dynamic simulation of Greenhouse Environment. Transactions of the ASAE; 1232-1239
9. Kittas. C, J. P. Chiapale, O. De villele et F. Aries. 1987. Modele d'estimation de la temperature de paroi d'une serre. Agricultural and Forest Meteorology, 39 ; 131-142.
10. Kittas, C.. 1987. Un modele d'estimation des deperditions energetiques diurnes des serres. Agronomie, 7(3) ; 175-181.
11. Levit. H. J and R. Gaspar.1988. Energy budget for greenhouses in humid-temperate climate. Agricultural and Forest Meteorology, 42 ; 241-254.
12. Levit, H. J and R. Gaspar. 1988. Energy budget for greenhouses in

humid-temperate climate. Agricultural and Forest Meteorology, 42 ; 241-254.

13. Mahrer, Y., R. Avissar, O. Naot and J. Katan. 1987. Intensified soil solarization with closed greenhouses: Numerical and experimental study. Agric. and For. Meteor., 41(1997); 325-334
14. McInnes, K.J., 1981. Thermal conductivities of soils from dry-land wheat regions of Easter Washington. M.S. Thesis, Washington State Univ., Pullman
15. Papadakis, G., M. Mermier, J. F. Meneses and T. Boulard, 1996. Measurement and Analysis of Air exchange rates in a greenhouse with continuous Roof and side openings. Agric. Engng Res. 63; 219-228
16. Philip. J. K. 1957. Evaporation, and moisture and heat fields in the soil. Journal of moteorology, 354-366.
17. Seginer Ido, Louis D. Albright. 1980. Rational Operation of Greenhouse Thermal-Curtains. American Society of Agricultural Engineers, 80 ; 1240-1245.
18. Suzuki Haruo and Hideo Tanada. 1988. Deviations in the Horizontal Distribution of Soil Temperatures beneath Film Mulch and evaluation of the Mulch Effect. J. of Agric. and Meteor., 44(2) ; 119-126.
19. Takakura.T, A. Jordan and L. L. Boyd. 1971. Dynamic Simulation of plant Growth and Environment in the Greenhouse. Trans. of ASAE; 964-971
20. Geometry of long-wave radiation at the ground. (I) angular distribution of incoming radiation. Quart. J. Royal Meteor. Soc. 101; 13-24

제4장

1. Aikman, D. P., 1996. A procedure for optimizing carbon dioxide enrichment of a glasshouse tomato crop. J. Agric. Engng. Res. 63: 171-184
2. Challa, H. 1993. Optimal diurnal climate control in greenhouses as related to greenhouse management and crop requirements. p.119 - 137 in: Y. Hashimoto, G.P.A. Bot, W. Day, H.-J. Tantau, H. Naomi(Eds.). The computerized greenhouse. Academic Press, Inc.
3. Critten, D. L., 1991. Optimization of CO_2 concentration in greenhouses: a modelling analysis for the lettuce crop. J. Agric. Engng. Res. 48: 261-271
4. Hashimoto, Y., 1989. Recent strategies of optimal growth regulations by speaking plant concept. Acta Hort. 260: 115-122

5. van Heneten, E. J. and J. Bontsema, 1991. Optimal control of greenhouse climate. p.27 - 32 in Proc. of IFAC/ISHS workshop, Matsuyama, Japan
6. Marsh, L. S. and L. D. Albright, 1991 Economically optimum day temperatures for greenhouse hydroponic lettuce production. Part I: A computer model. Trans. ASAE 34(2): 550-556
7. Marsh, L. S. and L. D. Albright, 1991 Economically optimum day temperatures for greenhouse hydroponic lettuce production. Part II: Results and simulations. Trans. ASAE 34(2): 557-562
8. Seginer, I., A. Angel, S. Gal, and D. Kantz, 1986. Optimal CO_2 enrichment strategy for greenhouses. A simulation study: J. Agric. Eng. Res. 34: 285-304
9. Seginer, I., 1989. Optimal greenhouse production under economic constraints. Agric. systems 29: 67-80
10. Seginer, I. and A. Sher, 1993. Optimal greenhouse temperature trajectories for a multi-state-variable tomato model. p.153 - 172 in: Y. Hashimoto, G.P.A. Bot, W. Day, H.-J. Tantau, H. Naomi(Eds.). The computerized greenhouse. Academic Press, Inc.
11. Tantau, H.-J. 1993. Optimal control for plant production in greenhouses. p.139 - 152 in: Y. Hashimoto, G.P.A. Bot, W. Day, H.-J. Tantau, H. Naomi(Eds.). The computerized greenhouse. Academic Press, Inc.

제5장

1. Aikman, D. P. and Houter, G., 1990. Influence of radiation and humidity on transpiration: Implications for calcium levels in tomato leaves. J. Hortic. Sci. 65(3): 245-253.
2. Ansley R. J, W. A. Dugas, M. L. Heuet and B. A. Trevino. 1994. Stem flow and porometer measurements of transpiration from honey mesquite (*Prosopis glandulosa).* Journal of Experimental Botany, 45 ; 847-856
3. Baille M, J. C. Lauty and A. Baille. 1992. Some comparative results on evapotranspiration of greenhouse ornamental crops, using lysimeter, greenhouse H_2O balance and LVDT sensors. Acta Horticulturae, 304 ; 199-208
4. Bakker, J. C., 1990. Effects of day and night humidity on yield and fruit quality of glasshouse tomatoes. J. Hortic. Sci. 65(3): 323-331.
5. Boulard, T., A. Baille, M. Memoire and F. Vignette, 1991. Measures et

modelisation de la resistance stomatique foliaire et de la transpiration d'un couvert de tomates de serre. Agronomie(1991) 11: 259-274.

6. De Graaf, R. and J. van den Ende, 1981. Transpiration and evapotranspiration of the glasshouse crops. Acta Hortic. 119: 147-158.
7. De Koning A. N. M and J. C. Bakker. 1992. In situ plant weight measurement of tomato with an electronic force gauge. Acta Horticulturae, 304 ; 183-186
8. FAO, 1977. Crop water requirements. FAO irrigation and Drainage Paper 24: 15-44.
9. Garnier E and Berger. 1986. Effect of water stress on stem diameter changes of peach trees growing in the field. Journal of Applied Ecology. 23 ; 193-209
10. Genslet W and F. Diaz-Munoz. 1983. Simultaneous stem diameter expansions and apoplastic electropotential variations following irrigation or rainfall in cotton. Crop Science, 23 ; 921-923
11. Genslet W and F. Diaz-Munoz. 1983. Stem diameter variations in cotton under field conditions. Crop Science, 23 ; 907-912
12. Grime, V.L., J. I. L. Morison and L. P. Simmonds, 1995. Including the heat storage term in sapflow measurements with the stem heat balance method. Agric. and For. Meteor. 74(1995): 1-25.
13. Grun R, J. P. Toutnier. 1992. Micrometric measurenent of stem diameter changes as a means to detect nutritional stress of tomato plants. Acta Horticulturae, 304 ; 265-272
14. Higgs, K. H. and H. G. Jones. 1984. A microcomputer-based system for continuous measurement and recording fruit diameter in relation to environmental factors. J. Exp. Bot. 35:1646-1655
15. Huguet, J. G. 1985. Apprèciation de l'eta hydrique d'une plante a partir des variations micrometrique de la dimension des fruits ou des tiges en cours de Journees. Agronomie 5; 733-741
16. Hck M. G. and Betty Klepper. 1977. Water relations of cotton. II. Continuous estimates of plant water potential from stem diameter measurements. Agronomy Journal, 69 ; 593-597
17. Jehoshua, R., E. Rendon-Poblete, M. Allen Stevens, and Abdel-Ilah Ambri. 1981. Use of leaf water potential to determine water stress in field-grown tomato plants. J. Amer. Soc. Hort. Sci, 106(6) ; 732-736

18. Jolliet, O. and B. J. Bailey, 1992. The effect of climate on tomato transpiration in greenhouse: measurements and models comparison. Agric. and For. Meteor. 58(1992):43-62.

19. Jolliet, O., 1994. HORTITRANS, a model for predicting and optimizing humidity and transpiration in greenhouses. J. Agric. Engng Res.(1994) 57: 23-37.

20. Katerji. N., P. G. Schoch, P. Rimgoto and J. C. L'Hotel. 1990. Diagnostic des périodes de contrainte hydrique chez des plantes d'aubergine cultivées en serre, au moyen des microvariations des tiges. Agronomie 10; 541-549

21. Katerji, N., T. Francois, B. Olivie and Q. Philippe. 1994. Behavior of maize stem diameter during drying cycles: Comparison of two methods for detecting water stress. Crop Sci. 34 ; 165-169

22. Klepper, B., V. D. Browning and H. M. Taylor. 1971. Stem diameter in relation to plant water status. Plant Physiol. 48; 683-685

23. Kramer, P. J. and J. S. Boyer. 1995. Water relations of plants and soils. Academic Press: 495pp

24. Lassoie J. P. 1973. Diurnal dimensional fluctuations in a douglas-fir stem in response to tree water status. Forest. Science, 19(4) ; 251-255

25. Lee, B. W., 1996. Transpiration measurement and models comparison in greenhouse tomato. 1996세계한민족과학기술자종합학술대회 논문집(농학.식품과학그룹): 2067-2073.

26. 이 변 우. 1997. 온실재배 토마토의 증산모델 개발 및 검증. 생물생산시설환경 6(3): 투고중

27. Li, S. H. and J. G. Huguet. 1990. Controlling water status of plant and scheduling irrigation by the micromorphometric method for fruit trees. Acta Hort. 278: 332-342

28. Monteith, J. L. and M. H. Unsworth, 1990. Principles of environmental physics, 2nd ed. Edward Arnold: 291pp.

29. Moriya H, K. Iwao, H. Kageyama. 1992. Studies on non-destructive and continuous measuremint of water contents and applications to irrigation in crop culture. Acta Horticulturae, 304 ; 345-351

30. Namken L. N, J. F. Bartholic, and J. R. Runkles. 1969. Monitoring cotton plant stem radius as an indication of water stress. Agronomy Journal, 61 ; 891-893

31. Okuya, A. and T. Okuya, 1988. The transpiration of greenhouse tomato plants in

rockwool culture and its relationship to climatic factors. Acta Hortic. 230: 307-31.

32. Papadakis, G., A. Frangoudakis and S. Kyritsis, 1994. Experimental investigation and modelling of heat and mass transfer between a tomato crop and greenhouse environment. J. Agric. Engng Res.(1994) 57: 217-227.
33. Parkhurst, D. F., P. R. Duncan, D. M. Gates and F. Kreith, 1968. Convection heat transfer from broad leaves of plants. J. Heat Transfer(Trans. ASME) 90: 71-76.
34. Pieters, J., 1995. Influence of condensation on the heat balance and light transmission of a greenhouse. Ph. D. Dissertation, Ghent University: 260pp.
35. Schoch P. G, J. C. L'Hôtel, P. Dauplé, G. Conus et M.J. Fabre. 1989. Micròvariations de diamètre de tige pour le pilotage de l'irrigation. Agrononie, 9 ; 137-142
36. Schoch P. G, J. C. L'Hôtel et B. Brunel. 1990. Croissance du diametre de la tige de tomate : Errets du rayonnement et de la temperature nocturne. Agricultural and Forest Meteòrology, 50 ; 229-238
37. Schoch P. G, N. Katerje, P. Rimgoto, M. Tchamitchian, P. Malet. J. c. Yhotel and M. C. Daunay. 1987. Influece du niveau d'alimentation hydrique sur les variations du diametre dis eiges, du poeneiel hydrique, de la resistance stomatique, de la transpiraeion et de la photoswynthese de l'aubergine(*Solanum melangena* L.). Agricultural and Forest Meteorology, 40 ; 89-104
38. Simonneau T, R. Habib, J.-P. Goutouly and J.-G. Huguet. 1993. Diurnal changes in stem diameter depend upon variations in water content : Direct evidence in peach trees. Journal of Experimental Botany, 44(260) ; 615-621
39. Stanghellini, C., 1987. Transpiration of greenhouse crops, an aid to climate management. Ph.D. Dissertation, Wageningen Agricultural University: 150pp.
40. Stanghellini, C. and T. de Jong, 1995. A model of humidity and its applications in a greenhouse.. Agric. and For. Meteor. 76(1995): 129-140.
41. Stoffers, J. A. 1975. Radiation absorption of canopy rows. Acta Hortic. 46: 91-95.
42. Urban L, A. Jaffrin and A. Chraibi. 1993. Analysis of pressure-volumi curves of leaves of *Rosa hybrida* cv. sonia. Journal of Experimental Botany, 44 ; 605-613
43. Van Leperen, W. and H. Madery. 1994. A new method to measure plant water uptake and transpiration simultaneously. Journal of Experimental Botany, 45 ; 51-60
44. Werkhoven C. 1992. Sensors for irrigation scheduling of cultures in the field. Acta Horticulturae, 304 ; 259-264

부 록

APPENDIX 1. LIST OF TOMATO GROWTH MODEL "GREENTOM"

1. Definition of the abbreviations used in the model

ABNF	number of daily-aborted fruits on each fruit-bearing truss(1/day-truss)
ABOR	total number of aborted fruits on each fruit-bearing truss(1/day-truss)
ABORMX	maximum ratio of fruit abortion
AGF	age classes of fruits belonging to each truss
AGLS	age classes of leaves belonging to each truss
ASCSP	daily growth rate of shoot dry matter(gDW/day-plant)
ASRF	assimilate requirement of leave(gCH_2O/g DM)
ASRL	assimilate requirement of leave(gCH_2O/g DM)
ASRS	assimilate requirement of stem(gCH_2O/g DM)
BOX	age classes for POL and POF function
CD	compensation point of CO_2(ppmv)
CO2L	carbon dioxide concentration(ppmv)
CPOOL	carbon pool
DELT	simulation interval
DMGF	dryweight of fruit(gDW/plant)
DMGL	dry weight of growing leaves belonging to each truss(gDW/truss-plant)
DMMF	dry weight of mature fruits(gDW/plant)
DTFAST	fraction of one day, DTFAST = 1/NFAST
DWF	dry weight of each fruit on each age class(on each truss)(gDW/fruit)
DWL	leaf dry weight on each age class(on each truss)(gDW/leaves on a truss)
DWS	stem dry weight on each age class(on each truss)(gDW/stemss on a truss)
DWTR	dry weight of fruits on each truss(gDW/fruits on a truss)
FABOR	ratio of fruit abortion
FPN	number of fruit initiated per new node depending on PLSTN
FPNPT	potential number of fruit initiated per new node
FRLG	number of nodes initiated between the first truss appearance and fruit set
FRPT	ratio of petiole weight to leaf blade weight
FRST	ratio of stem segment to leaf growth rate
FTRUSN	node number initiating the first truss
GENR	rate of node initiation(nodes/day) (=TEMFAC*GENRAT)
GENRAT	maximum rate of node initiation(nodes/day)
GENTEM	temperature function of vegetative development
GP	daily gross photosysnthesis(gCH_2O/plant/day)
GPF	instantaneous leaf gross photosynthesis(gCH_2O/plant/day)
GREF	growth efficiency(g biomass/ g CH_2O)
GRESP	daily growth respiration(gCH_2O/plant/day)
IHE	hour in a day
JUL	Julian day
NBFPT	potential number of fruit on a truss depending on truss number
NBLV	number of leaves on a given day
NBRU	number of truss/plant on a given day
NBRUP	number of fruit-bearing trusses
NDAYS	number of days for simulation

NFAST	number of fast loop. if time-step = 1 hour, NFAST = 24
NSF	number of set fruits on each trusses
NSTART	starting Julian day of simulation
PAR	photosynthetically active radiation(W/m^2)
PGF	potential growth rate(sink demand) of fruits(gDW/plant/day) belonging to each truss
PGL	potential growth rate(sink demand) of leaves(gDW/plant/day) belonging to each truss
PGRED	function to modify PMAX under suboptimal daytime temperature
PGS	potential growth rate(sink demand) of stem segments(gDW/plant/day) belonging to each truss
PLAR	leaf area per plant(m^2/plant)
PLAR2	leaf area(m^2/plant) considering the leaf senescence
PLARI	initial leaf area per plant(m^2/plant)
PLE	potential expansion rate of leaves belonging to each truss(m^2/day/leaf)
PLM2	planting density(plants/m^2)
PLSTN	total number of nodes/plant on a given day(plastochron index)
PLSTNI	initial number of nodes/plant(initial plastochron index)
PMAX	maximum leaf photosynthetic rate(umolCO_2/m^2-leaf/s)
PNGP	potential growth rate(sink strength) of total shoot(gDW/plant/day)
POF	potential fruit growth rate function in response to age class
POL	potential leaf expansion rate function in response to age class
PPFD	photosynthetic photon flux density(umol/m^2/s)
PROOT	proportion of root biomass
PTNFRT	potential growth rate(sink strength) of total fruits on a plant(gDW/plant/day)
PTNLVS	potential growth rate(sink strength) of total leaves on a plant(gDW/plant/day)
PTNSTM	potential growth rate(sink strength) of total stem segments on a plant (gDW/plant/day)
Q10	temperature coefficient for maintenance respiration
QE	leaf quantum use efficiency(umol CO_2 fixed/umol photon absorbed)
RCDRW	daily total production of biomass(gDM/plant/day)
RCNF	newly-initiated fruits on each truss(1/day/truss)
RDVFR	daily fruit development rate(1/day)
RDVFRF	instantaneous leaf development rate (1/day)
RDVFRT	fruit development rate table depending on air temperature
RDVLV	daily leaf development rate(1/day)
RDVLVF	instantaneous leaf development rate(1/day)
RDVLVT	leaf development rate table depending on air temperature
RMAINT	daily total maintenance respiration(gCH_2O/plant/day)
RMAINTF	instantaneous maintenance respiration(gCH_2O/plant/day)
RMRF	maintenance respiration coefficient for fruit(gCH_2O/g-DM/day)
RMRL	maintenance respiration coefficient for leaf and stem(gCH_2O/g-DM/day)
RMRR	root maintanence respiration rate(gCH_2O/g-DM/day)
RMRS	stem menintanence respiration rate(gCH_2O/g-DM/day)
SLAMN	minimum specific leaf area(m^2/g)
SLAMX	maximum specific leaf area(m^2/g)

SOLRAD	outside solar radiation(W/m^2)
SOSIR	source/sink ratio
TABF	total number of aborted fruits per plant
TABNF	number of newly-aborted fruits per plant per day
TDMF	dry weight of fruits(gDM/plant)
TDML	dry weight of leaves(gDM/plant)
TDML2	dry weight of leaves(gDM/plant) considering defoliated leaves
TDMS	dry weight of stems(gDM/plant)
TEMFAC	temperature effects on nodes, leaf and fruit development rate on daily basis
TEMFCF	instantaneous temperature effects on node, leaf and fruit development rate
TFAST	time step of simulation in green house
TMPA	air temperature(℃) in greenhouse
TMPG	temperature(℃) for temperature function PGRED
TNF	total number of fruit per plant
TNMF	total number of mature fruit per plant
TNSF	total number of set fruit per plant
TPL	ratio of new fruiting trusses to new leaves
TRCNF	total number of fruits newly-initiated
TRESP	daily total respiration($gCH_2O/m^2/day$)
TRGH	light transmission coefficient of green house
VPD	vapor pressure deficit(kPa)
WLVSI	initial DM of leaves(gDM/plant)
WSTMI	initial DM of stem(gDM/plant)
XBOX	age class(AGLS*100 or AGF*100)
XFR	temperature for temperature function table of fruit development(C)
XGEN	node number for GENRAT function(node number/plant)
XK	measured light extinction coefficient of tomato canopy
XLA	area of leaves belonging to each truss(m^2/leaves on a truss)
XLV	temperature for temperature function table of leaf development(C)
XM	leaf light transmission rate
XNBLV	total number of leaves/plant
XNBRU	total number of trusses/plant
XNFT	number of fruit on each truss
XROOT	PLSTN for PROOT function table
XTEM	temperature for GENTEM function table

2. Listing of the tomato growth model GREENTOM

```
*************************************************************************
TOMATO GROWTH MODEL TO SIMULATE THE GROWTH, DEVELOPMENT
AND YIELD
*************************************************************************
     DIM POL(12),POF(12),BOX(12),NBFPT(30),GENTEM(6),XTEM(6),GENRAT(8)
     DIM XGEN(8),RDVLVT(9),XLV(9),rdvfrt(9),xfr(9),XFPN(10),PROOT(6)
     DIM XROOT(6),PGRED(8),TMPG(8),DWL(30),DWS(30),AGLS(30),DWF(30,7)
     DIM AGF(30,7),DWTR(30),PGS(30),PGL(30),PGF(30,15),ABNF(30),RCNF(30)
     DIM XNFT(30),XNSFT(30),ABOR(30),D(30,7),ple(30),xla(30),nsf(30)
     DIM XVAL(30),XARG(30),WAZI(6),WSLO(6),WIN(6),AC(6),XGAUSS(3),WGAUSS(3)

********* DEFINE FUNCTION FOR CALCULATION OF SATURATION VAPOR
          PRESSURE
    DEF FN ES(X)=610.78D0*EXP(17.269D0*X/(X+237.30D0))
    DEF FN ESP(X)=610.78D0*EXP(17.269D0*X/(X+237 30D0))*
          (4097.9337D0/((X+237.30D0)^2 D0))
*************************************************************************
    DEF FNAMIN1(AYY,BYY)
     IF AYY<BYY THEN FNAMIN1=AYY ELSE FNAMIN1=BYY
    END DEF
*************************************************************************
    DEF FNMAX(AXX,BXX)
     IF AXX>BXX THEN FNMAX=AXX ELSE FNMAX=BXX
    END DEF
*************************************************************************
    DEF FNTABEX(XVAL,XARG,DUMMY,K)
     FOR J=2 TO K
     IF DUMMY > XARG(J) THEN GOTO 7100
     GOTO 7200
7100 NEXT J
     J=K
7200 FNTABEX=(DUMMY-XARG(J-1))*(XVAL(J)-XVAL(J-1))/(XARG(J)-XARG(J-1))+XVAL(J-1)
    END DEF
*************************************************************************************
CO2O=350:'PPMCO2 IN OUTSIDE AIR
   CO2L=340:'CO2 IN GREENHOUSE
   cp=1004.D0:      'in J/(kg.K), specific heat of air at constant pressure
   SIG = 5.67D-8:' / 100000000.: 'STEFAN-BOLTZMAN CONST.(W/M2/K)
   gamma=66 2D0:'64.6*(1+0.000946*xxp)
   rhoa=1.204D0:'1.204*293.2/(273.2+xxp)
   nhu=15.5D-6:'15.1*((xxp+273.2)/293.2)^1.75/1000000' 'kinematic viscosity(m2/s)
   kapa=22.2D-6::'21.5*((xxp+273.2)/293.2)^1 75/1000000:'thermal diffusivity(m2/s)
   aex=1 D0/273 16D0''1/(xxp+273.16): 'thermal expansion of air
   lambda=(2500.25D0-2.365D0*20.D0)*1000.D0: 'J/Kg, latent heat of vaporization

*************************************************************************************'*********
****** data for GAUSSIAN integration
   DATA 0.112702,0.500000,0.887298
   READ XGAUSS(1),XGAUSS(2),XGAUSS(3)
   DATA 0.277778,0.444444,0.277778
   READ WGAUSS(1),WGAUSS(2),WGAUSS(3)
*********************************************************************
   open"ı",#11,"c:\tomgro\greentom\inw3_95W.dat"
   OPEN"O",#15, "c.\tomgro\simtom\gro3_95W.dat"
   open"o",#16,"c:\tomgro\simtom\dev3_95W.dat
 simulation beginning
 parameters are read in CROPPARM.TOM and MGT.TOM
     GOSUB INPUTT
*************************************************************************
 initial conditions
     GOSUB INIT
 daily loop
 **********
     IDD=0
     FOR JDAY = 287 to 343:'1 TO NDAYS
     NFAST=24 D0*6.D0
     DTFAST = 1 0/NFAST
     DATE = jday
 initialize variables that accumulate during the day
     GP = 0.
     TEMFAC = 0.
     RMAINT = 0.
     RDVLV = 0.
```

```
      RDVFR = 0.
      DTRAN = 0.
 ************************************************************************
 fast loop (if time-step = 1 hour, NFAST = 24)
 ********************************************
      FOR JF=1 TO NFAST
      IDD=IDD+1
 TFAST is the hour of the day (h)
      TFAST = (JF-1)*24.D0/NFAST
 input climatic data
      input#11,date,ttime,tmpa,inhum,solrad,co2l
      vpai=fn es(tmpa)*inhum/100.d0
********** read   co2 concentration  ***************************
 calculation of astronomical parameter and greenhouse transmission of radiation
      GOSUB SOLAR
 calculation of development rate
      GOSUB DEVFAST
 calculation transpiration rate
      GOSUB TRANS
 calculation of gross photosynthesis
      GOSUB PHOTO
 calculation of maintenance respiration
      GOSUB RESP
 integration of variables on 24 hours
      TEMFAC = TEMFAC + TEMFCF*DTFAST
      RMAINT = RMAINT + RMAINTF*DTFAST
      RDVLV = RDVLV + RDVLVF*DTFAST
      rdvfr = rdvfr + rdvfrf*dtfast
      GP = GP + GPF*DTFAST
      DTRAN=DTRAN+TRAN*DTFAST
100   NEXT JF
 end of fast loop
 ************************************************************************
 development and sink strength
      GOSUB DEVRATE
 calculation of conversion efficiency(GREF) of assimilates to dry matter
      ASRL=1.39:ASRS=1.45:ASRF=1.39
      GREF=PNGP/(ASRL*PTNLVS+ASRS*PTNSTM+ASRF*PTNFRT)
 daily production of biomass
      RCDRW = GREF*(GP+CPOOL-RMAINT)
      IF RCDRW<0 THEN
                        RCDRW = 0.
                        RMAINT = GP+CPOOL
      END IF
      FOR TT=1 TO 6:XVAL(TT)=PROOT(TT):XARG(TT)=XROOT(TT):NEXT TT
      RCDRW = RCDRW * (1. - FNTABEX(PROOT,XROOT,PLSTN,6))
 calculation of source/sink ratio
      SOSIR = FNAMIN1(1. , RCDRW / (PNGP+EPS))
 daily growth respiration
      GRESP = (GP+CPOOL-RMAINT) * (1 - GREF*1.125)
 daily total respiration
      TRESP = RMAINT + GRESP
 fruit setting
      GOSUB LOSRATE
 dry matter partitioning
      GOSUB DMRATE
 outputs
      GOSUB OUTPUT
       NEXT JDAY
** end of daily loop**
************************************************************************
      CLOSE (9)
      CLOSE (10)
      CLOSE (11)
      CLOSE (15)
      STOP
      END

*****************************************************************************************
SUBROUTINE: READ CROP PARAMETER INPUT
*****************************************************************************************
      INPUTT:
      OPEN"I",#10,"c:\tomgro\greentom\CROPPARM.DAT"
      OPEN"I",#9,"c:\tomgro\greentom\grnparm.DAT"
READ THE PARAMETERS AND THA TABLES
      INPUT#10,xk,xm,ck,vpdl,XXX$
      INPUT#10,TPL,EPS,GREF,fpnpt,xxx$
```

```
INPUT#10,ABORMX,Q10,RMRL,RMRF,FTRUSN,XXX$
INPUT#10,slamx,slamn,FRLG,FRPT,FRST,xxx$
FOR IX=1 TO 12
 INPUT#10,BOX(IX)
NEXT IX
 INPUT#10,XXX$
FOR IX=1 TO 12
 INPUT#10,POL(IX)
NEXT IX
 INPUT#10,XXX$
FOR IX=1 TO 12
 INPUT#10,POF(IX)
NEXT IX
  INPUT#10,XXX$
FOR IX=1 TO 30
 INPUT#10,NBFPT(IX)
NEXT IX
  INPUT#10,XXX$
FOR IX=1 TO 8
 INPUT#10,PGRED(IX)
NEXT IX
  INPUT#10,XXX$
FOR IX=1 TO 8
 INPUT#10,TMPG(IX)
NEXT IX
 INPUT#10,XXX$
FOR IX=1 TO 6
 INPUT#10,GENTEM(IX)
NEXT IX
 INPUT#10,XXX$
FOR IX=1 TO 6
 INPUT#10,XTEM(IX)
NEXT IX
  INPUT#10,XXX$
FOR IX=1 TO 8
 INPUT#10,GENRAT(IX)
NEXT IX
  INPUT#10,XXX$
FOR IX=1 TO 8
 INPUT#10,XGEN(IX)
NEXT IX
  INPUT#10,XXX$
FOR IX=1 TO 9
 INPUT#10,RDVLVT(IX)
NEXT IX
  INPUT#10,XXX$
FOR IX=1 TO 9
 INPUT#10,XLV(IX)
NEXT IX
 INPUT#10,XXX$
FOR IX=1 TO 9
 INPUT#10,RDVFRT(IX)
NEXT IX
 INPUT#10,XXX$
FOR IX=1 TO 9
 INPUT#10,XFR(IX)
NEXT IX
 INPUT#10,XXX$
FOR IX=1 TO 6
 INPUT#10,PROOT(IX)
NEXT IX
 INPUT#10,XXX$
FOR IX=1 TO 6
 INPUT#10,XROOT(IX)
NEXT IX
  INPUT#10,XXX$
INPUT#9,NSTART,NDAYS,DELT,NFAST,INTOUT,TRGH,XXX$
INPUT#9,PLM2,PLSTNI,WLVSI,PLARI,WSTMI,XXX$
FOR IX=1 TO 6
INPUT#9,WAZI(IX)
NEXT IX
INPUT#9,XXX$
FOR IX=1 TO 6
INPUT#9,WSLO(IX)
NEXT IX
INPUT#9,XXX$
FOR IX=1 TO 6
```

```
      INPUT#9,AC(IX)
      NEXT IX
      INPUT#9,XXX$
      INPUT#9,FAFN,AG,NSL,EWL,WZL,GZL,XXX$
      INPUT#9,FI,KC,FFL,TRD,LAT,LON,XXX$
      CLOSE#10
      CLOSE#9
      RETURN

'*****************************************************************************************
' SOUBROUTINE INITIALIZE
'*****************************************************************************************
 INIT:
' number of "aborted" fruits, set fruits and total fruit number
      TABF = 0.
      TNSF = 0.
      TNF = 0.
' all variables at planting stage (1st truss flowering)
      DWL(1) = WLVSI
      DWS(1) = WSTMI
      DWTR(1) = 0.
      PLAR = PLARI
      XLA(1) = PLAR
      ABOR(1) = 0.
' the organ age at starting date is very important for the
' final weight of first organs
      AGLS(1) = 0 3:'0.005:'0.15
' here, up to 30 trusses may be simulated
     FOR I=2 TO 30
      DWL(I)=0.
      DWS(I)=0
      AGLS(I)=0.
      ABOR(I)=0.
      XNFT(I)=0.
      DWTR(I)=0.
     NEXT I
      FOR I=1 TO 30
        FOR J=1 TO 7
         DWF(I,J)=0.
         AGF(I,J)=0.
        NEXT J
      NEXT I

' set initial conditions on a per plant basis
      PLSTN = PLSTNI
' CPOOL is supposed to start at 10% leaf d.wt
      CPOOL = 0.1 * WLVSI
      TDML = WLVSI
      TDMS = WSTMI
      TDMF = 0.
      DMGF = 0.
      DMGL = WLVSI
      RETURN

'*****************************************************************************************
SUBROUTINE HOURLY CALCULATION OF TEMP-DEPENDENT FACTOR
'*****************************************************************************************
DEVFAST:
' temperature effect on development rate
      FOR TT=1 TO 6:XVAL(TT)=GENTEM(TT):XARG(TT)=XTEM(TT):NEXT TT
      TEMFCF=FNTABEX(GENTEM,XTEM,TMPA,6)
' leaf aging
      FOR TT=1 TO 9:XVAL(TT)=RDVLVT(TT):XARG(TT)=XLV(TT):NEXT TT
      RDVLVF=FNTABEX(RDVLVT,XLV,TMPA,9)
' fruit aging
      FOR TT=1 TO 9:XVAL(TT)=RDVFRT(TT):XARG(TT)=XFR(TT):NEXT TT
      rdvfrf = FNTABEX(rdvfrt,xfr,TMPA,9)
      RETURN

'*************************************************************************
'SUBROUTINE FOR CALCULATION OF INSTANTANEOUS GROSS PHOTOSYNTHESIS
'*************************************************************************
PHOTO:
     RAH=RAH*0.3
     RLSI=RLSI*0.15
     TTMPA=TMPA
    Q10=2.D0
```

```
  Tleaf=TMPA''Tv
  CCO2L=CO2L:'350' 'CO2 CONCENTRATION IN ul/l
  LAI=PLAR2*PLM2
  SCP=0.2
  Q10=2.0
  RDTL20=1.1364: 'LEAF DARK RESPIRATION AT Tleaf=20C(umol CO2/m2 leaf/s)
  KDF=0.68:'0.58: 'MEASURED EXTINCTION COEFFICIENT OF DIFFUSE PAR
  CD=42.7+1.68*(Tleaf-25)+0.012*(Tleaf-25)^2: 'CO2 COMPENSATION POINT(PPMV)
  EFF=0.084*(CCO2L-CD)/(CCO2L+2*CD): ' INITIAL LIGHT USE EFFICIENCY
                                     '(molCO2/mol photon absorbed)
  RDTL=RDTL20*Q10^(0.1*(Tleaf-20))
  rm=200:   'mesophyll resistance(s/m)
     if Tleaf<=5 or Tleaf>=40 then rm=10000000000000
     if Tleaf>5 and Tleaf<15  then rm=2000/(Tleaf-5)
     if Tleaf>30 and Tleaf<40 then rm=200/(1-(Tleaf-30)/10)
  RM=RM:'*0.4
  PMM=45.0
  IF TLEAF<15. OR TLEAF>50. THEN
  PMM=(45*100/RM)/0.5
  END IF
  IF TLEAF>40 AND TLEAF<=50. THEN
  PMM=45.*(1-(TLEAF-40.)/10.)
  END IF
  IF Tleaf>=40 OR  TLEAF<=5 THEN PMM=0
  PNC=41.5843*(293/(273+Tleaf))*(CCO2L-CD)/(rm+1.36*rah+1.6*rlsi)
  PNMAX=PNC
  IF PNC>PMM THEN PNMAX=PMM
  PMAX=PNMAX+RDTL: 'max. gross photosynthetic rate(umolCO2/m2-leaf/s)
 CALCULATION OF REFLECTION FOR HORIZONTAL AND SPHERICAL LEAF ANGLE
 DISTRIBUTION
  SQV=SQR(1-SCP)
  REFH=(1-SQV)/(1+SQV)
  REFS=REFH*2/(1+2*COSZ)
CALCULATION OF EXTINCTION COEFFICIENT FOR DIRECT RADIATION AND
TOTAL DIRECT FLUX
 IF COSZ<0.01 THEN COSZ=0.01
  CLUSTF=KDF/(0.8*SQV)
  KBL = (0.5/COSZ)*CLUSTF
  KDRT= KBL*SQV
SELECTION OF CANOPY DEPTH AND CANOPY ASSIMILATION IS SET TO ZERO
  GPF=0:  ' INSTANTANEOUS CANOPY ASSIMILATION RATE
  FOR IZ=1 TO 3
  LAIC=LAI*XGAUSS(IZ)
ABSORBED DIFFUSE PAR, TOTAL DIRECT FLUX AND DIRECT COMPONENT OF
DIRECT FLUX PER UNIT LEAF AREA
  VISDF=(1-REFH)*PPFDD*KDF*EXP(-KDF*LAIC)
  VIST =(1-REFS)*PPFDB*KDRT*EXP(-KDRT*LAIC)
  VISD =(1-SCP)*PPFDB*KBL*EXP(-KBL*LAIC)
ABSORBED FLUX(umol/M2 LEAF/S) AND ASSIMILATION OF SHADED LEAVES
  VISSHD=VISDF+VIST-VISD
  IF PMAX>0 THEN
     GPFSH=PMAX*(1-EXP(-VISSHD*EFF/PMAX))
  ELSE
     GPFSH=0
  END IF
DIRECT FLUX ABSORBED BY LEAVES PERPENDICULAR TO DIRECT PAR AND
ASSIMILATION OF SUNLIT LEAF
  VISPP = (1-SCP)*PPFDB/COSZ
  GPFSUN=0
  FOR IP=1 TO 3
  VISSUN=VISSHD+VISPP*XGAUSS(IP)
  IF PMAX>0 THEN
     GPFS=PMAX*(1-EXP(-VISSUN*EFF/PMAX))
  ELSE
     GPFS=0
  END IF
  GPFSUN=GPFSUN+GPFS*WGAUSS(IP)
  NEXT IP
FRACTION OF SUNLIT LEAF AREA(FSLLA) AND LOCAL ASSIMILATION RATE(GPL)
  FSLLA=CLUSTF*EXP(-KBL*LAIC)
  GPL=FSLLA*GPFSUN+(1-FSLLA)*GPFSH
INTEGRATION OF LOCAL ASSIMILATION TO CANOPY ASSIMILATION(GPF)
  GPF=GPF+GPL*WGAUSS(IZ)
  NEXT IZ
  GPF=GPF*LAI/PLM2
CONVERSION FROM umol CO2/plant/s TO umol CO2/plant/day
  GPF=GPF*86400
```

```
' CONVERSION FROM umol CO2  TO g CH2O
   GPF=GPF*30/1000000
   RETURN

'*****************************************************************************************
SUBROUTINE HOURLY CALCULATION OF MAINTENANCE RESPIRATION
'*****************************************************************************************
 RESP:
'Q10=1.4
' Effect of temperature on maintenance respiration
     RMRL=0.03:RMRS=0.015:RMRF=0.01: ' MAINTENANCE RESPIRATION RATE AT 25C
     RMAINTF = (RMRL*TDML2+RMRS*TDMS+RMRF*DMGF) * (Q10^(0.1*TMPA-2.5))
     RETURN

'*****************************************************************************************
SUBROUTINE DAILY CALCULATION OF ORGAN DEVELEOPMENT
'*****************************************************************************************
 DEVRATE
' ************************************************************************
' node number
      FOR TT=1 TO 8:XVAL(TT)=GENRAT(TT):XARG(TT)=XGEN(TT):NEXT TT
      GENR = TEMFAC*FNTABEX(GENRAT,XGEN,PLSTN,8)
      PLSTN = PLSTN + GENR
' truss number
      NBRU = INT((PLSTN-FTRUSN+(1.+TPL)/TPL)*TPL/(1.+TPL))
      NBRU = FNMAX(0,NBRU)
' number of trusses bearing fruit
      NBRUP=INT((PLSTN-FTRUSN-FRLG+(1.+TPL)/TPL)*TPL/(1.+TPL))
      NBRUP=FNMAX(0;NBRUP)
' leaf number
      NBLV = INT(PLSTN) - NBRU
' number of fruits (diameter>20mm) on each truss
' Note. FPN is now the ratio of fruit initiation on a truss per node initiation
      trcnf = 0.
        FOR I=1 TO NBRUP
         xx=xnft(I)
' FPN is now replaced by FPNPT=0.8
         XNFT(I) = FNAMIN1 (NBFPT(I) , XNFT(I)+GENR*FPNPT)
         rcnf(i)=xnft(i)-xx
         trcnf=trcnf+rcnf(i)
' rncf(i) AND trcnf ISt USED IN LOSRATE
     NEXT I
      tnf=tnf+trcnf
' ************************************************************************
' aging and sink strength (determined for leaves and stems
' and for each fruit on each reproductive unit)
      PTNLVS = 0.
      PTNSTM = 0.
      PTNFRT = 0.
' leaves and stems
' new leaves start aging when a new truss appears (2 leaves below
' and 1 leaf above)
      FOR I=1 TO NBRU
         IF AGLS(I)=-1*EPS THEN
          PLE(I)=0.
          PGL(I)=0.
          PGS(I)=0.
         ELSE
          AGLS(I) = FNAMIN1 (1.,AGLS(I)+RDVLV)
          XBOX = 100.*AGLS(I)
           FOR TT=1 TO 12:XVAL(TT)=POL(TT):XARG(TT)=BOX(TT):NEXT TT
          IF I=1 THEN
           PLE(I)=temfac*FNMAX(0.,FNTABEX(POL,BOX,XBOX,12))*FTRUSN
          ELSE
           PLE(I)=temfac*FNMAX(0.,FNTABEX(POL,BOX,XBOX,12))/TPL
          END IF
          pgl(i) = ple(i)*(1.+frpt)/slamn
          PGS(I) = PGL(I)*FRST
         END IF
         PTNLVS = PTNLVS + PGL(I)
         PTNSTM = PTNSTM + PGS(I)
      NEXT I
   fruit
      FOR I=1 TO NBRU
      FOR J=1 TO INT(XNFT(I))
      agf(i,j) = FNamin1 (1.,agf(i,j)+rdvfr)
      XBOX = 100.*AGF(I,J)
```

```
        IF DWF(I,J)<0 THEN
                        PGF(I,J) = 0.
                        ELSE
         FOR TT=1 TO 12:XVAL(TT)=POF(TT):XARG(TT)=BOX(TT):NEXT TT
        PGF(I,J)= temfac * FNMAX(0.,FNTABEX (POF,BOX,XBOX,12))
        PTNFRT = PTNFRT + PGF(I,J)
        END IF
    NEXT J
    NEXT I
' total sink demand
      PNGP = PTNLVS + PTNSTM + PTNFRT
      RETURN

'*****************************************************************************************
SUBROUTINE for CALCULATION OF ORGAN LOSS RATE
'*****************************************************************************************
LOSRATE:
' number of "aborted" fruits
      TABNF = 0.
      IF TDMF<EPS THEN GOTO 6010
      FABOR = FNAMIN1(1.,(0.67-ABORMX*SOSIR))
      FABOR = FNMAX(0.,FABOR)
      TABNF =FABOR*TRCNF
  6010  'CONTINUE
      TABF = TABF + TABNF
      TNSF = TNF - TABF
' location of "aborted" fruits: distal position
      BBB=0.
      FOR I=1 TO NBRUP
         IF RCNF(I) =0 OR XNFT(I)<=2. OR BBB >=TABNF THEN GOTO 6040
         ABNF(I) = FNAMIN1(4.,RCNF(I))
         ABNF(I) = FNAMIN1(ABNF(I),TABNF-BBB)
         ABNF(I) = FNAMIN1(ABNF(I),XNFT(I)-2)
         ABNF(I) = FNAMIN1(ABNF(I),-ABOR(I))
         ABNF(I) = FNMAX (0.,ABNF(I))
         BBB = BBB + ABNF(I)
         ABOR(I) = ABOR(I) + ABNF(I)
         NSF(I) = INT(XNFT(I) - ABOR(I))
         IF ABOR(I)<1 THEN GOTO 6040
         FOR J = NSF(I)+1 TO INT(XNFT(I))
               DWF(I,J) = -1*EPS
          NEXT J
   6040     NEXT I
      RETURN

'*****************************************************************************************
SUBROUTINE CALCULATION OF DRY MATTER GROWTH RATE
'*****************************************************************************************
 DMRATE:
      ASCSP = 0
      TDML = 0.
      TDML2 = 0
      DMGL = 0.
      TDMS = 0.
      TDMF = 0
      DMGF = 0.
      TNMF = 0.
      DMMF = 0.
      PLAR = 0.
      PLAR2 = 0.
' leaf dry weight
      FOR I=1 TO NBRU
         XASC = FNAMIN1(PGL(I),PGL(I)*SOSIR)
         ASCSP = ASCSP + XASC
         DWL(I) = DWL(I) + XASC
' dry weight of growing leaves
         IF AGLS(I)<1 THEN DMGL = DMGL + DWL(I)
' Now leaf area expansion is either potential or limited by a max SLA
' It is available for each unit
         XLA(I) = XLA(I) + FNAMIN1 (PLE(I) , XASC*SLAMX/(1.+FRPT))
         TDML = TDML + DWL(I)
' total leaf area
         PLAR = PLAR + XLA(I)
         IF AGLS(I)>0 THEN
                        PLAR2 = PLAR2 + XLA(I)
                        TDML2 = TDML2 + DWL(I)
         END IF
```

```
      NEXT I
' stem dry weight
      FOR I=1 TO NBRU+1
         XASC = FNAMIN1(PGS(I),PGS(I)*SOSIR)
         ASCSP = ASCSP + XASC
         DWS(I) = DWS(I) + XASC
         TDMS = TDMS + DWS(I)
      NEXT I
' fruit dry weight
      TDMF = 0.
      XASC = 0.
      FOR I=1 TO NBRU
         DWTR(I) = 0.
         NFT=INT(XNFT(I))
         FOR J=1 TO NFT
' for "aborted" fruit, DWF = -EPS
            IF agf(I,J)< 1 AND dwf(I,J)>=0. THEN
            XASC = FNAMIN1(PGF(I,J),PGF(I,J)*SOSIR)
            ASCSP = ASCSP + XASC
            DWF(I,J) = DWF(I,J) + XASC
            end if
            if dwf(i,j)>0. THEN DWTR(I) = DWTR(I) + DWF(I,J)
' mature fruit
            IF AGF(I,J)=1. AND DWF(I,J)>0 THEN
            DMMF = DMMF + DWF(I,J)
            TNMF = TNMF + 1.
            END IF
          NEXT J
            IF AGF(I,nbfpt(i))>=1. THEN AGLS(I) = -1*EPS
            TDMF = TDMF + DWTR(I)
         NEXT I
' carbohydrate pool
            CPOOL = FNMAX(0.,(RCDRW-ASCSP)/GREF)
' when CPOOL is higher than a threshold value then GP is limited
            CPOOLMX = 0.06*TDML/((1.+FRPT)*GREF)
      IF CPOOL> CPOOLMX THEN
         GP = GP - (CPOOL - CPOOLMX)
         CPOOL = CPOOLMX
      END IF
      RETURN

'*************************************************************************
' SUBROUTINE CALCULATES transpiration AND LATENT HEAT FLUX
'*************************************************************************
TRANS

'**** this subroutine is the same as in "GREENCLI"

'*************************************************************************
'SUBROUTINE FOR CALCULATION OF BOUNDARY LAYER RESISTANCES
'*************************************************************************
RESIST:

'**** this subroutine is the same as in "GREENCLI"

'*************************************************************************
'Subroutine for calculation of sun-earth astronomical parameters and transmission
'of solar radiation through greenhouse cover layer
'*************************************************************************
SOLAR:

'**** this subroutine is the same as in "GREENCLI"

'*************************************************************************
'SUBROUTINE CALCULATES ABSORPTION, TRANSMISSION AND REFLECTION OF COVER
'*************************************************************************
CALAB:

'**** this subroutine is the same as in "GREENCLI"


'*************************************************************************'
SUBROUTINE OUTPUT FOR SIMULATION RESULT
'*************************************************************************
OUTPUT:
' per plant results:
' # nodes, # leaves, # trusses, leaf DW, stem DW, fruit DW, total DW
```

```
' leaf area, mature fruits DW, # set fruits, # "aborted" fruits,
' #mature fruits

   IF DATE=280 OR DATE=285 OR DATE=287 OR DATE=294 OR DATE=296 OR
DATE=300 THEN
      PRINT#16,USING"#####";DATE;:PRINT#16,USING"####.##";NBLV;NBRU;PLSTN;
      PRINT#16,USING"####";TNF;TNSF;TNF-TNSF;TNMF
   END IF

   IF DATE=303 OR DATE=307 OR DATE=311 OR DATE=315 OR DATE=318 OR
DATE=322 THEN
      PRINT#16,USING"#####";DATE;:PRINT#16,USING"####.##";NBLV;NBRU;PLSTN;
      PRINT#16,USING"####";TNF;TNSF;TNF-TNSF;TNMF
   END IF

   IF DATE=325 OR DATE=330 OR DATE=335 OR DATE=339 OR DATE=343 THEN
      PRINT#16,USING"#####";DATE;:PRINT#16,USING"####.##";NBLV;NBRU;PLSTN;
      PRINT#16,USING"####";TNF;TNSF;TNF-TNSF;TNMF
   END IF

   IF DATE=280 OR DATE=287 OR DATE=294 OR DATE=301 OR DATE=308 THEN
      PRINT#15,USING"#####";DATE;
      PRINT#15,USING"#####.##";(TDML+CPOOL)*PLM2;TDMS*PLM2;TDMF*PLM2;
                               (TDML+TDMS+TDMF+CPOOL)*PLM2;
      PRINT#15,USING"#####.##";PLAR*10000./TDML;PLAR*PLM2;DMMF*PLM2;SOSIR
   END IF

   IF DATE=315 OR DATE=322 OR DATE=329 OR DATE=336 OR DATE=343 THEN
      PRINT#15,USING"#####";DATE;
      PRINT#15,USING"#####.##";(TDML+CPOOL)*PLM2;TDMS*PLM2;TDMF*PLM2;
                               (TDML+TDMS+TDMF+CPOOL)*PLM2;
      PRINT#15,USING"#####.##";PLAR*10000./TDML;PLAR*PLM2;DMMF*PLM2;SOSIR
   END IF
7082   RETURN
```

APPENDIX 2. LIST OF GREENHOUSE MICROCLIMATE MODEL "GREENCLI

1. Definition of the abbreviations used in the model

AA	absorptivity of greenhouse cover
ABCB	average absorptance of beam radiation at the wall surface
ABG	soil surface absorptance for total solar radiation
ABM	mulching absorption for total solar radiation
ABSS	screen absorptance for total solar radiation
ABV	canopy absorptance for total solar radiation
AC	surface area of walls(m')
AG	ground area of greenhouse(m')
aex	thermal expansion rate of air(1/K)
AINC	average incidence angle of solar beam on wall surface
ATR	atmospheric transmission rate
Aw	window area opened(m')
AZIM	azimuth angle of sun
CFCS	average shafe factor which wall surface see the sky
CFGC	average shafe factor which outside ground see the wall surface
CLF	clustering factor
COSZ	cosine of solar zenith
Coni	cover inner surface-air conductance(m/s)
Conm	mulch-air conductance(m/s)
cp	specific heat of air at constant pressure(J/kg/K)
delta	slope of saturation vapor pressure curve(Pa/K)
DL	characteristic length of leaf(m)
DTRN	daily tatal transpiration(g/plant/day)
EPC	cover emissivity for long-wave radiation
EPG	soil surface emissivity for long-wave radiation
EPM	mulching emissivity for total solar radiation
EPS	screen emissivity for long-wave radiation
EPV	canopy emissivity of long-wave radiation
EQOT	equation of time in hour
EWL	east-west length of greenhouse(m)
Eci	latent heat flux between cover inner surface and air(w/m')
Eg	latent heat flux from soil surface(w/m')
Em	latent heat flux between mulch surface and air(w/m')
Ev	latent heat flux between air and canopy(w/m')
Even	latent heat flux by ventilation(w/m')
FI	refraction index of covering material
FL	film thickness(mm)
FNFA	fenestration factor
gamma	psychrometric constant(Pa/K)
GSOL	global solar radiation in greenhouse(W/m^2)
GSOLB	beam solar radiation in greenhouse(W/m^2)
GSOLD	diffuse solar radiation in greenhouse(W/m^2)
GZL	gabble height of greenhouse(m)

Ghf	soil surface heaf flux(w/㎡)
Gr	Grashof number
HANG	hour angle
Hci	sensible heat between inner cover surface and air(w/㎡)
Hco	sensible heat flux between outer cover surface and air(w/㎡)
Hg	sensible heat flux from soil surface(w/㎡)
Hm	sensible heat flux between mulch and air(w/㎡)
Hmg	sensible heat between mulch and soil(w/㎡)
Hv	sensible heat flux between air and canopy(w/㎡)
Hven	sensible heat flux by ventilation(w/㎡)
kapa	thermal diffusivity of air(m^2/s)
KC	light extinction coefficient of film(1/mm)
KDRL	extinction coefficient for direct radiation
KDRT	extinction coefficient for total direct radiation
KKH	cover outer surface-air conductance(m/s)
LAI	leaf area index
lambda	latent heat of water vaporization(J/g)
LAT	lattitude in degree
LON	longitude in digree
LVPD	leaf vapor pressure deficit(Pa)
MT	fraction of diffuse radiation
nhu	kinematic viscosity of air(m^2/s)
NSL	north-south length of greenhouse(m)
Nuf	Nusselt number for forced convection
Nus	Nusselt number for free convection
Nusm	Nusselt number for mixed convection
PAR	photosynthetically active radiation in greenhouse(W/m^2)
PARB	photosynthetically active beam radiation in greenhouse(W/m^2)
PARD	photosynthetically active diffuse radiation in greenhouse(W/m^2)
PCSH	proportion of shaded wall surface area
PCSL	proportion of sunlit wall surface area
PHI	ventilation rate(㎥/sec)
PPFD	photosynthetic photon flux density in greenhouse($\mu mol/m^2/s$)
PPFD	diffuse photosynthetic photon flux density in greenhouse($\mu mol/m^2/s$)
PPFDB	beam photosynthetic photon flux density in greenhouse($\mu mol/m^2/s$)
RB	beam solar radiation(W/m^2) outside greenhouse
RD	diffuse solar radiation(W/m^2) outside greenhouse
RE	reflectivity of greenhouse cover
RECB	average reflectance of beam radiation at the wall surface
REG	soil surface reflectance for total solar radiation
RELC	cover reflectance of long-wave radiation
RELG	soil surface reflectance of long-wave radiation
RELM	mulching reflectance of long-wave radiation
RELS	screen transmittance of long-wave radiation
RELV	canopy reflectance of long-wave radiation
RES	screen reflectance for total solar radiation

REV	canopy reflection coefficient for total solar radiation
REVH	canopy reflection coefficient for diffuse radiation
REVS	canopy reflection coefficient for direct radiation
REXT	extraterestrial solar radiation(W/m^2)
RHAI	inside relative humidity(%)
RHAO	outside relative humidity(%)
RREM	mjulching reflectance for total solar radiation
Rah	Boundary layer resistance(s/m)
Rahmg	contact resistance between soil and mulch(s/m)
Re	Reynold's number
rhoa	air density(kg/m^3)
rhod	reflectance of deep canopy
rhos	reflectance of soil
Rlsi	internal(stomatal) resistance(s/m)
Rn	net radiation of canopy(w/m^2)
Rnc	net radiation of cover(w/m')
Rng	net radiation of soil(w/m')
Rnm	net radiation of mulch(w/m')
Rnv	net radiation of canopy(w/m')
SDEC	solar declination angle
SOLC	solar constant(W/m^2)
SSI	shaded area of wall on greenhouse bed
TALP	tangent of solar elevation
TMPA	inside air temperature(℃)
TR	transmissivity of greenhouse cover
TRAN	instantaneous transpiration rate(g/m^2-ground/s)
TRCB	average transmittance of beam radiation at the wall surface
TRCGB	average transmittance of beam radiation at the vegetation surface
TRD	diffuse radiation transmittance(transmittance at incident angle of 64 gegree)
TRLC	cover transmittance of long-wave radiation
TRLM	canopy transmittance of long-wave radiation
TRLS	screen transmittance of long-wave radiation
TRLV	canopy transmittance of long-wave radiation
TRM	mulching transmittance for total solar radiation
TRV	canopy transmittance for total solar radiation
Tair	outside air temperature(℃)
Tc	cover temperature(℃)
Tg	soil surface temperature(℃)
Tleaf	leaf temperature(℃)
Tm	mulch temperature(℃)
Ts	screen temperature(℃)
Tv	canopy temperature(℃)
U	inside wind speed(m/sec)
VPA	vapor pressure of outsixe air(Pa)
VPAi	inside vapor pressure(Pa)
VPb	vapor pressure deflicit of inside air(Pa)

WAT	water content of soil(㎤/㎤)
WAZI	azimuth angle of greenhouse walls
WINC	solar incidence angle on wall surface
Ws	outside wind speed(m/sec)
WSLO	slope angle of greenhouse walls(degree)
WZL	wall height of greenhouse(m)

2. Listing of the greenhouse microclimate model GREENCLI

```
'*************************************************************************
'PROGRAMM FOR PREDICTING the greenhouse climate
'*************************************************************************
 DIM VB(10), MA(10, 10), DX(10), T(10), TN(10), DELZ(21)
 DIM kcs(21), TSO(21), TSN(21),VBB(10,10),IR(10)
 DIM AAA(21), CCC(21), BBB(21), DDD(21),XGAUSS(3),WGAUSS(3)
 OPEN"O",#2,"C:\TOMGRO\greentom\resul295.DAT"
 OPEN"I",#1,"C:\TOMGRO\greentom\outcl295.DAT"
 OPEN"O",#3,"C:\TOMGRO\greentom\reSOL295.DAT"
'*************** data for GAUSSIAN integration
    DATA 0.112702,0.500000,0.887298
    READ XGAUSS(1),XGAUSS(2),XGAUSS(3)
    DATA 0.277778,0.444444,0.277778
    READ WGAUSS(1),WGAUSS(2),WGAUSS(3)
'*************************************************************************
' IF SCREEN IS OPENED, SCRI=0 ELSE SCRI=1
    SCRI=0
' IF FAN VENTILATOR IS OPERATED, FANI=1 ELSE FANI=0
    FANI=0
    PHIFAN=0.36:'M3/S
' IF HEATER ARE OPERATED, THEN heater=1 ELSe heater=0
    heater=1
'    QQH=3900:'J/SEC
'*************************************************************************
'  IF WALL IS DOUBLE-LAYERED, THEN DOU=1, ELSE DOU=0
   DOU=1
'*************************************************************************
'  input leaf area index
    LAI= 1.2
    PLTH=0.495+0.439*LAI:'PLANT HEIGHT IN m
'****** define function for calculation of saturation vapor pressure and its slope
  def fn es(x)=610.78D0*exp(17.269D0*x/(x+237.30D0))
  def fn esp(x)=610.78D0*exp(17.269D0*x/(x+237.30D0))*(4097.9337D0/((x+237.30D0)^2.D0))
'***************START DAILY CACULATION*********************************
   DATE = 295:  'JULIAN DAY OF YEAR
   WAT = .20D0: ' SOIL WATER CONTENT(M3/M3)
'*************************************************************************
'READ GREENHOUSE PARAMETER
'*************************************************************************
  DATA 90,90,-90,-90,0,180:'AZIMUTH ANGLE OF GREENHOUSE WALL AND ROOF
  DATA 90,23.75,23.75,90,90,90:' SLOPE OF WALL AND ROOFS
  DATA 32.2,38.23,38.23,32.2,14.25,14.25:'WALL SURFACE AREA
  DATA 1.015,70,14,5,2.3,1.1:'FAFN,AG,NSL,EWL,WZL,GZL
  DATA 1.5,0.165,0.05,37 25,126.983:'FI, KCC,FFL,LAT,LON
  FOR I=1 TO 6:READ WAZI(I):NEXT I
  FOR I=1 TO 6:READ WSLO(I):NEXT I
  FOR I=1 TO 6:READ AC(I):NEXT I
  READ FAFN,AG,NSL,EWL,WZL,GZL
  READ FI,KCC,FFL,LAT,LON
  ACT=0:FOR I=1 TO 6:ACT=ACT+AC(I):NEXT I
  GOSUB INPARM:'input parameters
'*************************************************************************
  LLAT = 37.15D0 * 3.141592D0 / 180.D0:' lattitude
'***************** CALCULATE HOURLY ***************************************
   NFAST = 24.D0 * 6.D0
         DTIM = 3600.D0 * 24.D0 / NFAST: 'TIME STEPSIZE
         dum=0
   FOR    JDAY=1 TO 365
         DATE=JDAY
        FOR JF = 1 TO NFAST
        if eof(1) then goto 2
        INPUT#1,JJDAY,TTIME,tair,ws,solrad,outh
        VPA=(FN ES(Tair))*OUTH/100.D0
        VPAII=FN ES(WBt)
        TFAST = (JF - 1) * 24.D0 / NFAST+0.083D0
    GOSUB SOLAR
'******* INTIALIZE ABOVE GROUND WEATHER VARIABLE
NNN = 6
      dum=dum+1
      if dum>1 then goto 1
      FOR YY = 1 TO NNN+1
        T(YY) = Tair
      IF YY=6 THEN T(6)=CO2O+300
      IF YY=7 THEN T(7)=VPA+1000.D0
      NEXT YY
```

```
        Tc =T(1):Tm=T(2)
        Tg =T(3):TMPA=T(4):Tv=T(5):CO2L=T(6)
        VPai=T(7)
1     'continue
      DUM1=0
11    DUM1=DUM1+1
'********** EFFECTIVE ATMOSPHERIC EMMISSIVITY(EPa)*********************
      AHS = 2.167D0 * VPA / (Tair + 273.16D0)
      EPA = .58D0 * (AHS ^ (1.D0 / 7.D0))
      IF TTIME<1000 THEN EPA=0.83D0
      IF SOLRAD > 0 THEN
        CLOUD = 2.33D0 - 3.33D0 * ATR: 'CLOUD COVER
        IF CLOUD < 0 THEN CLOUD = 0
        IF CLOUD > 1 THEN CLOUD = 1
        EPA = (1-.84D0 * CLOUD) *  .58D0 * (AHS  ^ (1.D0 /  7.D0)) +  .84D0 * CLOUD:
      END IF
'********* HEATER OPERATION *********************************************
 QQH=0
 IF HEATER=1 AND TMPA=<9.5 THEN QQH=1850:'3900
'*********START ITERATION;NEWTON-RAPHSON  METHOD**************
       MAXIT = 100
       SIGNUM = 1.D-5
    FOR ITER = 1 TO MAXIT
'**** NUMERICAL CALCULATION OF DEIVATIVES OF ENERGY BALANCE ****
'**** EQUATIONS
      RRRI=10.0D0
      HIJJ=5.0D-3
      FOR CX=1 TO 2
      HIJ=HIJJ
      FOR KZ=0 TO NNN
         IF SOLRAD<=0 THEN
           RB=0.:RD=0.: GSOL=0.
         END IF
       IF KZ=0 THEN GOTO 9999
       IF CX=2 THEN HIJ=RRRI*HIJJ
         IF KZ=1 THEN Tc=Tc+HIJ
         IF KZ=2 THEN Tm=Tm+HIJ
         IF KZ=3 THEN Tg=Tg+HIJ
         IF KZ=4 THEN TMPA=TMPA+HIJ
         IF KZ=5 THEN Tv=Tv+HIJ
         IF KZ=6 THEN CO2L=CO2L+HIJ
'*****************************************************************************
' CALCULATION OF NET RADIATION
'*****************************************************************************
9999   GOSUB NETRAD
'*****************************************************************************
'   CALCULATION OF  BOUNDARY  LAYER  RESISTANCES AND  SOIL  AND MULCH
CONTACT RESISTANCE
'*****************************************************************************
'************* OVER OUTER COVER
      KKH=3.49D0*WS
      Hco=KKH*(Tc-Tair)
      FDOU=2.D0*(NSL+EWL)*WZL/ACT
      FDOU=1-FDOU*1.35D0:'0.5
      IF DOU=0 THEN FDOU=1
'******* CALCULATION OF VENTILATION RATE ************************
   WOH=0.0D0:'WOH=HEIGHT OF SIDE WINDOW OPENING
      Aw=(NSL-2)*WOH*2''5.13:'area of side window opened
      ACTI=ACT-Aw
       PHI=(0.170D0+0.045D0*Aw*Ws)*FDOU''SIDE OPEN VENTILATION
       IF FANI=1 THEN PHI=PHI+PHIfan
'***************  OVER INNER COVER
     U=PHI/Af:'wind speed in the greenhouse
    CONi=1.525D0*((ABS(Tc-TMPA))^(1.D0/3.D0))/1000.D0:' conductance(m/s)
    CONih=CONi
    IF DOU=1 THEN CONih=CONi*FDOU
    Hci=CONih*(TC-TMPA)*RHOa*Cp
    QQQc=0.622D0*(FN_ES(Tc))/(1013.D0*100.D0)
    QQQai=0.622D0*VPai/(1013.D0*100.D0)
    Eci=CONi/0.93D0*RHOa*LAMBDA*(QQQc-QQQai)
    IF QQQc>QQQai THEN Eci=0.0D0
'*****************************************************************************
' VENTILATION FLUX CALCULATION
'*****************************************************************************
     QQQa=0.622D0*VPa/(1013.D0*100.D0)
     Hven= RHoa*Cp*PHI*(TMPA-Tair)/AG
     Even= RHOa*LAMBDA*PHI*(QQQai-QQQa)/AG
```

```
'*************** OVER MULCH *****************
   CONm=0.4D-2*(ABS(TM-TMPA)^(1.D0/3.D0)):'conductance(m/s)
    IF Tm-TMPA<0 THEN CONm=CONm/2.D0
   QQQm=0.622D0*(FN ES(Tm))/(1013.D0*100.D0)
   Hm=FMA*RHOa*Cp*(Tm-TMPA)*CONm:'/RAHm
   Em=FMA*LAMBDA*RHOa*(QQQm-QQQai)*CONm/0.93D0
    IF QQQai<QQQm THEN Em=0.D0
'*************** OVER SOIL SURFACE ***************
   CONg=0.4D-2*(ABS(Tg-TMPA)^(1.D0/3.D0)):'CONDUCTANCE(M/S)
    IF Tg-TMPA<0 THEN CONg=CONg/2.D0
   QQQSg=0.622D0*(FN ES(Tg))/(1013.D0*100.D0)
  '**********soil surface humidity calculation
  'for sandy load
   Thethas=0.42D0
   Thethar=0.0825
   mmmm=0.433D0
   nnnn=1.76D0
   AAAA=0.0872D0
  'suction head(SUCH, IN m)
   watt=WAT
   SUCH= (((WATT-THETHAR)/(THETHAS-THETHAR))^(-1/MMMM)-1)^(1/NNNN)/AAAA
   evprat= EXP(-SUCH/46.97D0/(Tg+273.16D0))
   QQQG=EVPRAT*QQQSg
   Eg=(1.D0-FMA)*RHOa*LAMBDA*(QQQg-QQQai)*CONg/0.93D0
   Hg=(1.D0-FMA)*RHOa*Cp*(Tg-TMPA)*CONg
'*************** BETWEEN SOIL AND MULCH
   RAHmg = MSDIS / KAPA: 'contact resistance between soil and mulch
   Hmg=FMA*RHOa*Cp*(Tm-Tg)/RAHmg
'******************** Heater Operation ************************************************
   GOSUB TRANS:'TRANSPIRATION CALCULATION
   GOSUB PHOTO2:'CALCULATION OF  PHOTOSYNTHESIS
'************GROUND HEAT FLUX(Ghf, w/M2)
   Ghf = -kcs(1)*(Tg-TSO(2))-Ccvs*(Tg-TSO(1))*DELZ(1)/2.D0/DTIM
'************ ENERGY BALANCE AT EACH LAYER
'**** COVER LAYER
    VBB(1,KZ) = Rnc - Hco - Hci - Eci
'**** MULCH LAYER
    VBB(2,KZ) = Rnm - Hm -Hmg-Em
'**** SOIL SURFACE LAYER
    VBB(3,KZ) = Rng + Hmg + Ghf-Eg-Hg
'**** SENSIBLE HEAT OF INTERNAL AIR
    VBB(4,KZ) = Hci*ACTI/Ag + Hm + Hv +Hg+ QQH/Ag-Hven
'**** VEGETATION LAYER
    VBB(5,KZ) = RNv-Hv-Ev
'************ CO2 BALANCE ************************************************************
    SOURC=0.104D-4*3^(TSN(2)/10):'8.8D-5*3.D0:'SOIL RESPIRATION
    VBB(6,KZ)=PHI/ag*(CO2O-CO2L)*0.52923D0/(TMPA+273.D0)-GPFM*4.4D-5
              +RMAINTF+SOURC
      IF KZ=1 THEN Tc=Tc-HIJ
      IF KZ=2 THEN Tm=Tm-HIJ
      IF KZ=3 THEN Tg=Tg-HIJ
      IF KZ=4 THEN TMPA=TMPA-HIJ
      IF KZ=5 THEN Tv=Tv-HIJ
      IF KZ=6 THEN CO2L=CO2L-HIJ
      IF KZ=0 THEN GOTO 25
    IF CX=2 THEN GOTO 35
    FOR KKZ=1 TO NNN
    MA(KKZ,KZ)=(VBB(KKZ,KZ)-VBB(KKZ,0))/HIJ
    NEXT KKZ
35  FOR KKZ=1 TO NNN
    TEMP=MA(KKZ,KZ)
    MA(KKZ,KZ)=(VBB(KKZ,KZ)-VBB(KKZ,0))/HIJ
    MA(KKZ,KZ)=(RRRI^2.D0*TEMP-MA(KKZ,KZ))/(RRRI^2.D0-1.D0)
    NEXT KKZ
25  VB(KZ)=VBB(KZ,0)
    NEXT KZ
    NEXT CX
   GOSUB MATRIX
     FOR KKV=1 TO NNN
     dx(KKV)=-dx(KKV)
     NEXT KKV
     TN(1) = Tc + DX(1)
     TN(2) = Tm + DX(2)
     TN(3) = Tg + DX(3)
     TN(4) = TMPA+DX(4)
     TN(5) = Tv+DX(5)
     TN(6) = CO2L+DX(6)
```

```
    TN(7)=VPAI
    Tc=TN(1)
    Tm=TN(2)
    Tg=TN(3)
    TMPA=TN(4)
    Tv=TN(5)
    CO2L=TN(6)
    VPAI=TN(7)
'********* CALCULATE VAPOR PRESSURE *****************************
  RHOa=1.204D0
  CHIv=0.622D0*RHOA*(FN ES(TMPA))/(1013.D0*100.D0)
  CHIv=CHIv+EPP*(RAH/0.93)*Rnv/2/LAI/LAMBDA
  CHIc=0.622D0*RHOA*(FN ES(Tc))/(1013.D0*100.D0)
  CHIm=0.622D0*RHOA*(FN ES(Tm))/(1013.D0*100.D0)
  CHIo=0.622D0*RHOA*VPA/(1013.D0*100.D0)
  CHIg=0.622D0*RHOA*(FN ES(Tg))/(1013.D0*100.D0)*EVPRAT
  CHIi=0.622D0*RHOA*VPAi/(1013.D0*100.D0)
   EPP=DELTA/GAMMA
   Gtran=2.D0*LAI/((0.93D0+EPP)*RAH+RLSI)
   Gcond=ACTI/AG*CONi/0.93D0
   Gvent=PHI/Ag
   Ggrnd=(1-FMA)*CONg/0.93D0
   Gmulc=FMA*CONm/0.93D0
     IF CHIi<CHIm THEN Gmulc=0.D0
     IF CHIi<CHIc THEN Gcond=0 D0
   Gtot=Gtran+Gcond+Gvent+Gmulc+Ggrnd
   CHItot=Gtran*CHIv+Gcond*CHIc+Gvent*CHIo+Gmulc*CHIm+Ggrnd*CHIg
   CHIun=CHItot/Gtot
   tcon=Grh/Gtot
   CHIold=0.622D0*RHOA*T(7)/(1013.D0*100.D0)
   VPAi=(CHIun-(CHIun-CHIold)*EXP(-DTIM/tcon))*(1013.D0*100.D0)/RHOa/0.622D0
   TN(7)=VPAi
'*******************************************************************************
     FOR ICC = 1 TO NNN+1
      BBT = 1
       IF ABS(TN(ICC)) >= 1 THEN BBT = ABS(TN(ICC))
       IF ABS(DX(ICC)) >= SIGNUM * BBT THEN GOTO 8010
     NEXT ICC
     GOTO 8020
8010   IF ITER = MAXIT THEN PRINT "***NOT CONVERGE***":NOTC=NOTC+1
     NEXT ITER
8020 GOSUB SOILT:'SOIL TEMPERATURE CALCULATION
'************ SUBSTITUTE PREVIOUS WEATHER VAR. FOR NEW WEATHER VAR
    FOR IA = 1 TO NNN+1
     T(IA) = TN(IA)
    NEXT IA
'******** SUBSTITUTE PREVIOUS SOIL TEMP FOR NEW TEMP.
    FOR IA = 1 TO MM
     TSO(IA) = TSN(IA)
    NEXT IA
    RHAO=VPA/FN ES(TAIR)*100.D0
    RHAI=VPAI/FN ES(TMPA)*100.D0
    RHAIN=FN ES(WBT)/FN ES(DBT)*100.D0
  IF DUM1<1 AND DUM=1 THEN GOTO 11
  print#2, using"#####.#";TMPA;tc;TV;RHAI;VPAI/1000;TRAN*1000.*1000.
  PRINT#3,USING"#####.##";GSOL;SOLRAD;RNC;RnV;RNM;RNg;ghf;eg;hmg
 NEXT JF
 NEXT JDAY
2 END

'*************************************************************
' SUBROUTINE CALCULATES SOIL TEMPERATURE AT EACH DEPTH
'*************************************************************
SOILT:
    TSN(1) = Tg: 'SOIL SURFACE BOUNDARY
    AAA(1)=0.0D0:BBB(1)=1.0D0:CCC(1)=0.0D0:DDD(1)=TSN(1)
   FOR IA = 2 TO MM-1
    AAA(IA) = -0.5D0*(KCS(IA-1)*DELZ(IA-1)+KCS(IA)*DELZ(IA))/CCVS
    CCC(IA) = -0.5D0*(KCS(IA)*DELZ(IA)+KCS(IA+1)*DELZ(IA+1))/CCVS
    BBB(IA) = (DELZ(IA)^2.D0)/DTIM-AAA(IA)-CCC(IA)
    DDD(IA) = (DELZ(IA)^2.D0)/DTIM*TSO(IA)
   NEXT IA
    AAA(MM)=0.D0:' -(KCS(MM-1)*DELZ(MM-1)+KCS(MM)*DELZ(MM))/CCVS
    BBB(MM)=1.D0
    CCC(MM)=  0.D0
    DDD(MM) =TSN(MM):'(DELZ(MM)^2)/DTIM*TSO(MM)
'************** SOLUTION OF TRIDIAGONAL MATRIX
```

```
    FOR IA=2 TO MM
     RRATIO=AAA(IA)/BBB(IA-1)
     BBB(IA)=BBB(IA)-RRATIO*CCC(IA-1)
     AAA(IA)=RRATIO
    NEXT IA
    FOR IA=2 TO MM
     DDD(IA)=DDD(IA)-AAA(IA)*DDD(IA-1)
    NEXT IA
     TSN(MM)=DDD(MM)/BBB(MM)
    FOR IAA=2 TO MM
     IA=MM-IAA+1
     TSN(IA)=(DDD(IA)-CCC(IA)*TSN(IA+1))/BBB(IA)
  NEXT IAA
  RETURN
'************************************************************************
' SUBROUTINE CALCULATES transpiration AND LATENT HEAT FLUX
'************************************************************************
 TRANS:
    DL=0.06D0:       'characteristic length of tomato vegetation in m
    VPD=FN ES(TMPA)-VPAi
    LVPD=fn es(Tv)-VPAi
     if lvpd<=0. then lvpd=1.D0
     IF TMPA>35 OR TMPA<-10 OR Tv>35 OR Tv<-10 THEN GOTO 3
    RLSI=79.4D0*(1.D0+(exp(0.0234D0*(Gsol-76.6D0)))^-1.D0)*(1.D0+exp(0.079D0*(Lvpd/100.D0-2.832D0)))
    GOTO 4
3   RLSI=142.7D0+953.9D0*EXP(-0.0081D0*GSOL)
4    TTC=Tv:TTA=TMPA
     GOSUB RESIST
    RAHv=RAH

Tran=((delta*RNv/Gamma)+2.D0*LAI*rhoa*cp*VPD/gamma/RAHv)/(0.93D0+delta/gamma+rlsi/
RAHv)/lambda
    Hv=2.D0*LAI*RHOa*Cp*(Tv-TMPA)/RAHv
    Ev=LAMBDA*TRAN
    RETURN

'************************************************************************
'SUBROUTINE FOR CALCULATION OF INSTANTANEOUS GROSS PHOTOSYNTHESIS
'************************************************************************
PHOTO2:
    TTMPA=TMPA
    IF TTMPA>45 THEN TTMPA=45.D0
    IF TTMPA<0 THEN TTMPA=0
    Q10=2.D0
    Tleaf=Tv
    IF TLEAF>45 THEN TLEAF=45.D0
    IF TLEAF<0 THEN TLEAF=0.D0
    TDML2=51.25:'56.7
    TDMS=27.24
    DMGF=1.69
    RMRL=0.03:RMRS=0.015:RMRF=0.01
    RMAINTF=(RMRL*TDML2+RMRS*TDMS+RMRF*DNGF)*44.D0/30.D0
    RMAINTF=RMAINTF/(60.D0*60.D0*24.D0)*Q10^(0.1D0*TTMPA-2.5D0)
    CCO2L=CO2L:'350: 'CO2 CONCENTRATION IN ul/l
    LAI=PLAR2*PLM2
    SCP=0.2
    Q10=2.0
    RDTL20=1.1364: 'LEAF DARK RESPIRATION AT Tleaf=20C(umol CO2/m2 leaf/s)
    KDF=0.58: 'MEASURED EXTINCTION COEFFICIENT OF DIFFUSE PAR
    CD=42.7+1.68*(Tleaf-25)+0.012*(Tleaf-25)^2:  'CO2 COMPENSATION POINT(PPMV)
    EFF=0.084*(CCO2L-CD)/(CCO2L+2*CD): ' INITIAL LIGHT USE EFFICIENCY
                                        '(molCO2/mol photon absorbed)
    RDTL=RDTL20*Q10^(0.1*(Tleaf-20))
    rm=250:    'mesophyll resistance(s/m)
       if Tleaf<=5 or Tleaf>=40 then rm=10000000000000
       if Tleaf>5 and Tleaf<15  then rm=2500/(Tleaf-5)
       if Tleaf>25 and Tleaf<40 then rm=250/(1-(Tleaf-25)/15)
    PMM=(45*100/RM)/0.4
    IF Tleaf>=40 OR  TLEAF<=5 THEN PMM=0
    PNC=41.5843*(293/(273+Tleaf))*(CCO2L-CD)/(rm+1.36*rah+1.6*rlsi)
    PNMAX=PNC
    IF PNC>PMM THEN PNMAX=PMM
    PMAX=PNMAX+RDTL: 'max. gross photosynthetic rate(umolCO2/m2-leaf/s)
 ' CALCULATION OF REFLECTION FOR HORIZONTAL AND SPHERICAL LEAF ANGLE
 ' DISTRIBUTION
    SQV=SQR(1-SCP)
    REFH=(1-SQV)/(1+SQV)
```

```
    REFS=REFH*2/(1+2*COSZ)
' CALCULATION OF EXTINCTION COEFFICIENT FOR DIRECT RADIATION AND
' TOTAL DIRECT FLUX
    CLUSTF=KDF/(0.8*SQV)
    KBL = (0.5/COSZ)*CLUSTF
    KDRT= KBL*SQV
' SELECTION OF CANOPY DEPTH AND CANOPY ASSIMILATION IS SET TO ZERO
    GPF=0:  ' INSTANTANEOUS CANOPY ASSIMILATION RATE
    FOR IZ=1 TO 3
    LAIC=LAI*XGAUSS(IZ)
' ABSORBED DIFFUSE PAR, TOTAL DIRECT FLUX AND DIRECT COMPONENT OF
' DIRECT FLUX PER UNIT LEAF AREA
    VISDF=(1-REFH)*PPFDD*KDF*EXP(-KDF*LAIC)
    VIST =(1-REFS)*PPFDB*KDRT*EXP(-KDRT*LAIC)
    VISD =(1-SCP)*PPFDB*KBL*EXP(-KBL*LAIC)
' ABSORBED FLUX(umol/M2 LEAF/S) AND ASSIMILATION OF SHADED LEAVES
    VISSHD=VISDF+VIST-VISD
    IF PMAX>0 THEN
       GPFSH=PMAX*(1-EXP(-VISSHD*EFF/PMAX))
    ELSE
       GPFSH=0
    END IF
' DIRECT FLUX ABSORBED BY LEAVES PERPENDICULAR TO DIRECT PAR AND
' ASSIMILATION OF SUNLIT LEAF
    VISPP = (1-SCP)*PPFDB/COSZ
    GPFSUN=0
    FOR IP=1 TO 3
    VISSUN=VISSHD+VISPP*XGAUSS(IP)
    IF PMAX>0 THEN
       GPFS=PMAX*(1-EXP(-VISSUN*EFF/PMAX))
    ELSE
       GPFS=0
    END IF
    GPFSUN=GPFSUN+GPFS*WGAUSS(IP)
    NEXT IP
' FRACTION OF SUNLIT LEAF AREA(FSLLA) AND LOCAL ASSIMILATION RATE(GPL)
    FSLLA=CLUSTF*EXP(-KBL*LAIC)
    GPL=FSLLA*GPFSUN+(1-FSLLA)*GPFSH
' INTEGRATION OF LOCAL ASSIMILATION TO CANOPY ASSIMILATION(GPF)
    GPF=GPF+GPL*WGAUSS(IZ)
    NEXT IZ
     GPFM=GPF
'    GPF=GPF*LAI/PLM2
' CONVERSION FROM umol CO2/plant/s TO umol CO2/plant/day
    GPF=GPF*86400
' CONVERSION FROM umol CO2  TO g CH2O
    GPF=GPF*30/1000000
    RETURN

'*************************************************************************
'SUBROUTRINE FOR SOLUTION OF SIMULTANEOUS EQUATIONS
'*************************************************************************
MATRIX:
'********** FIND LARGEST VALUE IN KO-TH COLUMN
  FOR KO = 1 TO NNN - 1
   IR(KO) = KO: PP = ABS(MA(KO, KO))
     FOR JX = KO TO NNN
      IF PP > ABS(MA(JX, KO)) THEN GOTO 8000
      IR(KO) = JX: PP = ABS(MA(JX, KO))
8000 NEXT JX
'********* SWAP KO-TH EQN AND Pp-TH EQN
     FOR JX = KO TO NNN
      TEMQ = MA(KO, JX): MA(KO, JX) = MA(IR(KO), JX): MA(IR(KO), JX) = TEMQ
     NEXT JX
'********** L-U DECOMPOSITION
     FOR IZ = KO + 1 TO NNN
      Z = MA(IZ, KO) / MA(KO, KO): MA(IZ, KO) = Z
     FOR JX = KO + 1 TO NNN
       MA(IZ, JX) = MA(IZ, JX) - Z * MA(KO, JX)
       NEXT JX
     NEXT IZ
    NEXT KO
'********** SWAP IZ-TH ELEMENT AND P-TH ELEMENT
     FOR IZ = 1 TO NNN - 1
      TEMQ = VB(IZ): VB(IZ) = VB(IR(IZ)): VB(IR(IZ)) = TEMQ
      FOR JX = IZ + 1 TO NNN: VB(JX) = VB(JX) - MA(JX, IZ) * VB(IZ): NEXT JX
     NEXT IZ
```

```
'********** BACK SUBSTITUTION
    DX(NNN) = VB(NNN) / MA(NNN, NNN)
    FOR IZ = NNN - 1 TO 1 STEP -1
      DX(IZ) = VB(IZ)
      FOR JX = IZ + 1 TO NNN
       DX(IZ) = DX(IZ) - MA(IZ, JX) * DX(JX)
      NEXT JX
       DX(IZ) = DX(IZ) / MA(IZ, IZ)
      NEXT IZ
   RETURN

'*************************************************************************
'SUBROUTINE FOR CALCULATION OF BOUNDARY LAYER RESISTANCES
'*************************************************************************
RESIST:
      xxp=(TTC+TTA)/2.D0
      delta= fn esp(xxp)
      Re=U*Dl/nhu:  'Reynold's number
      Nuf=0.66D0*(Re^0.5D0)*(0.7D0^0.33D0): 'Nusselt number for forced convection
      Gr=aex*9.8D0*dl^3.D0*abs((TTC-TTA))/nhu/nhu:'Grashof number
      Nust=0.26D0*(Gr*0.7D0)^0.25D0
      Nusb=0.50D0*Gr^0.25D0
      Nusm=((Nust+Nusb)/2+Nuf)' 'mixed convection Nusselt number
      if TTC-TTA<=0 then Nusm=((Nuf^3.55D0+Nust^3.55D0)^0.28D0+
                               (Nuf^3.55D0+Nusb^3.55D0)^0.28D0)/2.D0
      Rah=Dl/(kapa*Nusm)
      return

'*************************************************************************
'Subroutine for calculation of sun-earth astronomical parameters and transmission
'of solar radiation through greenhouse cover layer
'*************************************************************************
SOLAR'
        PI=3.141592D0:PPI=PI/180.D0
        LONG=LON*PPI:LATT=LAT*PPI
'rem calculation of SUN-EARTH astronomical parameters
        SDEC=23.45D0*SIN(2*PI*(date+284.D0)/365.D0)*PPI:'SOLAR DECLINATION ANGLE
        TTTT=2.D0*PI*(date-1.D0)/365.D0
EQOT=(0.000075+0.001868*COS(TTTT)-0.032077*SIN(TTTT)-0.014615*COS(2*TTTT)-0.04089*
SIN(2*TTTT))*229.18/60:'IN HOUR
          SOLT=tfast+4*(LON-135.D0)/60.D0+EQOT
          HANG=(SOLT-12.D0)*15.D0*PPI
       COSZ=SIN(SDEC)*SIN(LATT)+COS(SDEC)*COS(LATT)*COS(HANG)
       TALP=COSZ/SQR(1-COSZ*COSZ):'TANGENT OF SOLAR ELEVATION
       IF TALP<=0.001 THEN TALP=0.001
     AZCO=(COSZ*SIN(LATT)-SIN(SDEC))/(SQR(1-COSZ*COSZ)*COS(LATT))
      'AZIM=AZIMUTH ANGLE OF SUN IN DEGREE
     IF   AZCO>=0      AND      HANG>=0   THEN   AZIM=(-1*ATN(AZCO/SQR(1-AZCO*AZCO
                                          +0.00001))+PI/2)/PPI
     IF   AZCO>=0   AND   HANG<=0   THEN   AZIM=-1*(-1*ATN(AZCO/SQR(1-AZCO*AZCO
                                      +0.00001))+PI/2)/PPI
     IF    AZCO<0    AND    HANG>=0    THEN    AZIM=(-1*ATN(AZCO/SQR(1-AZCO*AZCO+
                                          0.00001))+PI/2)/PPI
     IF    AZCO<0    AND    HANG<0    THEN    AZIM=-(-1*ATN(AZCO/SQR(1-AZCO*AZCO+
                                          0.00001))+PI/2)/PPI
   REM CALCULATION OF INCIDENCE ANGLE OF SOLAR RAD. ON WALL SURFACE
        PCSL=0.D0:PCSH=0.D0:'ACT=0.D0
        CFCS=0.D0:CFGC=0.D0:AINC=0.D0:ATR=0.D0
        TRCB=0.D0:RECB=0.D0:ABCB=0.D0: TRCGB=0.D0:REXT=0.D0
        TRCD=0.D0:RECD=0.D0:ABCD=0.D0:TRCGB=0.D0
         GSOL=0.D0:GSOLD=0.D0
         GSOLB=0.D0:GSOLD=0.D0:PARB=0.D0:PARD=0.D0:PAR=0.D0:
         PPFDB=0.D0:PPFDD=0.D0:PPFD=0.D0
        IF COSZ<0.D0 THEN GOTO 4900
        indi=1
        GOSUB CALAB
        indi=0
        TRCD=TR: RECD=RE  : ABCD=AB
       FOR IZ=1 TO 6
       COSI=COS(WSLO(IZ)*PPI)*COSZ+SIN(WSLO(IZ)*PPI)*SQR(1-COSZ*COSZ)*
   COS((AZIM-WAZI(IZ))*PPI)
       WINC=-1*ATN(COSI/SQR(1-COSI*COSI))+PI/2:'SOLAR    INCIDENCE    ANGLE    ON
                                          'WALL SURFACE IN RADIAN
       IF COSI>0 THEN PCSL=PCSL+ AC(IZ)
       IF COSI<=0 THEN PCSH=PCSH+AC(IZ)
       CFCS=CFCS+AC(IZ)*(1+COS(WSLO(IZ)*PPI))/2
       CFGC=CFGC+AC(IZ)*(1-COS(WSLO(IZ)*PPI))/2
```

```
      IF COSI>0 THEN AINC=AINC+AC(IZ)*WINC
   REM CALCULATION  OF TRANSMISSIVITY,REFLECTIVITY, ABSORPTIVITY OF        REM
COVERING FILM
        IF COSI<0 THEN GOTO 5035
         IF DOU=1 THEN
           FL=FFL
           IF IZ=1 OR IZ=4 OR IZ=5 OR IZ=6 THEN FL=2.D0*FFL
         ELSE
           FL=FFL
         END IF
         GOSUB CALAB
         GOTO 5040
5035     TR=0:RE=1:AB=0
5040     rem calculation of average tr, re and ab at the wall surface
         TRCB=TRCB+AC(IZ)*TR/ACT:'AVERAGE     TRANSMITTANCE  AT     THE  WALL
                          ' SURFACE
         RECB=RECB+AC(IZ)*RE/ACT:'  "     REFLECTANCE          "
         ABCB=ABCB+AC(IZ)*AB/ACT:'  "     ABSORPTANCE          "
rem calculation of average tr of greenhouse cover on the vegetation surface
         PAZI=ABS(AZIM*PPI)
         IF ABS(AZIM)>90 THEN PAZI=ABS(AZIM*PPI)-PI/2
           GX=WZL*COS(PAZI)/TALP
           GY=WZL*SIN(PAZI)/TALP
           GGX=(WZL+GZL)*COS(PAZI)/TALP
           GGY=(WZL+GZL)*SIN(PAZI)/TALP
       IF EWL<=GY THEN GOSUB 5000
       IF GY<EWL AND GGY>EWL/2 THEN GOSUB 5100
       IF GY<EWL AND GGY<= EWL/2 THEN GOSUB 5110
       IF ABS(AZIM)>90 THEN GOSUB 5120
         TRCGB=TRCGB+SSI*TR/AG
       IF AZIM>=0 AND iZ=2 THEN TRCGBI=TR
       IF AZIM<0 AND IZ=3 THEN TRCGBI=TR
    NEXT IZ
       AINC=AINC/PCSL:'AVERAGE INCIDENCE   ANGLE  OF   SOLAR  BEAM   RAD.  ON
            'WALL SURFACE
       PCSL=PCSL/ACT:'PROPORTION OF SUNLIT WALL SURFACE AREA
       PCSH=PCSH/ACT:'PROPORTION OF SHADED WALL SURFACE AREA
       CFCS=CFCS/ACT:'AVERAGE   SHAFE  FACTOR   WHICH   WALL  SURFACE   SEE
            'THE SKY
       CFGC=CFGC/ACT:'AVERAGE  SHAFE  FACTOR    WHICH  OUTSIDE  GROUND  SEE
            'THE WALL SURFACE
       ACOSI=COS(AINC)
'SEPERATION OF DIFFUSE AND BEAM RADIATION COMPONENT
       SOLC=1367.D0*(1+0.033D0*COS(2.D0*PI*DATE/365.D0)):'SOLAR CONSTANT
                                                         'IN(W/M2)
       REXT=SOLC*COSZ:'EXTRATERESTRIAL SOLAR RADIATION IN W/M2
       ATR=RT/REXT:'ATMOSPHERIC TRANSMISSION RATE

'*******************************************************************************
DE JONG(1980)

'*******************************************************************************
 RRRR=0.847-1.61*COSZ+1.04*COSZ*COSZ
        KKKK=(1.47-RRRR)/1.66
       IF ATR>=0 AND ATR<=0.22 THEN MT=1
       IF ATR>0.22 AND ATR<=0.35 THEN MT=1.D0-6.4D0*(ATR-0.22D0)^2.D0
       IF ATR>0.35 AND ATR<=KKKK THEN MT=1.47D0-1.66D0*ATR
       IF ATR>KKKK THEN MT=RRRR

'*******************************************************************************
RT=SOLRAD
       RD=SOLRAD*MT:'DIFFUSE RADIATION IN W/M2:'DIFFUSE RADIATION
       RB=SOLRAD-RD:'BEAM RADIATION IN W/M2:'BEAM RADIATION
       GSOLB=RB*TRCGB*FAFN:'BEAM RADIATION(W/M2) IN GREENHOUSE
       GSOLD=RD*TRCD*FAFN:''DIFFUSE RADIATION(W/M2) IN GREENHOUSE
       GSOL=GSOLD+GSOLB:'GLOBAL RADIATION(W/M2) IN GREENHOUSE
       PARB=GSOLB*0.47D0:'BEAM PAR(W/M2) IN GREENHOUSE
       PARD=GSOLD*0.47D0:'DIFFUSE PAR(W/M2) IN GREENHOUSE
       PAR=PARB+PARD:'GLOBAL PAR(W/M2) IN GREENHOUSE
       PPFDB=PARB*4.57D0:'BEAM PPFD(UMOL/M2) IN GREENHOUSE
       PPFDD=PARD*4.57D0:'DIFFUSE PPFD(UMOL/M2) IN GREENHOUSE
       PPFD=PPFDB+PPFDD:'GLOBAL PPFD(UMOL/M2) IN GREENHOUSE
       IF SCRI=1 THEN
       GSOLD=(GSOLB+GSOLD)*TRs:GSOLB=0.0D0
       GSOL=GSOLD
       PARB=0.0D0
       PARD=GSOLD*0.47D0
```

```
        PAR=PARB+PARD
        PPFDB=0.D0
        PPFDD=PARD*4.57D0
        PPFD=PPFDB+PPFDD
        END IF
        IF SOLRAD=0 THEN GSOL=0
4900    RETURN
5000    REM CALCULATION OF SHADED AREA OF WALL AZIM>=0 AND GY>=EWL
          SSI=0.0:W1=1:W2=5:'SHADED AREA OF WALL ON GREENHOUSE BED
          IF AZIM<0 THEN W1=4
          IF IZ=W1 THEN SSI=EWL*NSL-EWL*EWL/TAN(PAZI)/2
          IF IZ=5 THEN SSI=EWL*EWL/TAN(PAZI)/2
          RETURN
5100  REM  CALCULATION OF SHADED AREA OF WALL AZIM>=0  AND GY<EWL<=GGY
          SSI=0:WW1=1:WW2=2
          IF AZIM<0 THEN
             WW1=4
             WW2=3
          END IF
           AANG=ATN((GGY+EWL/2-GY)/(GGX-GX))
           BANG=PI/2-AANG
            IF IZ=WW1 THEN SSI=(2*NSL-GX)*GY/2
            IF IZ=WW2 THEN SSI=(2*NSL-2*GX-(EWL-GY)*TAN(BANG))*(EWL-GY)/2
            IF IZ=5 THEN SSI=(2*EWL-GY)*GX/2+(EWL-GY)*(EWL-GY)*TAN(BANG)/2
          RETURN
5110      SSI=0 :WWW1=1:WWW2=2:WWW3=3
          IF AZIM<0 THEN
             WWW1=4
             WWW2=3
             WWW3=2
          END IF
            if iZ=WWW1 then ssi=(2*NSL-GX)*GY/2
            IF IZ=WWW2 THEN SSI=(2*NSL-GX-GGX)*(EWL/2+GGY-GY)/2
            IF IZ=WWW3 THEN SSI=(NSL-GGX)*(EWL/2-GGY)
            IF IZ=5 THEN SSI=(2*EWL-GY)*GX/2+(3*EWL/2-GY-GGY)*(GGX-GX)/2
          RETURN
5120      SSI=0: W1=1:W2=2
          GX=ABS(GX):GY=ABS(GY):GGX=ABS(GGX):GGX=ABS(GGX)
          IF AZIM<0 THEN W1=3:W2=4
           AANG=ATN((GGY+EWL/2-GY)/(GGX-GX))
           BANG=PI/2-AANG
            IF IZ=W1 THEN SSI=(2*NSL-GX)*GY/2
            IF IZ=W2 THEN SSI=(2*NSL-2*GX-(EWL-GY)*TAN(BANG))*(EWL-GY)/2
            IF IZ=6 THEN SSI=(2*EWL-GY)*GX/2+(EWL-GY)*(EWL-GY)*TAN(BANG)/2
          RETURN
'*************************************************************************
'SUBROUTINE CALCULATE NET RADIATION
'*************************************************************************
NETRAD:

      RLc=0.0D0:RSc=0.0D0:Rls=0.0D0:Rss=0.0D0
      Rlv=0.D0:Rsv=0 D0:Rnm=0.D0:RSm=0.D0
      Rlg=0.D0:RSg=0.D0:RTco=0.D0:RTci=0.D0
      RTso=0.D0:RTsi=0.D0
'************** SHORTWAVE RADIATION ************************************
   IF SCRI=0 THEN REs=0.D0:TRs=1.D0:ABSS=0.D0
     KDF=0.58D0:'KDF=0.63
     SCP=0.2D0
     CLF=KDF/0.8D0/SQR(1.D0-SCP)
     KDRBL=ABS(0.5D0*CLF/COSZ)
     KDRT=KDRBL*SQR(1.D0-SCP)
     REvh=(1.D0-SQR(1.D0-SCP))/(1.D0+SQR(1 D0-SCP))
     REvs=ABS(2.D0/(1.D0+1.6D0*COSZ))*REvh
'* IF SCREEN IS OPENED, THEN REv=REvh
   IF SCRI=1 THEN REv=REvh
     TRvD=EXP(-KDF*LAI)
     TRVB=EXP(-KDRT*LAI)
     ABvD=(1-REvH)*(1.D0-TRvD)
     ABvB= (1-REV)*(1.D0-TRvB)
     ABV=ABVD*MT+(1.D0-MT)*ABVB
     TRv=TRvd*MT+TRVB*(1.D0-MT)
     REv=REvH*MT+REvs*(1.D0-MT)
     TRlv=EXP(-0.8D0*SQR(1.D0-0.1D0)*LAI)
     EPv=0.95D0
     RElv=(1.D0-SQR(1.D0-0.1D0))/(1.D0+SQR(1.D0-0.1D0))
      BBX=0.9D0/1.5D0:AAX=PLTH/1.5D0
      DENN=1.D0-0.8D0*TRlv^(1.18D0*BBX)-0.2D0*TRlv^(2.75D0*BBX)
```

```
      NONN=1.D0-0.2D0*TRlv^1.18D0-0.2D0*TRlv^2.75D0
      FFWW=(1-0.07D0*(1-BBX)*(4.7D0-AAX))*DENN/NONN
      ABv=ABV*FFWW
      TRv=1.D0-FFWW+FFWW*TRv
      TRlv=1.D0-FFWW+FFWW*TRLv
      REv=REv*FFWW
      RElv=RElv*FFWW
    FOR GHX=1 TO 2
      IF GHX=1 THEN GOTO 7
      TRLm=1.D0:RElm=0.D0:EPlm=0.D0
      TRm=1.D0: RREm=0.D0: AMm=0.D0
7     '************ COVER LAYER
      RECC=RES+TRS^2.D0*REv+(TRs*TRv)^2.D0*RREm+(TRs*TRv*TRm)^2 D0*REg
      RSC=(RB+RD)*ABCd*(1:D0+TRc*RECC/(1.D0-RECd*RECC))
     '************ SCREEN LAYER
      RTso=ABss*(TRcb*RB*(ACOSI/COSZ)*PCSL+TRcd*RD)*(1+REcd*REs)
      RTsi=ABss*TRcb*TRs*RB*(ACOSI/COSZ)*PCSL
      RTsi=RTsi+ABss*(TRcgb*TRs*RB+TRcd*TRs*RD)*(REv+TRv^2.D0*
           (RREm+TRm^2.D0*REg))
      RSs=RTso+RTsi
     '************ VEGETATION LAYER
      RESV=REv+RREm*TRv^2 D0+REg*(TRv*TRm)^2.D0
      RESC=REs+RECd*TRs^2.D0
      RSv= ABv*GSOL/(1.D0-RESV*RESC)
     '*********** MULCHING LAYER
      REMM1=REv+REs*TRv^2.D0+REcd*(TRv*TRs)^2.D0
      REMM2=RREm+REg*TRm^2.D0
      RSm=GSOL*TRs*TRv*ABm*(1.D0/(1 D0-REMM1*REMM2)+REg*TRm/
           (1.D0-RREm*REg))
     '*********** SOIL SURFACE LAYER
      REGG1=RREm+REv*TRm^2.D0+(TRv*TRm)^2.D0*REs+REcd*(TRv*TRs*TRm)^2.D0
      RSg=GSOL*TRv*TRm*(1.D0-REg)/(1.D0-REg*REGG1)
     '*********************************************
*** IF SCREEN WERE NOT COVERED, EPs=0:TRls=1:REls=0
    IF SCRI=0 THEN
       EPs=0.D0
       TRls=1.D0
       REls=0.D0
    END IF
    QLa=EPa*SIG*(Tair+273.16D0)^4.D0
    QLc=EPc*SIG*(Tc+273.16D0)^4.D0:' QLc3=4*QLc/(Tc+273.16)
    QLs=EPs*SIG*(Ts+273.16D0)^4.D0:' QLs3=4*QLs/(Ts+273.16)
    QLv=EPv*SIG*(Tv+273.16D0)^4.D0:' QLv3=4*QLv/(Tv+273.16)
    QLm=EPm*SIG*(Tm+273.16D0)^4.D0:' QLm3=4*QLm/(Tm+273.16)
    QLg=EPg*SIG*(Tg+273.16D0)^4.D0:' QLg3=4*QLg/(Tg+273.16)
   '************** COVER LAYER
    rlvs=REls+TRls^2.D0*RElv*(1-TRlv)+RElm*TRls^2.D0*TRlv^2.D0+REIg*
         (TRls*TRlv*TRlm)^2.D0
    Rlc11=EPc*(QLs+(1.D0-TRlv)*TRls*QLv+TRls*TRlv*QLm+TRls*TRlv*TRlm*QLg)/
          (1.D0-REIc*RLVS)
    RLc12=EPc*RLVS*QLc/(1.D0-REIc*RLVS)
    RLc13=EPc*TRlc*RLVS*QLa/(1.D0-REIc*RLVS)
    RLc=RLC11+RLc12+RLc13+EPc*QLa-2.D0*QLc
   '************** SCREEN LAYER
    RLVS=RElv*(1.D0-TRlv)+RElm*TRlv^2.D0+REIg*(TRlv*TRlm)^2.D0
    RLs11=((1.D0-TRlv)*QLv+TRlv*QLm+TRlv*TRlm*QLg)*EPs/(1.D0-RLVS*REls)
    RLs12=QLs*RLVS*EPs/(1.D0-RLVS*REls)
    RLs13=(QLa*TRlc+QLc)*EPs
    RLs14=(QLa*TRlc+QLc)*RLVS*TRls*EPs/(1.D0-RLVS*REls)
    RLs15=(QLa*TRlc+QLc)*EPs/(1.D0-REIc*REls)
    RLs=RLs11+RLs12+RLs13+RLs14+RLs15-2.D0*QLs
   '************** VEGETATION LAYER
      RELV1=REls+REIc*TRls^2.D0
      RELV2=RELV+TRlv^2.D0*REIm+REIg*(TRlv*TRlm)^2.D0
    RLv31=(QLa*TRlc*TRls+QLc*TRls+Qls)*(1.D0-TRlv)/(1.D0-RELV1*RELV2)
      RELV3=RElv+TRlv^2.D0*REls+REIc*(TRLv*TRls)^2.D0
      RELV4=REIm+TRlm^2.D0*REIg
    RLV32=(QLm+QLg*TRm)*(1.D0-TRlv)/(1.D0-REIv3*REIv4)
    RLV33=QLv*(1.D0-TRlv)^2.D0*(1.D0-REIv)*RELV1/(1.D0-RELV1*RELV2)
    RLV34=QLv*(1.D0-TRlv)^2.D0*(1.D0-REIv)*RELV4/(1.D0-RELV3*RELV4)
    RLV35=2.D0*QLv*(1.D0-TRlv)
    RLv=RLV31+RLV32+RLV33+RLV34-RLV35
    '************ MULCH LAYER
    RELM1=RElv+REls*TRlv^2.D0+REIc*(TRlv*TRls)^2.D0
    RLm111=(QLa*TRlc*TRls*TRlv+QLc*TRls*TRlv+QLs*TRlv+QLv*(1.D0-TRlv))
    RLm11=RLm111*EPm*(1.D0+TRlm*REIg/(1.D0-REIg*REIm))
    RLm12=QLm*(2.D0-EPm*REIg/(1.D0-REIm*REIg))
```

```
      RLm13=QLg*EPm/(1.D0-REIm*REIm*EPg)
      RLM131=QLg*TRIm*RELM1*REIm/(1.D0-RELM1*REIm)
      RLm14=RLm111*REIm*RELM1*EPm/(1.D0-RELM1*RELm)
      RLm15=QLm*RELM1*EPm/(1.D0-RELM1*RELM)
      RLm=RLm11+RLm13+RLM131+RLm14+RLm15-RLm12
     '********** SOIL LAYER
      RELG1=REIm+REIv*(1.D0-TRIv)*TRIm^2.D0+REIs*(TRIm*TRIv)^2.D0+REIc*
            (TRIm*TRIv*TRIs)^2.D0
      RLg11=EPg*(QLa*TRIc*TRIs*TRIv*TRIm+QLc*TRIs*TRIv*TRIm)/(1.D0-REIG1*REIg)
      RLg12=EPg*(QLs*TRIv*TRIm+(1.D0-TRIv)*QLv*TRIm+QLm)/(1.D0-RELG1*REIg)
      RLg=RLg11+RLg12-(1.D0-EPg*REIG1/(1.D0-REIg1*REIg))*QLg
     '************ NET RADIATION ***************************************
    IF GHX=2 THEN GOTO 8
       RNc1=(RLc+RSc)*FMA: ' COVER
       RNs1=(RLs+RSs)*FMA: ' SCREEN
       RNv1=(RLv+RSv)*FMA:'FWW:'*0.83*(1-0.7*EXP(-0.24*LAI)): ' VEGETATION
       RNm1=(RLm+RSm)*FMA: ' MULCH
       RNg1=(RLg+RSg)*FMA: ' SOIL SURFACE
       GOTO 9
8      RNc2=(RLc+RSc)*(1.D0-FMA): ' COVER
       RNs2=(RLs+RSs)*(1.D0-FMA): ' SCREEN
       RNv2=(RLv+RSv)*(1.D0-FMA):'FWW:'*0.83*(1-0.7*EXP(-0.24*LAI)): ' VEGETATION
       RNm2=(RLm+RSm)*(1.D0-FMA): ' MULCH
       RNg2=(RLg+RSg)*(1.D0-FMA): ' SOIL SURFACE
9      NEXT GHX
       RNC=RNC1+RNC2
       RNS=RNS1+RNS2
       RNV=RNV1+RNV2
       RNM=RNM1+RNM2
       RNG=RNG1+RNG2
      RETURN
'*************************************************************************
'SUBROUTINE CALCULATES ABSORPTION, TRANSMISSION AND REFLECTION OF 'COVER
'*************************************************************************
CALAB:
          SWIN=SIN(WINC)
          CWIN=COS(WINC)
           IF INDI=1 THEN
           SWIN=SIN(64*PPI)
           CWIN=COS(64*PPI)
           END IF
          IF SWIN=0.0 THEN GOTO 5010
          SRFR=SWIN/FI
          CRFR=SQR(1-SRFR*SRFR)
          SMWR=SWIN*CRFR-CWIN*SRFR
          SPWR=SWIN*CRFR+CWIN*SRFR
          CMWR=CWIN*CRFR-SWIN*SRFR
          CPWR=CWIN*CRFR+SWIN*SRFR
          TANP=SPWR/CMWR
          TANM=SMWR/CPWR
          SMW2=SMWR*SMWR
          SPW2=SPWR*SPWR
          TANP2=TANP*TANP
          TANM2=TANM*TANM
          FR=0.5*(SMW2/SPW2+TANM2/TANP2)
          GOTO 5020
5010      FR=(1-1/FI)*(1-1/FI)/((1+1/FI)*(1+1/FI))
          CRFR=1
5020      AA=EXP(-KCC*FL/CRFR)
          TR=(1-FR)*(1-FR)*AA/(1-(FR*AA)^2)
          RE=FR+FR*(1-FR)^2*AA^2/(1-(FR*AA)^2)
          AB=1-FR-(1-FR)*(1-FR)*AA/(1-FR*AA)
          RETURN
'*************************************************************
' SUBROUTINE FOR INPUT OF INITIAL PARAMETERS
'*************************************************************
INPARM:
    'LAI=PLAR2*PLM2
    CO2O=350:'PPMCO2 IN OUTSIDE AIR
    cp=1004.D0:       'in J/(kg.K), specific heat of air at constant pressure
    SIG = 5.67D-8:' / 100000000.: 'STEFAN-BOLTZMAN CONST.(W/M2/K)
    gamma=66.2D0:'64.6*(1+0.000946*xxp)
    rhoa=1.204D0:'1.204*293.2/(273.2+xxp)
    nhu=15.5D-6:'15.1*((xxp+273.2)/293.2)^1.75/1000000: 'kinematic viscosity(m2/s)
    kapa=22.2D-6::'21.5*((xxp+273.2)/293.2)^1.75/1000000:'thermal diffusivity(m2/s)
    aex=1.D0/273.16D0:'1/(xxp+273.16):                    'thermal expansion of air
    lambda=(2500.25D0-2.365D0*20.D0)*1000.D0: 'J/Kg, latent heat of vaporization
```

```
'********SOIL PHYSICAL PROPERTIES
   BULKD = 1.1D0: 'BULK DENSITY OF SOIL(G/CM3)
   PARTD = 2.65D0: 'PARTICLE DENSITY OF SOIL(G/CM3)
   CLAYC = .2D0: 'CLAY FRACTION
'********** SURFACE AERODYNAMIC PARAMETERS
   VK = .41D0: ' VON KARMAN CONSTANT
   DZERO = .14D0: ' ZEROPLANE DISPLACEMENT HEIGHT(M)
   ZM0 = .005D0:    ' ROUGHNESS LENGTH FOR MOMENTUM(M)
   ZH0 = ZM0 * .2D0: 'ROUGHNESS LENGTH FOR HEAT(M)
   ZWIN = 3.D0:    'HEIGHT OF WIND MEASUREMENT(M)
'************
   MSDIS =0.65D-3:' .65/1000: 'DISTANCE BETWEEN MULCH AND SOIL SURFACE
'******** PHOTOMETRICAL PROPERTIES OF COVER
   TRc = .83D0:'0.888 ' SOLAR RADIATION TRANSMITTANCE OF SCREEN
   REc = .12D0:'0.079 'SOLAR RADIATION REFLECTANCE OF SCREEN
   ABc = .05D0:'0.033 'SOLAR RADIATION ABSORPTANCE OF SCREEN
   TRlc =.791D0: 'IR RADIATION TRANSMITTANCE OF SCREEN
   RElc =.146D0:' .16: 'IR RADIATION REFLECTANCE OF SCREEN
   EPc=0.063D0:' .09: 'EMMISSIVITY OF IR RADIATION OF SCREEN
'******** PHOTOMETRICAL PROPERTIES OF SCREEN
   TRs =0.880:' .84:'0.880' SOLAR RADIATION TRANSMITTANCE OF SCREEN
   REs =0.084' .11:'0.084 'SOLAR RADIATION REFLECTANCE OF SCREEN
   ABss =0.036:' .05.'0.036 'SOLAR RADIATION ABSORPTANCE OF SCREEN
   TRls = .78: 'IR RADIATION TRANSMITTANCE OF SCREEN
   REls = .11: 'IR RADIATION REFLECTANCE OF SCREEN
   EPs = .11: 'EMMISSIVITY OF IR RADIATION OF SCREEN
'******** PHOTOMETRICAL PROPERTIES OF MULCH
   TRm =0.88)D0:'0.75:'8:' .84: ' SOLAR RADIATION TRANSMITTANCE OF MULCH
   RREm =0.084D0:'0.20:'0.15:' .11: 'SOLAR RADIATION REFLECTANCE OF MULCH
   ABm = 0.036D0:'.05: 'SOLAR RADIATION ABSORPTANCE OF MULCH
   TRlm =0.85D0:' .80: 'IR RADIATION TRANSMITTANCE OF MULCH
   RElm =0.12D0:'0.15:'0.05:'0.1:'0.04:'0.16: 'IR RADIATION REFLECTANCE OF MULCH
   EPm =0.03D0:' .04: 'EMMISSIVITY OF IR RADIATION OF MULCH
'*************** SOIL OPTICAL PROPERTIES
   REg = .25D0: 'SOIL ALBEDO
     IF WAT > .1 AND WAT < .25 THEN REg = .35D0 - WAT
     IF WAT >= .25 THEN REg = .1D0
   ABg=1.D0-REg
   EPg = .9D0 + 18D0 * WAT: '    SOIL SURFACE EMMISSIVITY
   RElg=1.D0-EPg
'******** SOIL STEP SIZE(DELZ)
MM=21
    FOR YY=1 TO MM
    DELZ(YY)=0.05D0
    NEXT YY
'******** VOLUMETRIC HEAT CAPACITY OF SOIL(Ccvs, J/M3/K)
    Ccvs = 2390000.D0 * BULKD / PARTD + 4180000.D0 * WAT
'******** CALCULATION OF SOIL THERMAL CONDUCTANCE(Kcs,W/M/K)
     CC1 = .65D0 - .78D0 * BULKD + .6D0 * BULKD ^ 2.D0
     CC2 = 1.06D0 * BULKD * WAT
     CC3 = 1.D0 + 2.6D0 / SQR(CLAYC)
     CC4 = .03D0 + .1D0 * BULKD * BULKD
   FOR YY = 1 TO MM
     kcs(YY)=(CC1 + CC2 * WAT - (CC1 - CC4) * EXP(-(CC3 * WAT) ^
             4.D0))/DELZ(YY)
   NEXT YY
'*********BOTTOM BOUNDARY(SOIL TEMP AT 1M DEEP SOIL(TSB),LEE(1989)********
'********* ANNUAL CYCLE OF SOIL TEMP AT SUWON*****************************
     TSB = 13.5D0
    FOR YY = 1 TO 4
     SDD = 0.0D0: CDD = -.09D0
      IF YY = 1 THEN SDD = -6.95D0: CDD =-6.80D0
      IF YY = 2 THEN SDD = -.22D0: CDD = -0.35D0
      IF YY = 3 THEN SDD = .01D0: CDD = 0.25D0
     TSB = TSB + SIN(2.D0 * 3.1416D0 * YY * DATE / 365.D0) * SDD
     TSB=  TSB + COS(2.D0 * 3.1416D0 * YY * DATE / 365.D0) * CDD
   NEXT YY
'**********INITIALIZE GROUND LAYER ******
   FOR IA = 1 TO MM
    TSO(IA) = TSB+10.D0
   NEXT IA
    TSn(MM) = TSB
    TSN(MM) = TSO(MM)
    TSn(1) =TsB
'************ GREENHOUSE DIMENSION ******
    ACT=(NSL+EWL)*2*WZL+SQR((EWL/2)^2+GZL^2)*NSL*2+EWL*GZL:'TOTAL                COVER
'AREA
```

```
   GRv=Ag*WZL+AG*GZL/2:'GREENHOUSE VOLUME
   GRh=GRv/Ag:'EFFECTIVE HEIGHT OF GREENHOUSE
   Af=GRv/EWL
   FMA=0.935D0:'0.915:'(NSL-0)*0.9*3/AG:'PROPORTION OF MULCHED AREA
********** SCREEN DIMENSION **********
  'GAP(GAPW) BETWEEN SCREEN AND WALL COVER
  'GAP(GAPR) BETWEEN SCREEN AND WALL COVER
   GAPW=0.1D0:'IN M
   GAPR=1.0D0:'IN M
   Ags= (EWL-GAPW*2)*(NSL-GAPw*2)
   ACTs=AGs+(NSL+EWL-GAPW*4)*2*WZL
   GRvs=Ags*WZL
   GRhs=GRvs/Ags
   Afs=GRvs/(EWL-2*GAPW)
********* WINDOW AREA OPENED ********************************
IF WINDOW IS CLOSED, THEN WOH=0.1
    WOH=0.005D0
    Aw=(NSL-4.D0)*WOH*2.D0:'WOH=HEIGHT OF SIDE WINDOW OPENING
   WOS=0.16D0
   Aws=(NSL-2.D0*GAPW)*WOS
    RETURN
```

APPENDIX 3. LIST OF GREENHOUSE CLIMATE CONTROL SYSTEM PROGRAM

```
/* ************************************** *
 *       온실제어 프로그램 version 0.0
 * ************************************** */

#include <alloc.h>
#include <string.h>
#include <ctype.h>
#include "hanlib.h"
#include "extkey.h"
#include "hanin.h"
#include "ascii.h"
#include "mystdio.h"
#include "mymenu.h"

int (*menufunc)(void);

int check_mou_is_in_sub(popupmenu_t *m)
{
 if(m)
        if(mou_ax > m->menuwin.lx+1       && mou_ax < m->menuwin.rx-1) {
         if(mou_ay > m->menuwin.ly && mou_ay < m->menuwin.ry
                        && m->menu[mou_ay - m->menuwin.ly - 1].active) {
                m->lastsel = mou_ay - m->menuwin.ly - 1;
                ungetxch(CR);
                return true;
         }
        }
 return false;
}

int openpopup(popupmenu_t *m)
{
 int i;
 // open window
 if(!open_menuwindow(&m->menuwin))
        return false;
 // display menu
 htempwindow(m->menuwin.lx+1, m->menuwin.ly+1,
                                                        m->menuwin.rx-1, m->menuwin.ry-1);
 pushcolor();
 hsetcolor(m->normal_c);
 hsetbkcolor(m->normal_bc);
 for(i = 0; i < m->menunum ; i++) {
        if(m->menu[i].sub) {
         if(m->menu[i].active)
                ctrlprintfxy(1, i+1, "%sIII",m->menu[i].menu);
         else
                ctrlprintfxy(1, i+1, "^F%sIII^f",m->menu[i].menu);
        } else if(m->menu[i].active) {
         ctrlprintfxy(1, i+1, "%s   ",m->menu[i].menu);
        } else
         ctrlprintfxy(1, i+1, "^F%s   ^f",m->menu[i].menu);
 }
 popcolor();
 hlastwindow();
 return true;
}

void closepopup(popupmenu_t *m)
{
 close_menuwindow(&m->menuwin);
}

//
// 팝업메뉴를 끝내고자 할 경우 true를 리턴하고,
// 계속할 경우 false를 리턴한다
//
int popup(popupmenu_t *m)
{
 int sel, key, done = false, i, opened = false, escout = false;
 pushcursor();
 pushhanmode();
 _showcursor = false;
 _hangulmode = false;
 htempwindow(m->menuwin.lx+1, m->menuwin.ly+1,
```

```
                                                    m->menuwin.rx-1, m->menuwin.ry-1);
sel = m->lastsel;
do {
        // show cursor
        hsetcolor(m->selected_c);
        hsetbkcolor(m->selected_bc);
        if(!opened) {
         if(m->menu[sel].sub) {
                  if(m->menu[sel].active)
                   ctrlprintfxy(1, sel+1, "%sIII",m->menu[sel].menu);
                  else
                   ctrlprintfxy(1, sel+1, "^F%sIII^f",m->menu[sel].menu);
         } else if(m->menu[sel].active) {
                  ctrlprintfxy(1, sel+1, "%s  ",m->menu[sel].menu);
         } else
                  ctrlprintfxy(1, sel+1, "^F%s  ^f",m->menu[sel].menu);
        }
        status_help(m->menu[sel].help);

        if(m->menu[sel].sub)
         if(m->menu[sel].active && m->menu[sel].autosub && !opened) {
                  openpopup(m->menu[sel].sub);
                  opened = true;
         }
        // getkey
        key = hgetch();
        switch(key) {
         case LEFTARROW: case RIGHTARROW:
                  if(m->topdnmember) {
                   escout = true;
                   //done = true;
                   ungetchext(key);
                  }
                  break;
         case UPARROW:
                  sel = (sel + m->menunum -1) % m->menunum;
                  break;
         case DOWNARROW: case ' ':
                  sel = (sel + 1) % m->menunum;
                  break;
         case HOMEKEY:  case PGUPKEY:
                  sel = 0;
                  break;
         case ENDKEY:    case PGDNKEY:
                  sel = m->menunum - 1;
                  break;
         case ESC:
                  if(m->escout)
                   escout = true;
                  break;
         case F1:
                  break;
         case MOUKEY:
                  if(check_mou_is_in_sub(m->menu[sel].sub))
                   key = CR;
                  else if((mou_ax > m->menuwin.lx   && mou_ax < m->menuwin.rx)
                                   && (mou_ay >= m->menuwin.ly && mou_ay <= m->menuwin.ry
                                   && m->menu[mou_ay - m->menuwin.ly - 1].active))
                  {
                           sel = mou_ay - m->menuwin.ly - 1;
                           ungetxch(CR);
                  } else {
                           ungetxch(MOUKEY);
                           escout = true;
                  }
                  break;
         default:
                  //process hotkey
                  if(key < ' ')
                   break;
                  for(i=0; i < m->menunum; i++) {
                   if(m->menu[i].active && m->menu[i].hotkey == toupper(key)) {
                           sel = i;
                           ungetxch(CR);
                           break;
                   }
                  }
```

```
        }
        if(key == CR && m->menu[sel].active) {
         if(m->menu[sel].sub) {
                if(m->menu[sel].autosub) {
                 done = popup(m->menu[sel].sub);
//               popup(m->menu[sel].sub);
                } else {
                 if(openpopup(m->menu[sel].sub)) {
                        done = popup(m->menu[sel].sub);
//                      popup(m->menu[sel].sub);
                        closepopup(m->menu[sel].sub);
                 }
                }
         } else   if(m->menu[sel].itemfunc)
                if(m->closeandexecute) {
                 done = true;
                 menufunc = m->menu[sel].itemfunc;
                } else
                 done = m->menu[sel].itemfunc();
        }
        if(sel != m->lastsel || done || escout) {
         if(m->menu[m->lastsel].sub && m->menu[m->lastsel].autosub
                        && m->menu[m->lastsel].active) {
                closepopup(m->menu[m->lastsel].sub);
                opened = false;
         }
         // dispayhelp(m->menu[sel].help);
         hsetcolor(m->normal_c);
         hsetbkcolor(m->normal_bc);

         if(m->menu[m->lastsel].sub) {
                if(m->menu[m->lastsel].active)
                 ctrlprintfxy(1, m->lastsel+1, "%sIII",m->menu[m->lastsel].menu);
                else
                 ctrlprintfxy(1, m->lastsel+1, "^F%sIII^f",m->menu[m->lastsel].menu)
         } else if(m->menu[m->lastsel].active) {
                ctrlprintfxy(1, m->lastsel+1, "%s  ",m->menu[m->lastsel].menu);
         } else
                ctrlprintfxy(1, m->lastsel+1, "^F%s  ^f",m->menu[m->lastsel].menu);
        }
        m->lastsel = sel;
 } while(!done && !escout);
 status_help("");
 popcursor();
 pophanmode();
 hlastwindow();
 return done;
}

int popupmenu(popupmenu_t *m)
{
 int ret;
 if(!openpopup(m))
        return false;
 menufunc = NULL;
 ret = popup(m),
 closepopup(m);
 if(menufunc == NULL)
        return ret;
 else
        ret = menufunc();
 menufunc = NULL;
 return ret;
}

int opentopdn(topdnmenu_t *m)
{
 int i;
 if((m->bkgbuf = farmalloc(himagesize(-m->x,-m->y,

                m->x+ 8 * m->length, m->y+16))) == NULL) {
        return false;
 }
 hgetimage(-m->x,-m->y, m->x+8 * m->length, m->y+16, m->bkgbuf);
 m->old_fc = hgetcolor();
 m->old_bc = hgetbkcolor();
 pushcolor();
```

```
 hsetcolor(m->normal_c);
 hsetbkcolor(m->normal_bc);
 hprintfpxy(-m->x, -m->y,"%*s",m->length,"");
 m->xpos[0] = m->x;
 // make x position
 for(i = 1; i <= m->menunum; i++) {
        m->xpos[i] = m->xpos[i-1] + 8 * (strlen(m->topdnmenu[i-1].menu)+1);
 }
 for(i = 0; i < m->menunum; i++)
        ctrlprintfpxy(-m->xpos[i], -m->y,
                                                                m->topdnmenu[i].active?
"%s":"^F%s^f",
                                                        m->topdnmenu[i].menu);
 popcolor();
 return true;
}

void closetopdn(topdnmenu_t *m)
{
 if(m->bkgbuf) {
        hputimage(-m->x, -m->y, m->bkgbuf);
        farfree(m->bkgbuf);
        m->bkgbuf = NULL;
        hsetcolor(m->old_fc);
        hsetbkcolor(m->old_bc);
 }
}

int topdn(topdnmenu_t *m)
{
 int key, done = false, sel, i, j, isopen;
 sel = m->lastsel;
 isopen = m->autosub;
 pushcolor();
 pushhanmode();
 pushcursor();
 _showcursor = false;
 _hangulmode = false;
 do {
        // disp cursor
        hsetcolor(m->selected_c);
        hsetbkcolor(m->selected_bc);
        ctrlprintfpxy(-m->xpos[sel], -m->y,
                                                                m->topdnmenu[sel].active?
"%s":"^F%s^f",
                                                        m->topdnmenu[sel].menu);
        if(isopen && m->topdnmenu[sel].active) {
         done = popupmenu(m->topdnmenu[sel].popupmenu);
        }

        // get key
        if(done) {
         ungetxch(ESC);
        } else
         status_help(m->topdnmenu[sel].help);

        key = hgetch();
        switch(key) {
         case LEFTARROW:
                sel = (sel + m->menunum-1) % m->menunum;
                break;
         case RIGHTARROW:
                sel = (sel + 1) % m->menunum;
                break;
         case CR:
                if(!isopen)
                 isopen = true;
                break;
         case ESC:
                if(isopen)
                 isopen = false;
                if(m->escout)
                 done = true;
                break;
         case MOUKEY:
                if(mou_apx >= m->x && mou_apx <= m->x + 8 * m->length
                        && (mou_apy >= m->y && mou_apy < m->y+16)) {
```

```
                        for(i = 0; i < m->menunum; i++)
                         if(mou_apx >= m->xpos[i] && mou_apx < m->xpos[i+1] - 8)
                                 if(m->topdnmenu[i].active) {
                                  sel = i;
                                  ungetxch(CR);
                                 }
                } else
                 done = true;
                break;
        default:
                // process hotkey
                // search menu...
                for(i = 0; i < m->menunum; i++) {
                 if(!m->topdnmenu[i].active)
                         continue;
                 for(j = 0; j < m->topdnmenu[i].popupmenu->menunum; j++) {
                         if(m->topdnmenu[i].popupmenu->menu[j].hotkey == toupper(key) &&
                                 m->topdnmenu[i].popupmenu->menu[j].active) {
                          if(m->trace) {
                                  sel = i;
                                  if(!isopen)
                                   isopen = true;
                                  m->topdnmenu[i].popupmenu->lastsel = j;
                                  ungetxch(CR);
                          } else {
                                  done = m->topdnmenu[i].popupmenu->menu[j].itemfunc();
                          }
                         }
                 }
                }
                break;
        }
        // erase cursor
        if(sel != m->lastsel || done/* || key == CR*/) {
         hsetcolor(m->normal_c);
         hsetbkcolor(m->normal_bc);
         ctrlprintfpxy(-m->xpos[m->lastsel], -m->y,

m->topdnmenu[m->lastsel].active? "%s":"^F%s^f",

m->topdnmenu[m->lastsel].menu);
        }
        m->lastsel = sel;
 } while(!done);
 status_help("");
 popcolor();
 pophanmode();
 popcursor();
 return done;
}

int topdnmenu(topdnmenu_t *m)
{
 int ret;
 if(!opentopdn(m))
        return false;
 ret = topdn(m);
 closetopdn(m);
 return ret;
}

/*
 typedef struct _mywindow_
 {
        char lx, ly, rx, ry; // position
        char light_c, dark_c, board_c, glass_c, text_c,
                        title_frc, title_bkc;
        char oldfrc, oldbkc;
        char isinit;          // initialization occurred?
        char issavebkg;       // background saving flag
        char isactive;        // window activation flag
        char *winbuf;         // buffer for saving background
        char msg[71];         // title of window
        char menu;
//      int  win_manager;
 } mywindow_t;
```

```
 typedef struct _win_popupmenu_
 {
         mywindow_t menuwin;
         char menunum;
         char escout;
         char closeandexecute;
         char lastsel;
         char selected_c, normal_c, selected_bc, normal_bc;
         popup_item_t menu[_MAXPOPUPITEM_];
 } win_popupmenu_t;
*/

int win_openpopup(win_popupmenu_t *m)
{
 int i;
 // open window
 if(!open_window(&m->menuwin))
         return false;
 pushcolor();
 hsetcolor(m->normal_c),
 hsetbkcolor(m->normal_bc);
 for(i = 0; i < m->menunum ; i++) {
         if(m->menu[i].active)
          ctrlprintfxy(1, i+1, "%s",m->menu[i].menu),
         else
          ctrlprintfxy(1, i+1, "^F%s^f",m->menu[i].menu);
 }
 popcolor();
 return true;
}

void win_closepopup(win_popupmenu_t *m)
{
 close_window(&m->menuwin);
}

int win_popup(win_popupmenu_t *m)
{
 int sel, key, done = false, i, escout = false;
 pushcursor();
 pushhanmode();
 _showcursor = false;
 _hangulmode = false;
 sel = m->lastsel;
 do {
         // show cursor
         hsetcolor(m->selected_c);
         hsetbkcolor(m->selected_bc);
         if(m->menu[m->lastsel].active) {
                 ctrlprintfxy(1, m->lastsel+1, "%s",m->menu[m->lastsel].menu);
         } else
                 ctrlprintfxy(1, m->lastsel+1, "^F%s^f",m->menu[m->lastsel].menu);
         status_help(m->menu[m->lastsel].help);
         // getkey
         key = hgetch();
         switch(key) {
          case RIGHTARROW: case UPARROW:
                 sel = (sel + m->menunum -1) % m->menunum,
                 break;
          case LEFTARROW: case DOWNARROW:
          case ' ':
                 sel = (sel + 1) % m->menunum;
                 break;
          case HOMEKEY: case PGUPKEY:
                 sel = 0;
                 break;
          case ENDKEY:    case PGDNKEY:
                 sel = m->menunum - 1;
                 break;
          case ESC:
                 if(m->escout)
                  escout = true;
                 break;
          case F1:
                 break;
          case MOUKEY:
                 if((mou_ax > m->menuwin.lx          && mou_ax < m->menuwin.rx)
```

```
                                        && (mou_ay >= m->menuwin.ly+2 && mou_ay < m->menuwin.ry
                                        && m->menu[mou_ay - m->menuwin.ly - 2].active))
                {
                        sel = mou_ay - m->menuwin.ly - 2;
                        ungetxch(CR);
                } else {
                        ungetxch(MOUKEY);
                        escout = true;
                }
                break;
        default:
                //process hotkey
                if(key < ' ')
                 break;
                for(i=0; i < m->menunum; i++) {
                 if(m->menu[i].active && m->menu[i].hotkey == toupper(key)) {
                        sel = i;
                        ungetxch(CR);
                        break;
                 }
                }
         }
         if(key == CR && m->menu[sel].active) {
          if(m->menu[sel].itemfunc)
                if(m->closeandexecute) {
                 done = true;
                 menufunc = m->menu[sel].itemfunc;
                } else
                 done = m->menu[sel].itemfunc();
         }
          // dispayhelp(m->menu[sel].help);
         if(sel != m->lastsel) {
          hsetcolor(m->normal_c);
          hsetbkcolor(m->normal_bc);
          if(m->menu[m->lastsel].active) {
                ctrlprintfxy(1, m->lastsel+1, "%s",m->menu[m->lastsel].menu);
          } else
                ctrlprintfxy(1, m->lastsel+1, "^F%s^f",m->menu[m->lastsel].menu);
          m->lastsel = sel;
         }
 } while(!done && !escout);
 status_help("");
 popcursor();
 pophanmode();
 return done;
}

int win_popupmenu(win_popupmenu_t *m)
{
 int ret;
 if(!win_openpopup(m))
        return false;
 menufunc = NULL;
 ret = win_popup(m);
 win_closepopup(m);
 if(menufunc == NULL)
        return ret;
 else
        ret = menufunc();
 menufunc = NULL;
 return ret;
}
```

```
/* *************************************** *
 *       온실제어 프로그램 version 0.0
 * *************************************** */

#include <string.h>
#include <ctype.h>
#include <stdarg.h>
#include <stdio.h>
#include <dos.h>
#include <alloc.h>
#include <stdlib.h>
#include <dir.h>
#include <io.h>
#include <bios.h>


#include "ascii.h"
#include "extkey.h"
#include "hanout.h"
#include "hanin.h"
#include "hanwout.h"
#include "hanlib.h"
#include "mystdio.h"
#include "hanmou.h"
#include "hanmeq.h"
#include "mywindow.h"

#ifndef true
 #define true 1
#endif
#ifndef false
 #define false 0
#endif

int printsw = 0;

unsigned char far wait_mou_cursor[576] =
{
        12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12
        0, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 15, 0, 15, 0, 15, 0, 15, 0, 15, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 15, 0, 15, 0, 15, 0, 15, 0, 15, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 0, 4, 15, 0, 15, 0, 15, 0, 15, 0, 4, 0, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 0, 0, 4, 15, 0, 15, 0, 15, 0, 4, 0, 0, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 0, 0, 0, 4, 15, 0, 15, 0, 4, 0, 0, 0, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 0, 0, 0, 4, 0, 15, 0, 15, 4, 0, 0, 0, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 0, 0, 4, 0, 0, 0, 15, 0, 0, 4, 0, 0, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 0, 4, 0, 0, 0, 15, 0, 0, 0, 0, 4, 0, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 0, 0, 0, 0, 15, 0, 0, 0, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 0, 0, 0, 15, 0, 0, 0, 0, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 0, 0, 0, 0, 15, 0, 0, 0, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 0, 15, 0, 15, 0, 15, 0, 15, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 15, 0, 15, 0, 15, 0, 15, 0, 15, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 15, 0, 15, 0, 15, 0, 15, 0, 15, 0, 4, 0, 0, 12, 12, 0, 0,
        0, 0, 12, 12, 0, 0, 4, 0, 15, 0, 15, 0, 15, 0, 15, 0, 15, 4, 0, 0, 12, 12, 0, 0,
        0, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 0,
        12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12
};

unsigned char far wait_mou_cursor_mask[72] =
{
        0xff,
        0xff,
        0xff,
        0x7f,
        0xff,
        0xfe,
        0x32,
        0x00,
        0x4c,
        0x32,
        0x00,
```

```
        0x4c,
        0x32,
        0x00,
        0x4c,
        0x32,
        0x00,
        0x4c,
        0x32,
        0x00,
        0x4c,
        0x32,
        0xaa,
        0xcc,
        0x33,
        0x55,
        0x4c,
        0x31,
        0xaa,
        0x8c,
        0x30,
        0xd5,
        0x0c,
        0x30,
        0x6a,
        0x0c,
        0x30,
        0x56,
        0x0c,
        0x30,
        0x89,
        0x0c,
        0x31,
        0x10,
        0x8c,
        0x32,
        0x08,
        0x4c,
        0x32,
        0x10,
        0x4c,
        0x32,
        0x08,
        0x4c,
        0x32,
        0x55,
        0x4c,
        0x32,
        0xaa,
        0xcc,
        0x33,
        0x55,
        0x4c,
        0x32,
        0xaa,
        0xcc,
        0x7f,
        0xff,
        0xfe,
        0xff,
        0xff,
        0xff,
};


moucursor_t _wait_mouse_ = {
 24, 24,
 0, 0,
 wait_mou_cursor,
 wait_mou_cursor_mask,
 255,
 NULL
};

extern int (*savescreen)(void);
extern int (*restorescreen)(void);
```

```
typedef struct _dir_list_
{
        byte tag;
        char filename[9];
        char ext[5];
        char description[35];
        char attrib;
        long size;
        struct _dir_list_ *next;
        struct _dir_list_ *prev;
} dir_list;

////////////////////////////////
////////////////////////////////
extern int isautomatic;

int filecount;
char defaultfn[9] = "*";
char defaultext[5] = ".*";

int allow_mou_drag = false;
////////////////////////////////
////////////////////////////////


void (*_hookhwaitkey_bkg_job[MAXTASK])(void)
= {NULL, NULL, NULL, NULL, NULL,NULL, NULL, NULL, NULL, NULL,
        NULL, NULL, NULL, NULL, NULL,NULL, NULL, NULL, NULL, NULL}

int mou_x, mou_y, mou_ax, mou_ay, mou_px, mou_py, mou_apx, mou_apy;

int ishangulmodelock;

window_t statuswin;
int is_status_win_init = false;

static char _color_stack_[10][2];
static char _top = 0;
static char _cursor_stack_[10];
static char _cursor_top = 0;
static char _hanmode_stack_[10];
static char _hanmode_top = 0;

void run_beep(void)
{
        sound(4000);
        delay(2);
        nosound();
}

void mou_wait(int sw)
{
        if(sw)
                mou_changecursor(&_wait_mouse_);
        else
                mou_changecursor(MOU_ARROW256);
}

void pushcolor(void)
{
 if(_top < 10) {
        _color_stack_[_top][0] = hgetcolor();
        _color_stack_[_top++][1] = hgetbkcolor();
 }
}

void popcolor(void)
{
 if(_top > 0) {
        hsetcolor(_color_stack_[--_top][0]);
        hsetbkcolor(_color_stack_[_top][1]);
 }
}

void pushcursor(void)
{
 if(_top < 10)
```

```
        _cursor_stack_[_cursor_top++] = _showcursor;
}

void popcursor(void)
{
 if(_cursor_top > 0)
        _showcursor= _cursor_stack_[--_cursor_top];
}

void pushhanmode(void)
{
 if(_hanmode_top < 10)
        _hanmode_stack_[_hanmode_top++] = _hangulmode;
}

void pophanmode(void)
{
 if(_hanmode_top > 0)
        _hangulmode = _hanmode_stack_[--_hanmode_top];
}

void ctrlputsxy(int x, int y, uchar *s)
{
 if(x < 0 && y < 0)
        ctrlputspxy((x+1)*8, (y+1)*16, s);
 else
        ctrlputspxy((x-1)*8, (y-1)*16, s);
}

void ctrlputs(uchar *s)
{
 ctrlputsxy(hwherex(), hwherey(), s);
}

void ctrlputspxy(int x, int y, unsigned uchar *s)
{
 while(*s) {
        if(*s == '^' && strchr("RrSsUuTtFfBbAaOoLlCc", *(s+1))) {
         switch(*(++s)) {
                case 'R': //reverse on
                 hsetreverse(true);
                 break;
                case 'r': //         off
                 hsetreverse(false);
                 break;
                case 'S': //shadow   on
                 hsetshadow(true);
                 break;
                case 's': //         off
                 hsetshadow(false);
                 break;
                case 'U': //under    on
                 hsetunder(true);
                 break;
                case 'u': //         off
                 hsetunder(false);
                 break;
                case 'T': //threed   on
                 hsetthreed(true);
                 break;
                case 't': //         off
                 hsetthreed(false);
                 break;
                case 'F': //faint    on
                 hsetfaint(true);
                 break;
                case 'f': //         off
                 hsetfaint(false);
                 break;
                case 'B': //bold     on
                 hsetbold(true);
                 break;
                case 'b': //         off
                 hsetbold(false);
                 break;
                case 'A': //frame    on
                 hsetframe(true);
```

```
                    break;
                  case 'a': //          off
                    hsetframe(false);
                    break;
                  case 'O': //outline on
                    hsetoutline(true);
                    break;
                  case 'o': //          off
                    hsetoutline(false);
                    break;
                  case 'L': //block    on
                    hsetblock(true);
                    break;
                  case 'l': //          off
                    hsetblock(false);
                    break;
                  case 'C': // set foreground color
                    if(*++s - '0' > 9)
                            hsetcolor(toupper(*s)-'A'+10);
                    else
                            hsetcolor(*s-'0');
                    break;
                  case 'c': // set background color
                    if(*++s - '0' > 9)
                            hsetbkcolor(toupper(*s)-'A'+10);
                    else
                            hsetbkcolor(*s-'0');
                    break;
            }
          } else {
            if(*s > 0x80) {
                  _hputchpxy(*s, *(s+1), x, y);
                  s++;
                  if(x < 0)
                    x -= 8;
                  else
                    x += 8;
            } else
                  _eputchpxy(*s, x, y);
            if(x < 0)
                  x -= 8;
            else
                  x += 8;
          }
          s++;
  }
}

void ctrlputspxy_v(int x, int y, unsigned uchar *s)
{
  while(*s) {
          if(*s == '^' && strchr("RrSsUuTtFfBbAaOoLlCc", *(s+1))) {
            switch(*(++s)) {
                  case 'R': //reverse on
                    hsetreverse(true);
                    break;
                  case 'r': //          off
                    hsetreverse(false);
                    break;
                  case 'S': //shadow   on
                    hsetshadow(true);
                    break;
                  case 's': //          off
                    hsetshadow(false);
                    break;
                  case 'U': //under    on
                    hsetunder(true);
                    break;
                  case 'u': //          off
                    hsetunder(false);
                    break;
                  case 'T': //threed   on
                    hsetthreed(true);
                    break;
                  case 't': //          off
                    hsetthreed(false);
                    break;
```

```
                case 'F': //faint    on
                 hsetfaint(true);
                 break;
                case 'f': //         off
                 hsetfaint(false);
                 break;
                case 'B': //bold     on
                 hsetbold(true);
                 break;
                case 'b': //         off
                 hsetbold(false);
                 break;
                case 'A': //frame    on
                 hsetframe(true);
                 break;
                case 'a': //         off
                 hsetframe(false);
                 break;
                case 'O': //outline on
                 hsetoutline(true);
                 break;
                case 'o': //         off
                 hsetoutline(false);
                 break;
                case 'L': //block    on
                 hsetblock(true);
                 break;
                case 'l': //         off
                 hsetblock(false);
                 break;
                case 'C': // set foreground color
                 if(*++s - '0' > 9)
                         hsetcolor(toupper(*s)-'A'+10);
                 else
                         hsetcolor(*s-'0');
                 break;
                case 'c': // set background color
                 if(*++s - '0' > 9)
                         hsetbkcolor(toupper(*s)-'A'+10)
                 else
                         hsetbkcolor(*s-'0');
                 break;
          }
         } else {
          if(*s > 0x80) {
                 _hputchpxy(*s, *(s+1), x, y);
                 s++;
          } else
                 _eputchpxy(*s, x, y);
          if(y < 0)
                 y -= 16;
          else
                 y += 16;
         }
         s++;
 }
}

void ctrlprintf(uchar *format, ...)
{
 static uchar buffer[1024];
 va_list arglist;
 va_start(arglist, format);
 vsprintf(buffer, format, arglist);
 ctrlputs(buffer);
 va_end(arglist);
}

void ctrlprintfxy(int x, int y, uchar *format, ...)
{
 static uchar buffer[1024];
 va_list arglist;
 va_start(arglist, format);
 vsprintf(buffer, format, arglist);
 ctrlputsxy(x, y, buffer);
 va_end(arglist);
}
```

```
void ctrlprintfpxy(int x, int y, uchar *format, ...)
{
 static uchar buffer[1024];
 va_list arglist;
 va_start(arglist, format);
 vsprintf(buffer, format, arglist);
 ctrlputspxy(x, y, buffer);
 va_end(arglist);
}



void hookhwaitkey(void)
{
 mouevnque_t *mouevn;
 char task;
 if(!meq_isempty()) {
          mouevn = meq_getevn();
          if(meq_isldragevn(mouevn) && allow_mou_drag) {
                   meq_skipevn(XEVN_LDRAG);
                   mou_x  = meq_getx(mouevn);
                   mou_y  = meq_gety(mouevn);
                   mou_ax = meq_getax(mouevn);
                   mou_ay = meq_getay(mouevn);
                   mou_px = meq_getpx(mouevn);
                   mou_py = meq_getpy(mouevn);
                   mou_apx = meq_getapx(mouevn);
                   mou_apy = meq_getapy(mouevn);
                   ungetxch(MOUKEY);
          } else    if(meq_isldownevn(mouevn)) {
                    mou_x  = meq_getx(mouevn);
                    mou_y  = meq_gety(mouevn);
                    mou_ax = meq_getax(mouevn);
                    mou_ay = meq_getay(mouevn);
                    mou_px = meq_getpx(mouevn);
                    mou_py = meq_getpy(mouevn);
                    mou_apx = meq_getapx(mouevn);
                    mou_apy = meq_getapy(mouevn);
                    ungetxch(MOUKEY);
          }
          if(meq_isrdownevn(mouevn)) {
                   ungetxch(ESC);
          }
 }
 for(task = 0; task < MAXTASK; task++)
         if(_hookhwaitkey_bkg_job[task]!= NULL)
                  _hookhwaitkey_bkg_job[task]();
}

int registertask(void (*func)(void))
{
 char task;
 for(task = 0; task < MAXTASK; task++)
         if(_hookhwaitkey_bkg_job[task] == NULL) {
          _hookhwaitkey_bkg_job[task] = func;
          return task;
         }
 return -1;
}

int killtask(int jobnumber)
{
 if(jobnumber < 0 || jobnumber >= MAXTASK)
         return false;
 if(_hookhwaitkey_bkg_job[jobnumber] != NULL) {
          _hookhwaitkey_bkg_job[jobnumber] = NULL;
          return true;
 }
 return false;
}

int askanswer(char *msg,int dftans)
{
         mywindow_t askans;
         int sel, len, ret = dftans;
```

```
        len = (strlen(msg)+12);
        len += (len % 2);
        setmywindow(&askans, 64-len/2, 12, 64+len/2, 15, WHITE, BLACK,
                        LIGHTGRAY, CYAN, WHITE, BLACK, LIGHTGRAY, true, true, msg
false);
        pushcursor();
        pushhanmode();
        if(open_window(&askans)) {
                _hangulmode = false;
                _showcursor = false;
                do {
                        if(ret) {
                                pushcolor();
                                hsetbkcolor(LIGHTBLUE);
                                hprintfxy(1, 1,"%*s",len/2,"");
                                hprintfxy(1, 1,"%*s예",len/4-1,"");
                                popcolor();
                                hprintfxy(len/2+1,1,"%*s",len/2-1,"");
                                hprintfxy(len/2+1, 1,"%*s아니오",len/4-3,"");
                        } else {
                                hprintfxy(1, 1,"%*s",len/2,"");
                                hprintfxy(1, 1,"%*s예",len/4-1,"");
                                pushcolor();
                                hsetbkcolor(LIGHTBLUE);
                                hprintfxy(len/2+1, 1,"%*s",len/2-1,"");
                                hprintfxy(len/2+1, 1,"%*s아니오",len/4-3,"");
                                popcolor();
                        }
                        sel = hgetch();
                        switch(sel) {
                                case LEFTARROW:
                                        ret = true;
                                        break;
                                case RIGHTARROW:
                                        ret = false;
                                        break;
                                case ESC:
                                        ret = -1;
                                        break;
                                case 'Y': case 'y':
                                        ret = true;
                                        sel = CR;
                                        break;
                                case 'N': case 'n':
                                        ret = false;
                                        sel = CR;
                                        break;
                                case MOUKEY:
                                        if(mou_y==1) {
                                                if(mou_x >= 1 && mou_x < len /2) {
                                                        ret = true;
                                                        sel = CR;
                                                } else if(mou_x > len / 2 && mou_x < len) {
                                                        ret = false;
                                                        sel = CR;
                                                }
                                        }
                        }
                } while(sel != CR && sel != ESC);
                close_window(&askans);
        } else
         ret = -1;
        popcursor();
        pophanmode();
        return ret;
}

void puterror(char *msg)
{
        mywindow_t errmsg;
        int len;

        len = (strlen(msg)+12);
        len += (len % 2);
        setmywindow(&errmsg, 64-len/2, 13, 64+len/2, 17, WHITE, BLACK,
                        MAGENTA, RED, WHITE, BLACK, LIGHTGRAY, true, true,
                        "잘못되었습니다", false);
```

```
        if(open_window(&errmsg)) {
                pushcursor );
                pushhanmode();
                _showcursor = false;
                _hangulmode = false;
                hsettextjustify(CENTER_TEXT);
                hprintfxy(len/2,1,msg);
                hsetcolor(BLUE);
                hprintfxy(len/2,2,"[잠시 기다리세요]");
                hsettextjustify(LEFT_TEXT);
                sound(1000);
                delay(30);
                nosound();
                sleep(2);
                pophanmode();
                popcursor();
                close_window(&errmsg);
        }       else
                hprintf("%s\n", msg);
}

void putmsg(char *msg, int sw)
{
        static mywindow_t msgwin[10];
        static int stacktop = 0;
        int len;

        if(sw) { //sw == true => open else close
         if(stacktop < 9)
         {
                len = (strlen(msg)+12);
                len += (len % 2);
                setmywindow(&msgwin[stacktop], 64-len/2, 5, 64+len/2, 8, WHITE, BLACK,
                                GREEN, BLUE, WHITE, BLACK, LIGHTGRAY, true, true
                                "알려 드릴 말씀이 있어요", false);
                if(open_window(&msgwin[stacktop++]))
                        hsettextjustify(CENTER_TEXT);
                        hprintfxy(len/2,1,msg);
                        hsettextjustify(LEFT_TEXT);
         }
        } else {
                if(stacktop > 0)
                        close_window(&msgwin[--stacktop]);
        }
}

int win_getline(uchar *msg, uchar *str, int hanmodelock, int maxlen)
{
 int ret = false;
 mywindow_t getln;
 if(maxlen > 72)
        return false;
 setmywindow(&getln, 64-maxlen/2, 12, 64+maxlen/2, 15, WHITE, BLACK,
                LIGHTGRAY, CYAN, BLACK, BLACK, LIGHTGRAY, true, true,
                msg, false);
 if(open_window(&getln)) {
        pushhanmode();
        pushcursor();
        switch(hanmodelock)
        {
                case HANONLY:
                case HAN:
                        _hangulmode = true;
                        break;
                case ENGONLY:
                case ENG:
                        _hangulmode = false;
                        break;
        }
        ishangulmodelock = hanmodelock;
        _showcursor = true;
        ret = hgetln(str, maxlen);
        switch(hanmodelock)
        {
                case HANONLY:
                case ENGONLY:
                        ishangulmodelock = false;
```

```
                        break;
        }
        popcursor();
        pophanmode();
        close_window(&getln);
 }
 if(ret == ESC)
        return false;
 return ret;
}

int win_getdata(uchar *msg, uchar *data, char *pic, int hanmodelock)
{
 int ret = false, maxlen;
 mywindow_t g;
 maxlen = strlen(pic);
 if(maxlen <  strlen(msg))
        maxlen = strlen(msg)+6;
 setmywindow(&g, 64-maxlen/2, 12, 64+maxlen/2, 15, WHITE, BLACK,
                LIGHTGRAY, CYAN, BLACK, BLACK, LIGHTGRAY, true, true
                msg, false);
 if(open_window(&g)) {
        pushhanmode();
        pushcursor();
        switch(hanmodelock)
        {
                case HANONLY:
                case HAN:
                        _hangulmode = true;
                        break;
                case ENGONLY:
                case ENG:
                        _hangulmode = false;
                        break;
        }
        ishangulmodelock = hanmodelock;
        _showcursor = true;
        ret = hgetdata(data, pic);
        switch(hanmodelock)
        {
                case HANONLY:
                case ENGONLY:
                        ishangulmodelock = false;
                        break;
        }
        popcursor();
        pophanmode();
        close_window(&g);
 }
 if(ret == ESC)
        return false;
 return true;
}

void hookhangulmodetoggle(bool hangulmode)
{
 if(!is_status_win_init)
        return;
 if(ishangulmodelock == HANONLY) {
        if(!hangulmode)
                puterror("한글만 입력하실 수 있습니다");
        _hangulmode = true;
 }
 if(ishangulmodelock == ENGONLY) {
        if(hangulmode)
                puterror("영문만 입력하실 수 있습니다");
        _hangulmode = false;
 }
 htempcurwindow(&statuswin);

 hputsxy(1, 1, _hangulmode ? " 한글 |" : " 영문 |");
 if (_hangulmode)
        hsetbufcursor(FRAMECURSOR, FRAMECURSOR, _cursorcolor);
 else
        hsetbufcursor(UNDERCURSOR, UNDERCURSOR, _cursorcolor);

 hlastcurwindow();
```

```
}

void hookinsertmodetoggle(bool insertmode)
{
        if(!is_status_win_init)
          return;
        htempcurwindow(&statuswin);

        wputsxy(&statuswin, 9, 1, insertmode ? " 삽입 |" : " 수정 |");

        hsetbufcursor(NULL, NULL, _cursorcolor);

        hlastcurwindow();
}

void status_clock(void)
{

        static sec=60;
        struct time timeinfo;
        struct date dateinfo;
        static byte buffer[50];
        if(!is_status_win_init)
          return;
        gettime(&timeinfo);
        if(sec == timeinfo.ti_sec) return;
        if(sec % 30 == 0)
                run_beep();

        getdate(&dateinfo);
        sprintf(buffer, " | %4s | %4d년 %2d월 %2d일 | %2d:%02d:%02d%2s",
                isautomatic? "자동":"수동",
                dateinfo.da_year, dateinfo.da_mon, dateinfo.da_day,
                (timeinfo.ti_hour > 12)? timeinfo.ti_hour-12:timeinfo.ti_hour,
                timeinfo.ti_min,timeinfo.ti_sec,
                (timeinfo.ti_hour > 11)? " [오후]":" [오전]");
        wputsxy(&statuswin, 88, 1, buffer);
        if(timeinfo.ti_min == 0 && timeinfo.ti_sec == 0) {
                        sound(1500);
                        delay(40);
                        nosound();
        }
        sec = timeinfo.ti_sec;
}

void status_help(uchar *msg)
{
 if(!is_status_win_init)
        return;
 wprintfxy(&statuswin, 17, 1, "%-70s",msg);
}

void init_status_win(void)
{
 hsetwindow(&statuswin, 1, hgetmaxy(), hgetmaxx(), hgetmaxy(), BLACK, WHITE, DEFWIN)
 wclrscr(&statuswin);
 _hookhangulmodetoggle = hookhangulmodetoggle;
 _hookinsertmodetoggle = hookinsertmodetoggle;
 is_status_win_init = true;
 hookhangulmodetoggle(_hangulmode);
 hookinsertmodetoggle(_insertmode);
}

void switch_status_win(int sw)
{
 static int init = false;
 if(is_status_win_init)
        init = true;
 if(init)
        is_status_win_init = sw;
}

void allow_darg(int sw)
{
        allow_mou_drag = sw;
}
```

```
dir_list *addlist(dir_list *dirlist, char *fn, char *ext, int att, long size, char *des)
{
        if(dirlist == NULL)
        {
                if((dirlist = malloc(sizeof(dir_list))) != NULL)
                {
                        strcpy(dirlist->filename,fn);
                        strcpy(dirlist->ext, ext);
                        sprintf(dirlist->description,"%-34s", des);
                        dirlist->attrib = att;
                        dirlist->size = size;
                        dirlist->next = NULL;
                        dirlist->prev = NULL;
                        dirlist->tag = false;
                }
        } else {
                dirlist->next = addlist(dirlist->next, fn, ext, att, size, des);
                dirlist->next->prev = dirlist;
        }
        return dirlist;
}


/*
 FA_RDONLY  Read-only attribute
 FA_HIDDEN  Hidden file
 FA_SYSTEM  System file
 FA_LABEL   Volume label
 FA_DIREC   Directory
 FA_ARCH    Archive
*/

dir_list *readdir(dir_list *dirlist, char *path)
{
        struct ffblk ffblk;
        int done;
        char dirpath[_MAX_PATH];
        char drive[_MAX_DRIVE];
        char dir[_MAX_DIR];
        char file[_MAX_FNAME];
        char ext[_MAX_EXT];

        _splitpath(path,drive,dir,file,ext);
        strcpy(file,"*");
        strcpy(ext,".*");
        _makepath(dirpath, drive, dir, file, ext);

        done = findfirst(dirpath,&ffblk,FA_DIREC);
        while(!done)
                {
                if(ffblk.ff_attrib == FA_DIREC)
                {
                        _splitpath(ffblk.ff_name,drive,dir,file,ext);
                        if(strcmp(dir,"."))
                        {
                                if(*dir)
                                        dirlist = addlist(dirlist, dir, ext, ffblk.ff_attrib,ffblk.ff_fsize
"");
                                else
                                        dirlist = addlist(dirlist, file, ext, ffblk.ff_attrib,ffblk.ff_fsize
"");
                                filecount++;
                        }
                }
                done = findnext(&ffblk);
        }
        done = findfirst(path,&ffblk,FA_ARCH);
        while(!done)
        {

                _splitpath(ffblk.ff_name,NULL,NULL,file,ext);
                dirlist = addlist(dirlist, file, ext, ffblk.ff_attrib,ffblk.ff_fsize, "");
                done = findnext(&ffblk);
                filecount++;
        }
        return dirlist;
}
```

```
void freedir(dir_list *dirlist)
{
        if(dirlist->next)
                freedir(dirlist->next);
        if(dirlist)
                free(dirlist);
}


long diskfree(int drive)
{
         struct dfree free;
         getdfree(drive+1, &free);
         return (long) free.df_avail * (long) free.df_bsec          * (long) free.df_sclus;
}

uchar *dirbox(uchar *msg, uchar *retpath, uchar selected[100][81], uchar sel)
{
        dir_list *root = NULL, *cur, *top, *disp;
        mywindow_t dirboxwin;
        char drive[_MAX_DRIVE];
        char dir[_MAX_DIR];
        char file[_MAX_FNAME];
        char ext[_MAX_EXT];
        static char path[61]="";
        long freespace, tagedsize;
        int done = false, chgdir = false, key, i, pos, offset, tageditem, curpos, redraw = true;
        button_t btn[4];
        char *btnmsg[4]={"새경로 TAB","선택 SP","확인 CR","취소 ESC"};
        int btnkey[4] = {TAB,' ',CR,ESC};

        if(*retpath)
                strcpy(path, retpath);
        else
                sprintf(path,"%s%s", defaultfn, defaultext);
        *retpath = 0;

        for(i=0; i<4; i++)
        {
                set_button(&btn[i], 33 + 12*i, 22, 45+ 12*i, 24,
                                                                LIGHTCYAN,    BLACK,
CYAN, LIGHTGREEN, BLACK,
                                                                false, false, btnmsg[i]);
        }

//      if(win_getline("농장화일명을 입력하십시오", path, ENGONLY, 60))
//      {
                if(access(path,0))
                {
                        setmywindow(&dirboxwin, 32, 6, 99, 24, WHITE, BLACK, DARKGRAY,
BLUE,
                                                                WHITE,   WHITE,   GREEN,
false, true, msg, false);
                        savescreen();
                        if(open_window(&dirboxwin)) {
                                pushhanmode();
                                _hangulmode = true;
                                while(!done)
                                {
                                        hclrscr();
                                        strupr(path);
                                        _splitpath(path,drive,dir,file,ext);
                                        chgdir = false;
                                        if(!*drive)
                                                sprintf(drive,"%c:",getdisk()+'A');
                                        if(!*dir)
                                        {
                                                *dir = '\\';
                                                if(getcurdir(*drive-'A'+1,dir+1))
                                                {
                                                        puterror("드라이브명이   잘못되었습니다
");
                                                        break;
                                                }
                                        }
                                        if(!*file)
```

```
                                        strcpy(file, defaultfn);
                                if(!*ext)
                                        strcpy(ext, defaultext);
                                _makepath(path,drive,dir,file,ext);
                                _splitpath(path,drive,dir,file,ext);
                                filecount = 0;
                                tageditem = 0;
                                tagedsize = 0;
                                cur = top = disp = root = readdir(root, path);
                                if(root == NULL)
                                {
                                        puterror("그런 경로명은 없습니다");
                                        break;
                                }
                                if(sel)
                                {
                                        for(; cur != NULL; cur = cur->next)
                                        {
_makepath(retpath,drive,dir,cur->filename,cur->ext);
                                                for(i = 0; *selected[i]; i++)
                                                        if(!strcmp(selected[i], retpath))
                                                        {
                                                                cur->tag = true;
                                                                tageditem++;
                                                                tagedsize           +=
cur->size;
                                                                break;
                                                        }
                                        }
                                        cur = root;
                                }


                                pos = offset = 0;
                                freespace = diskfree(*drive - 'A');

                                for(i=0; i<4; i++)              open_button(&btn[i]);

                                ctrlputspxy_v(512,32,"^C3^R^A▲^a^r■■■■■■■■■^R^A
▼^a^r^CF");

                                hsetreverse(true);
                                hprintfxy(1,2,"%-66s","선택          화일명          구분/크기
    덧붙임말                    | ");
                                hsetreverse(false);
                                hprintfxy(1,1," 경로 %-60s", path);

                                while(!chgdir)
                                {
                                        hsetreverse(true);
                                        hsetunder(true);
                                        hprintfxy(1,13," %3d [%8ld 바이트] 로그화일이 선
택되었습니다                    ",
                                tageditem, tagedsize);
                                        hprintfxy(1,14," %12ld [%10.1f K, %7.2f M] 바이
트 남았습니다          ",
                                freespace, (float)freespace/1024.,
                                (float)freespace / 1048576.);
                                        hsetunder(false);
                                        hsetreverse(false);
                                        //display file
                                        for(i = 0, disp = top;
                                                        i<10    &&    disp!=NULL    &&
!_bioskey(1) && redraw;
                                                        i++, disp = disp->next)
                                        {
                                                if(disp->attrib == FA_DIREC)
                                                        hprintfxy(1,3+i,"           %2s
%-8s%-4s [디렉토리] %-34s  ",
                                        (disp->tag)? ">>":"  ", disp->filename,
                                        disp->ext, disp->description);
```

```
                                        else
                                                hprintfxy(1,3+i,"          %2s
%-8s%-4s %10ld %-34s  ",
                        (disp->tag)? ">>":"  ",disp->filename,
                        disp->ext, disp->size, disp->description);
                                }
                                if(!_bioskey(1))
                                        redraw = false;
                                // draw cursor
                                pushcolor();
                                hsetcolor(BLACK);
                                hsetbkcolor(GREEN);
                                if(cur->attrib == FA_DIREC)
                                        hprintfxy(1,3+offset,"  %2s  %-8s%-4s
[디렉토리] %-34s  ",
                (cur->tag)? ">>":"  ",cur->filename,
                cur->ext, cur->description);
                                else
                                        hprintfxy(1,3+offset,"  %2s  %-8s%-4s
%10ld %-34s  ",
                (cur->tag)? ">>":"  ",cur->filename,
                cur->ext, cur->size, cur->description);
                                popcolor();
                                if(filecount > 1)
                                        curpos = (int)(112. / (float)(filecount-1)
* (float)(pos+offset))+48;
                                else
                                        curpos = 48;
                                ctrlputspxy(512,curpos,"^A^R■^r^a");
                                // get key
                                key = hgetch();
                                // erase cursor
                                if(cur->attrib == FA_DIREC)
                                        hprintfxy(1,3+offset,"  %2s  %-8s%-4s
[디렉토리] %-34s  ",
                (cur->tag)? ">>":"  ",cur->filename,
                cur->ext, cur->description);
                                else
                                        hprintfxy(1,3+offset,"  %2s  %-8s%-4s
%10ld %-34s  ",
                (cur->tag)? ">>":"  ",cur->filename,
                cur->ext, cur->size, cur->description);
                                ctrlputspxy(512,curpos,"^C3■^CF");
                                switch(key)
                                {
                                        case DOWNARROW:
                                                if(offset < 9)
                                                {
                                                        if(cur->next)
                                                        {
                                                                offset++;
                                                                cur=
cur->next;
                                                        }
                                                } else
                                                if(pos < filecount - 10)
                                                {
                                                        pos++;
                                                        if(top->next)
                                                                top
top->next;
                                                        if(cur->next)
                                                        {
                                                                cur=
```

```
cur->next;
                                        }
                                        redraw = true;
                                }
                                break;
                        case UPARROW:
                                if(offset > 0)
                                {
                                        if(cur->prev)
                                        {
                                                offset--;
                                                cur     =
cur->prev;
                                        }
                                } else
                                if(top->prev)
                                {
                                        cur   =   top   =
top->prev;
                                        pos--;
                                        redraw = true;
                                }
                                break;
                        case PGDNKEY:
                                if(pos + 20 < filecount)
                                {
                                        for(i=0; i < 10; i++)
                                        {
                                                cur = top =
top->next;
                                                pos++;
                                        }
                                        offset = 0;
                                } else if(pos < filecount - 10)
                                {
                                        while(pos           <
filecount-10)
                                        {
                                                cur = top =
top->next;
                                                pos++;
                                        }
                                        offset  = 0;
                                } else
                                {
                                        while(cur->next)
                                        {
                                                cur       =
cur->next;
                                                offset ++;
                                        }
                                }
                                redraw = true;
                                break;
                        case PGUPKEY:
                                if(pos > 10)
                                {
                                        for(i=0; i < 10; i++)
                                        {
                                                cur = top =
top->prev;
                                                pos--;
                                        }
                                        offset = 0;
                                } else
                                {
                                        cur = top = root;
                                        pos = 0;
                                        offset = 0;
                                }
                                redraw = true;
                                break;
                        case ' ':
                                if(cur->attrib  !=  FA_DIREC
&& sel)
                                {
                                        if(cur->tag)
```

```
                                        {
false;                                          cur->tag =
tageditem--;
-= cur->size;                                   tagedsize
                                        } else
                                        {
true;                                           cur->tag =
tageditem++;
+= cur->size;                                   tagedsize
                                        }
                                }
                                break;
                        case CR:
                                chgdir = true;
                                if(cur->attrib != FA_DIREC)
                                {
_makepath(retpath,drive,dir,cur->filename,cur->ext);
                                        if(sel)
                                        {
memset(selected, 0, sizeof selected);
root, i=0; disp != NULL && i < 100; disp = disp->next)  for(disp =
                                        {
if(disp->tag)                                           {
        _makepath(retpath,drive,dir,disp->filename,disp->ext);
        strcpy(selected[i++], retpath);                 }
                                        }
                                }
                                done = true;
                                chgdir = true;
                        } else {
                                redraw = true;
if(!strcmp(cur->filename,".."))                         {
for(i=strlen(dir)-2; i > 0; i--)
if(dir[i] == '\\' || dir[i] == ':')
        break;                                          dir[i+1] = 0;
_makepath(path,drive,dir,file,ext);
                                        }
                                        else
                                        {
strcat(dir,cur->filename);
strcat(dir,cur->ext);
_makepath(path,drive,dir,file,ext);
                                        }
                                }
                                break;
                        case TAB:
                                if(win_getline("새로운  경로를
입력하십시오", path, ENGONLY, 60) == CR)
                                {
                                        if(!access(path,0))
                                        {
strcpy(retpath,path);
```

```
donc = true;

true;

13 && mou_ax > 8 && mou_ax < 72)

- 3)
if(cur->next)
cur = cur->next;
offset++;

- 3)
if(cur->prev)
cur = cur->prev;
offset--;

<= 66 && mou_y > 2 && mou_y < 13)

ungetxch(UPARROW);

ungetxch(DOWNARROW);

(int)((mou_py - 48) * (filecount)/ 128.);
= 0;
root;
curpos && pos < filecount - 10)
cur = top = top ->next;
pos++;
curpos)
while(pos+offset < curpos)
cur = cur->next;
offset++;
                                chgdir      =
                        }
                        else
                                chgdir      =
                }
                redraw = true;
                break;
        case ESC:
                chgdir = true;
                done = true;
                break;
        case MOUKEY:
                if(mou_y >=3 && mou_y <
                {
                        while(offset < mou_y
                                {
                                }
                        while(offset > mou_y
                                {
                                }
                        redraw = true;
                } else
                if(mou_x >= 65 && mou_x
                {
                        if(mou_y == 3)
                        else
                        if(mou_y == 12)
                        else
                        {
                                curpos      =
                                offset = pos
                                cur = top =
                                while(pos <
                                {
                                }
                                if(pos      !=
                                {
                                }
                        }
                        redraw = true;
```

```
                                                                        } else
!= -1)                                                                  if((i=  process_button(btn,  4))

                                                                                ungetxch(btnkey[i]);
                                                                        else
&& mou_x < 65)                                                          if(mou_y == 1 && mou_x > 0

                                                                                ungetxch(TAB);
                                                                        break;
                                                        }
                                                }
                                                freedir(root);
                                                root = NULL;
                                        }
                                        pophanmode();
                                        for(i=0; i<4; i++)              close_button(&btn[i]);
                                        close_window(&dirboxwin);
                                }
                                restorescreen();
                        }
//              } else
//                      *retpath = 0;
                return retpath;
}

int pputchar(char c)
{
        int status, errflag = 0;
        if(biosprint(2, c, 0) & 0x29)
        {
                openmsg("프린터 없거나 Off-Line입니다");
                sound(1000);
                delay(30);
                sound(500);
                delay(50);
                nosound();
                delay(500);
                sleep(1);
                closemsg();
                return 3;
        }
        status = biosprint(0,c, 0);

        if (status & 0x0001) {
                puterror("프린터의 응답이 없습니다");
                errflag |= 0x01;
        } else
        if (status & 0x0008) {
                puterror("입/출력 오류입니다");
                errflag |= 0x02;
        } else
        if (status & 0x0020) {
                puterror("프린터에 종이가 없습니다");
                errflag |= 0x20;
        }
        return errflag;
}

int pputs(char *s)
{
        int state;

        while(*s) {
                if(bioskey(1))
                        if(hgetch() == ESC)
                        {
                                openmsg("사용자에 의해 인쇄가 취소되었습니다");
                                sleep(1);
                                closemsg();
                                state = -1;
                                break;
                        }
                state = pputchar(*s++);
                if(state)
                        break;
        }
        return state;
```

```
}


int pprintf(char *fmt, ...)
{
        va_list argptr;
        char buf[1024];
        int ret=0;
        va_start(argptr, fmt);
        vsprintf(buf, fmt, argptr);
        va_end(argptr);
        ret = pputs(buf);
        if(ret)
                return -1;
        return 0;
}


char *expaned_tab(char *str, int tabsize)
{
        char buf[1024], *ptr, i;
        int j = 0;
        ptr = str;
        for(;*str && j < 1023;str++)
        {
                if(*str == '\t')
                        for(i = 0; i < tabsize; i++)
                                buf[j++] = ' ';
                else
                        buf[j++] = *str;
        }
        buf[j] = 0;
        strcpy(ptr, buf);
        return ptr;
}

void draw_v_scrollbar(int x, int y, int len)
{
  int i;
  ctrlputspxy(x, y,"^C3^R^A▲^a^r");
  for(i = 1; i <= len-2; i++)
    hputspxy(x, y+16*i, "■");
  ctrlputspxy(x, y + 16 * i,"^R^A▼^a^r^CF");
}

void draw_h_scrollbar(int x, int y, int len)
{
  int i;
  ctrlputspxy(x, y,"^C3^R^AЦ^a^r");
        for(i = 1; i <= len-2; i++)
    hputspxy(x+16*i, y, "■");
  ctrlputspxy(x+16*i, y,"^R^AШ^a^r^CF");
}

void draw_cursor_scrollbar(int x, int y, int mode)
{
        if(mode)
                ctrlputspxy(x, y,"^A^R■^r^a");
        else
                ctrlputspxy(x, y,"^C3■^CF");
}


int view_file(char *fileviewermsg,char *viewfilename, int gotoline)
{
        mywindow_t win;
        static char far buf[1024];
        long far *lines, ptr;
        int end = 10000;
        FILE *fp;
        int bottom, cursor, left, key, i, done = 0, curposv, curposh, view=false;
        uchar *btnmsg[6] = {"▲","▼","Ц","Ш", "인쇄 F5", "끝 ESC"}, *p, *chk;
        button_t btn[6];
        int btnkey[6] = { UPARROW, DOWNARROW, LEFTARROW, RIGHTARROW, F5, ESC};

        if((lines = farmalloc(10000l * 4l)) == NULL)
                return false;
```

```
        if( (fp=fopen(viewfilename, "rb")) == NULL)
        {
                puterror("화일이 없습니다");
                return false;
        }

        setmywindow(&win, 25, 3, 104, 29, WHITE, BLACK, DARKGRAY, BLUE,
                                                        WHITE, WHITE, GREEN, false, true,
fileviewermsg, false);

        pushcursor();
        savescreen();
        if(open_window(&win)) {
                for(i = 0 ; i < 4; i++) {
                        set_button(&btn[i], 26 + 7*i, 27, 33 + 7*i, 29,

LIGHTCYAN, BLACK, CYAN, LIGHTGREEN, BLACK,
                                                                        false, false,
btnmsg[i]);
                        open_button(&btn[i]);
                }
                for(i = 4 ; i < 6; i++) {
                        set_button(&btn[i], 54 + 25*(i-4), 27, 79+ 25*(i-4), 29,

LIGHTCYAN, BLACK, CYAN, LIGHTGREEN, BLACK,
                                                                        false, false
btnmsg[i]);
                        open_button(&btn[i]);
                }

                // READ FILE
                for(bottom = 0; bottom < gotoline - 1; bottom++)
                        fgets(buf, 1024, fp);

                bottom = 0;
                *(lines+bottom++) = ftell(fp);
                for(;bottom < 10000 && bottom < end && fgets(buf, 1024, fp); )
                {
                        *(lines+bottom++) = ftell(fp);
                }
                view = true;

                draw_v_scrollbar(608, 0, 21);
                draw_h_scrollbar(0, 336, 38);

                left = 0;
                cursor = 0;
                if(!view)
                        puterror("해당 사항이 없습니다.");
                while(!done && view)
                {
                        for(i = 0; i < 21 && !bioskey(1); i++)
                        {
                                buf[0] = 0;
                                if(i+cursor < bottom-1)
                                {
                                        fseek(fp, *(lines+i+cursor), SEEK_SET);
                                        if(fgets(buf, 1024, fp))
                                        {
                                                p = &buf[strlen(buf)];
                                                while(*--p < ' ') *p = 0;
                                                expaned_tab(buf, 8);
                                        }
                                        else
                                                buf[0] = 0;
                                }
                                if(i + cursor < bottom && strlen(buf) > left)
                                {
                                        if(ishangul2nd(buf, left))
                                                hprintfxy(1, i+1, " %-75.75s", buf+left+1);
                                        else
                                                hprintfxy(1, i+1, "%-76.76s", buf+left);
                                }
                                else
                                        hprintfxy(1, i+1, "%-76.76s", "");
                        }
```

```
                        _showcursor = false;
                        if(bottom-21 > 0)
                                curposv = (int)(288. / (float)(bottom-21) * (float)(cursor))+16;
                        else
                                curposv = 16;


                        draw_cursor_scrollbar(608, curposv, 1);
//                      ctrlputspxy(608,curposv,"^A^R■^r^a");

                        curposh = (int)(560. / (float)(940.) * (float)(left))+16;


                        draw_cursor_scrollbar(curposh, 336, 1);
//                      ctrlputspxy(curposh, 336, "^A^R■^r^a");

                        key = hgetch();
                        if(key == UPARROW||key == DOWNARROW||key == PGUPKEY
                          ||key  ==  PGDNKEY||key  ==  HOMEKEY||key  ==  ENDKEY||key  =
MOUKEY)
                                draw_cursor_scrollbar(608, curposv, 0);
//                              ctrlputspxy(608, curposv,"^C3■^CF");
                        if(key == CTRL_LEFT || key == CTRL_RIGHT ||
                                 key  ==  LEFTARROW  ||  key  ==  RIGHTARROW  ||  key  =
MOUKEY)
                                draw_cursor_scrollbar(curposh, 336, 0);
//                              ctrlputspxy(curposh,336, "^C3■^CF");

                        switch(key)
                        {
                                case ESC:
                                        done = 1;
                                        break;
                                case F5:
                                        for(i = 0; i < bottom; i++)
                                        {
                                                buf[0] = 0;
                                                fseek(fp, *(lines+i), SEEK_SET);
                                                if(fgets(buf, 1024, fp))
                                                {
                                                        expaned_tab(buf, 8);
                                                        if(pputs(buf))
                                                                break;
                                                }
                                                                sprintf(buf,"인쇄중...%4.1f%%
(float)(i+1)/(float)(bottom)*100.);
                                                status_help(buf);
                                        }
                                        pputs("\f\r");
                                        status_help("");
                                        //print file;
                                        break;
                                case LEFTARROW:
                                        if(left > 0)
                                                left--;
                                        break;
                                case RIGHTARROW:
                                        if(left < 940)
                                                left++;
                                        break;
                                case CTRL_LEFT:
                                        if(left > 76)
                                                left -= 77;
                                        else
                                                left = 0;
                                        break;
                                case CTRL_RIGHT:
                                        if(left < 864)
                                                        left += 77;
                                        else
                                                        left = 940;
                                        break;
                                case UPARROW:
                                        if(cursor > 0)
                                                cursor--;
                                        break;
                                case DOWNARROW:
```

```
                                        if(cursor + 21 < bottom)
                                                cursor++;
                                        break;
                                case PGDNKEY:
                                        if(cursor + 42 < bottom)
                                                cursor += 22;
                                        else
                                        {
                                                cursor = bottom - 21;
                                                if(cursor < 0)
                                                        cursor = 0;
                                        }
                                        break;
                                case PGUPKEY:
                                        if(cursor - 21 > 0)
                                                cursor -= 21;
                                        else
                                                cursor = 0;
                                        break;
                                case HOMEKEY:
                                        cursor = 0;
                                        break;
                                case ENDKEY:
                                        cursor = bottom - 21;
                                        if(cursor < 0)
                                                        cursor = 0;
                                        break;
                                case MOUKEY:
                                        if(mou_x == 77 || mou_x == 78) {
                                                if(mou_y == 1) {
                                                 ungetxch(UPARROW);
                                                } else if(mou_y == 21) {
                                                 ungetxch(DOWNARROW);
                                                }       else if(mou_y > 1 && mou_y < 21) {
                                                        if(bottom > 21)
                                                                cursor = (float)(bottom - 21)/
304. * (float)(mou_py-16);
                                                }
                                        } else if( mou_y == 22)
                                        {
                                                if(mou_x == 1 || mou_x == 2)
                                                        ungetxch(LEFTARROW);
                                                else if(mou_x == 75 || mou_x == 76)
                                                        ungetxch(RIGHTARROW);
                                                else if(mou_x > 2 && mou_x < 75)
                                                          left   =   (float)940.   /   576.   *
(float)(mou_px-16);
                                        } else  if((i= process_button(btn, 8)) != -1)
                                                        ungetxch(btnkey[i]);
                                        break;

                                }
                        }
                        farfree(lines);
                        for(i = 0; i < 6; i++)
                                close_button(&btn[i]);

                        close_window(&win);
                }
                fclose(fp);
                popcursor();
                //
                restorescreen();
                //
                return true;
}


int copyfile(char *from, char *to)
{
        unsigned char *p;
        int size;
        FILE *in, *out;
        if((p=malloc(4096)) == NULL) return false;
        if((in = fopen(from, "rb")) == NULL)
        {
                free(p);
```

```
                return false;
        }
        if((out = fopen(to, "wb")) == NULL)
        {
                free(p);
                fclose(in);
                return false;
        }
        while((size = fread(p, 1, 4096, in)) !=0)
                fwrite(p,1,size, out);
        free(p);
        fclose(in);
        fclose(out);
        return true;
}


/*
int view_file(char *fileviewermsg,char *viewfilename, int gotoline)
{
        mywindow_t win;
        char buf[1024];
        char **lines;
        FILE *fp;
        int bottom, cursor, left, key, i, done = 0, curposv, curposh;
        uchar *btnmsg[6] = {"▲","▼","Ⅱ","Ⅲ", "인쇄 @P", "끝 ESC"};
        button_t btn[6];
        int btnkey[6] = { UPARROW, DOWNARROW, LEFTARROW, RIGHTARROW, ALT_P, ESC};

        if((lines = farmalloc(10000l * 4l)) == NULL)
                return false;

        if( (fp=fopen(viewfilename, "rt")) == NULL)
        {
                puterror("화일이 없습니다");
                return false;
        }
        setmywindow(&win, 25, 3, 104, 29, WHITE, BLACK, DARKGRAY, BLUE,
                                                        WHITE, WHITE, GREEN, false, true
fileviewermsg, false);
        pushcursor();

        savescreen();

        if(open_window(&win)) {
                // READ FILE
                for(bottom = 0; bottom < 10000l && fgets(buf, 1024, fp); bottom++)
                {
                        expaned_tab(buf, 8);
                        if(buf[strlen(buf)-1] == '\n') buf[strlen(buf)-1] = 0;
                        if(buf[strlen(buf)-1] == '\r') buf[strlen(buf)-1] = 0;

                        *(lines+bottom) = (unsigned char far *)_fstrdup(buf);
                }
                for(i = 0 ; i < 4; i++) {
                        set_button(&btn[i], 26 + 7*i, 27, 33 + 7*i, 29,
LIGHTCYAN, BLACK, CYAN, LIGHTGREEN, BLACK,
                                                                false, false,
btnmsg[i]);
                        open_button(&btn[i]);
                }
                for(i = 4 ; i < 6; i++) {
                        set_button(&btn[i], 54 + 25*(i-4), 27, 79+ 25*(i-4), 29,
LIGHTCYAN, BLACK, CYAN, LIGHTGREEN, BLACK,
                                                                false, false,
btnmsg[i]);
                        open_button(&btn[i]);
                }

                ctrlputspxy_v(608,0,"^C3^cF^R^A▲^a^r"

        "■■■■■■■■■■■■■■■■■■■■■■"

        "^R^A▼^a^r^CF^c1");
                ctrlputspxy(0,336,"^C3^cF^R^AⅡ^a^r"
```

```
        "■■■■■■■■■■■■■■■■■■■■"
        "■■■■■■■■■■■■■■■■■■■■"
        "^R^AIII^a^r^CF^c1┘ ");

                left = 0;

                cursor = gotoline - 1;
//              hsetreverse(true);
//              hprintfxy(1,24,"%76s","");
//              hsetreverse(false);
                while(!done)
                {
                        for(i = 0; i < 21 && !bioskey(1); i++)
                                if(i + cursor < bottom && strlen(*(lines+i+cursor)) > left)
                                {
                                        if(ishangul2nd(*(lines+i+cursor), left))
                                                hprintfxy(1,        i+1,        "        %-75.75s
&(*(lines+i+cursor))[left+1]);
                                        else
                                                hprintfxy(1,            i+1,            "%-76.76s
&(*(lines+i+cursor))[left]);
                                }
                                else
                                        hprintfxy(1, i+1, "%-76.76s", "");
//                      hsetreverse(true);
//                      hprintfxy(1,24,"[총 %4d행,  %4d행 %3d열]", bottom, cursor+23, left+1);
//                      hsetreverse(false);
                        _showcursor = false;

                        curposv = (int)(288. / (float)(bottom-21) * (float)(cursor))+16;
                        ctrlputspxy(608,curposv,"^A^R■^r^a");

                        curposh = (int)(560. / (float)(940.) * (float)(left))+16;
                        ctrlputspxy(curposh, 336, "^A^R■^r^a");

                        key = hgetch();

                        if(key == UPARROW||key == DOWNARROW||key == PGUPKEY
                          ||key  ==  PGDNKEY||key  ==  HOMEKEY||key  ==  ENDKEY||key  =
MOUKEY)
                                ctrlputspxy(608, curposv,"^C3■^CF");
                        if(key == CTRL_LEFT || key == CTRL_RIGHT ||
                           key  ==  LEFTARROW  ||  key  ==  RIGHTARROW  ||  key  =
MOUKEY)
                                ctrlputspxy(curposh,336, "^C3■^CF");

                        switch(key)
                        {
                          case ESC:
                                done = 1;
                                break;
                          case ALT_P:
                                for(i = 0; i < bottom; i++)
                                {
                                        if(pprintf("%s\r\n",*(lines+i)))
                                                break;
                                }
                                //print file;
                                break;
                          case LEFTARROW:
                                if(left > 0)
                                        left--;
                                break;
                          case RIGHTARROW:
                                if(left < 940)
                                        left++;
                                break;
                          case CTRL_LEFT:
                                if(left > 76)
                                        left -= 77;
                                else
                                        left = 0;
                                break;
                          case CTRL_RIGHT:
```

```
                                        if(left < 864)
                                                left += 77;
                                        else
                                                left = 940;
                                        break;
                                case UPARROW:
                                        if(cursor > 0)
                                                cursor--;
                                        break;
                                case DOWNARROW:
                                        if(cursor + 21 < bottom)
                                                cursor++;
                                        break;
                                case PGDNKEY:
                                        if(cursor + 42 < bottom)
                                                cursor += 22;
                                        else
                                        {
                                                cursor = bottom - 21;
                                                if(cursor < 0)
                                                        cursor = 0;
                                        }
                                        break;
                                case PGUPKEY:
                                        if(cursor - 21 > 0)
                                                cursor -= 21;
                                        else
                                                cursor = 0;
                                        break;
                                case HOMEKEY:
                                        cursor = 0;
                                        break;
                                case ENDKEY:
                                        cursor = bottom - 21;
                                        if(cursor < 0)
                                                cursor = 0;
                                        break;
                                case MOUKEY:
                                        if(mou_x == 77 || mou_x == 78) {
                                                if(mou_y == 1) {
                                                 ungetxch(UPARROW);
                                                } else if(mou_y == 21) {
                                                 ungetxch(DOWNARROW);
                                                }       else if(mou_y > 1 && mou_y < 21) {
                                                        if(bottom > 21)
                                                                cursor = (float)(bottom - 21)/
304. * (float)(mou_py-16);
                                                }
                                        } else if( mou_y == 22)
                                        {
                                                if(mou_x == 1 || mou_x == 2)
                                                        ungetxch(LEFTARROW);
                                                else if(mou_x == 75 || mou_x == 76)
                                                        ungetxch(RIGHTARROW);
                                                else if(mou_x > 2 && mou_x < 75)
                                                        left    =    (float)940.    /    576.    *
(float)(mou_px-16);
                                        } else  if((i= process_button(btn, 8)) != -1)
                                                        ungetxch(btnkey[i]);
                                        break;
                                }
                        }

                        for(i = 0; i < bottom; i++)
                                if(*(lines+i) != NULL)
                                        farfree(*(lines+i));
                        farfree(lines);
                        for(i = 0; i < 6; i++)
                                close_button(&btn[i]);

                        close_window(&win);
                }
                fclose(fp);
                popcursor();
                //
                restorescreen();
                //
```

```
        return true;
}
*/
```

```
/* ****************************************** *
 *       온실제어 프로그램 version 0.0
 * ****************************************** */

#include <alloc.h>
#include <string.h>
#include <dos.h>

#include "hanlib.h"
#include "mywindow.h"
#include "mystdio.h"

int _maxtempwindowlevel = 20;
extern int disable_disp_sensor;
static stack[10], stacktop = 0;

void push_disp_disable(void)
{
        if(stacktop < 9)  {
                stack[stacktop] = disable_disp_sensor;
                stacktop++;
        }
}

void pop_disp_disable(void)
{
         if(stacktop > 0)
                        disable_disp_sensor = stack[--stacktop];
}


void setmywindow_act(mywindow_t *w, char isactive)
{
 w->isactive = isactive;
}

void rectangle_3d(int px, int py, int width, int depth, int left, int right)
{
 hhline(-px,-py, width, left);
 hhline(-(px+1), -(py+depth), width, right);
 hvline(-px, -(py+1), depth, left);
 hvline(-(px+width), -py, depth, right);
}

int msgbox(int xl, int yt, int xr, int yb,
                                                    int   light_color,   int   dark_color,      int
board_color,
                                                    int glass_color, int msgfrcolor, int msgbkcolor,
                                                    char *msg, char menuwin)
{
        int x1, y1, x2, y2, length, depth;
        x1 = ax2apx(xl); y1 = ay2apy(yt);
        x2 = ax2apx(xr)+6; y2 = ay2apy(yb)+14;
        length = x2 - x1 + 1; depth = y2 - y1 + 1;
        if(length < 40)
         return false;
        hsolidbarpxy(-x1,-y1,x2,y2,board_color);
        rectangle_3d(x1, y1, length, depth, light_color, dark_color);
        hsolidbarpxy(-(x1+24), -(y1+4), x1+length-5, y1+20, msgbkcolor);
        rectangle_3d(x1+4, y1+4, 15, 17, light_color, dark_color);
        rectangle_3d(x1+6, y1+6, 6, 6, dark_color, light_color);
        rectangle_3d(x1+4+20, y1+4, length-4*2-20, 17, light_color, dark_color);
        rectangle_3d(x1+4, y1+24+2, length-4*2, depth-24-4*2, dark_color, light_color);
        hsolidbarpxy(-(x1+5), -(y1+27), x2-4, y2-6, glass_color);
        if(menuwin) {
         hhline(-(x1+4),-(y1+44),length-8, light_color);
         hhline(-(x1+3),-(y1+45),length-6, glass_color);
         hhline(-(x1+4),-(y1+46),length-8, dark_color);
        }
        pushcolor();
        hsetbkcolor(msgbkcolor);
        hsetcolor(msgfrcolor);
        ctrlputspxy(-(x1+8+length / 2 - 8 * strlen(msg)/2+4),-(y1+ 5),msg);
        popcolor();
        return true;
}
```

```
void setmenuwindow(menuwindow_t *mw, int lx, int ly, int rx, int ry,
                                                        int lc, int bc, int fc)
{
 mw->lx = lx;
 mw->ly = ly;
 mw->rx = rx;
 mw->ry = ry;
 mw->line_c = lc;
 mw->board_c = bc;
 mw->fore_c = fc;
 mw->old_fc = mw->old_bc = 0;
 mw->bkgbuf = NULL;
}

int open_menuwindow(menuwindow_t *mw)
{
 int x1, y1, x2, y2, length, depth;
 long imgsize;
 x1 = ax2apx(mw->lx); y1 = ay2apy(mw->ly);
 x2 = ax2apx(mw->rx)+6; y2 = ay2apy(mw->ry)+14;
 length = x2 - x1 + 1; depth = y2 - y1 + 1;
 if(mw == NULL)
         return false;
 imgsize = htextsize(-mw->lx, -mw->ly, mw->rx, mw->ry);
 if((mw->bkgbuf = (char far *)farmalloc(imgsize)) == NULL) {
          return false;
 }
 hgettext(-mw->lx, -mw->ly, mw->rx, mw->ry, mw->bkgbuf);
 mw->old_fc = hgetcolor();
 mw->old_bc = hgetbkcolor();
 hsolidbarpxy(-(x1+8), -(y1+16), x2, y2-8, DARKGRAY);
 hsolidbarpxy(-(x1+6), -(y1+14), x2-6, y2-14, mw->board_c);
 rectangle_3d(x1+6, y1+14, length-12, depth-28, mw->line_c, mw->line_c);
 rectangle_3d(x1+5, y1+13, length-10, depth-26, mw->line_c, mw->line_c);
 hsetcolor(mw->fore_c);
 hsetbkcolor(mw->board_c);
 if(mw->rx > 11 && mw->ry > 3) {
          push_disp_disable();
          disable_disp_sensor = true;
 }


 return true;
}

void close_menuwindow(menuwindow_t *mw)
{
 if(mw->bkgbuf) {
         hputtext(-mw->lx, -mw->ly, mw->rx, mw->ry, mw->bkgbuf);
         farfree(mw->bkgbuf);
         mw->bkgbuf = NULL;
         hsetcolor(mw->old_fc);
         hsetbkcolor(mw->old_bc);
         if(mw->rx > 11 && mw->ry > 3)
                 pop_disp_disable();
 }
}

void setmywindow(mywindow_t *w, char lx, char ly, char rx, char ry,
                                                        char lc, char dc, char bc,
char gc, char tc, char tfc, char tbc,
                                                        char issave, char isact, char
*msg, char menu)
{
 memset(w, 0, sizeof(mywindow_t));
 w->lx = lx;
 w->ly = ly;
 w->rx = rx;
 w->ry = ry;
 w->light_c = lc;
 w->dark_c = dc;
 w->board_c = bc;
 w->glass_c = gc;
 w->text_c = tc;
 w->title_frc = tfc;
 w->title_bkc = tbc;
 w->issavebkg = issave;
```

```
 setmywindow_act(w, isact);
 strcpy(w->msg, msg);
 w->menu = menu;
}

int open_window(mywindow_t *w)
{
 if(w == NULL)
        return false;
 if(w->issavebkg && !w->isinit) {
        if((w->winbuf = (char far*)farmalloc(htextsize(-w->lx, -w->ly, w->rx, w->ry))) == NULL) {
         w->isinit = false;
         return false;
        }
        hgettext(-w->lx, -w->ly, w->rx, w->ry, w->winbuf);
 }
 w->oldfrc = hgetcolor();
 w->oldbkc = hgetbkcolor();
 if(!msgbox(w->lx, w->ly, w->rx, w->ry,
                                                        w->light_c,     w->dark_c,      w->board_c,
w->glass_c,
                                                        w->title_frc, w->title_bkc,
                                                        w->msg, (w->menu)? 1:0))
 {
        if(w->issavebkg)
         farfree(w->winbuf);
        w->winbuf = NULL;
        w->isinit = false;
        hsetcolor(w->oldfrc);
        hsetbkcolor(w->oldbkc);
        return false;
 }
 w->isinit = true;
 if(!w->menu)
        htempwindow(w->lx+1, w->ly+2, w->rx-1, w->ry-1);
 else {
        // display top_down menu
        htempwindow(w->lx+1, w->ly+3, w->rx-1, w->ry-1);
 }
 hsetcolor(w->text_c);
 hsetbkcolor(w->glass_c);
 if(w->rx > 11 && w->ry > 3) {
         push_disp_disable();
         disable_disp_sensor = true;
 }
 return true;
}

void close_window(mywindow_t *w)
{
 if(w == NULL || !w->isinit)
        return;
 if(w->issavebkg) {
        hputtext(-w->lx, -w->ly, w->rx, w->ry, w->winbuf);
        farfree(w->winbuf);
        w->winbuf = NULL;
        w->isinit = false;
 }
 hlastwindow();
 hsetcolor(w->oldfrc);
 hsetbkcolor(w->oldbkc);
 if(w->rx > 11 && w->ry > 3)
                pop_disp_disable();
}

void set_button(button_t *b, char lx, char ly, char rx, char ry,
                                                        char lc, char dc, char bc,
char stc, char utc,
                                                        char issave, char isactive,
char *text)
{
 memset(b, 0, sizeof(button_t));
 b->lx = lx;
 b->ly = ly;
 b->rx = rx;
 b->ry = ry;
 b->light_c = lc;
```

```
 b->dark_c = dc;
 b->board_c = bc;
 b->s_text_c = stc;
 b->u_text_c = utc;
 b->issavebkg = issave;
 b->isactive = isactive;
 strcpy(b->text, text);
}

int open_button(button_t *b)
{
 if(b == NULL)
        return false;
 if(b->issavebkg && !b->isinit) {
        if((b->button = (char far*)farmalloc(htextsize(-b->lx, -b->ly, b->rx, b->ry))) == NULL) {
         b->isinit = false;
         return false;
        }
        hgettext(-b->lx, -b->ly, b->rx, b->ry, b->button);
 }
 b->oldfrc = hgetcolor();
 b->oldbkc = hgetbkcolor();
 hsolidbarpxy(-(ax2apx(b->lx)+1), -(ay2apy(b->ly)+1),
                                                                ax2apx(b->rx)-2,
ay2apy(b->ry)-2,b->board_c);
 disp_button(b, b->isactive);
 b->isinit = true;
 hsetcolor(b->oldfrc);
 hsetbkcolor(b->oldbkc);
 return true;
}

void close_button(button_t *b)
{
 if(b == NULL || !b->isinit)
        return;
 if(b->issavebkg) {
        hputtext(-b->lx, -b->ly, b->rx, b->ry, b->button);
        farfree(b->button);
        b->button = NULL;
        b->isinit = false;
 }
}

void disp_button(button_t *b, char active)
{
 b->isactive = active;
 pushcolor();
 if(b->isactive)
        hsetcolor(b->s_text_c);
 else
        hsetcolor(b->u_text_c);
 hsetbkcolor(b->board_c);
 hputspxy(-(ax2apx(b->lx)+(b->rx - b->lx) * 4 - strlen(b->text) * 4),
                                        -(ay2apy(b->ly)+(b->ry - b->ly) * 8 - 8), b->text);
 if(b->isactive) {
        rectangle_3d(ax2apx(b->lx), ay2apy(b->ly),
                                                                8*(b->rx-b->lx)-1
16*(b->ry-b->ly)-1,
                                                        b->dark_c,b->light_c);
 } else {
        rectangle_3d(ax2apx(b->lx), ay2apy(b->ly),
                                                                8*(b->rx-b->lx)-1
16*(b->ry-b->ly)-1,
                                                        b->light_c, b->dark_c);
 }
 popcolor();
}


int process_button(button_t *btn, int button_num)
{
 int i;
 for(i = 0; i < button_num; i++) {
        if((btn[i].lx <= mou_ax && btn[i].rx > mou_ax)
                &&(btn[i].ly <= mou_ay && btn[i].ry > mou_ay))
        {
```

```
            disp_button(&btn[i], true);
            delay(100);
            disp_button(&btn[i], false);
            return i;
          }
  }
  return -1;
}
```

```
// ***************************************************************
//                                          농업과 컴퓨터 project
//
//          온실자동화 패키쥐 프로그램 #1
//
// ***************************************************************

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <dos.h>
#include <bios.h>
#include <mem.h>
#include <string.h>
#include <math.h>

#include "hanlib.h"
#include "mystdio.h"
//#include "asynch.h"
#include "mux.h"
#include "ctrldata.h"
#include "pc2ctrl.h"
#include "setctrl.h"
#include "setenv.h"

void getsunrise(struct date d, int *sh, int *sm, int *eh, int *em);

//#define port_id (osjk_env.c_haus + 1)
int port_id;
extern int israining;

//
ctrl_data_t _ctrldata_;
status_flag_t flag;

int upload_status_flag(status_flag_t *flag);
int dnload_status_flag(status_flag_t flag);
void log_sensor_data(int haus, int pos);  // 센서 자료 로그
void log_ctrl_data(int haus);

weathersensor_t wsensor = { 1, 0, 0, 64, 1, 0, 0};


#define COM1 0
#define COM2 1

/* Base addresses of serial ports */

#define COM1BASE 0x03f8
#define COM2BASE 0x02f8

#define COMBASE ((comport == COM1) ? COM1BASE : COM2BASE)

/* Registers */

#define THR (COMBASE + 0)  /* Transmit Holding Register         */
#define RBR (COMBASE + 0)  /* Receive Buffer Register           */
#define IER (COMBASE + 1)  /* Interrupt Enable Register         */
#define IIR (COMBASE + 2)  /* Interrupt Identification Register */
#define LCR (COMBASE + 3)  /* Line Control Register             */
#define MCR (COMBASE + 4)  /* Modem Control Register            */
#define LSR (COMBASE + 5)  /* Line Status Register              */
#define MSR (COMBASE + 6)  /* Modem Status Register             */

/* Parameters to bioscom function */

#define DATABIT7    0x02  /* Data bit */
#define DATABIT8    0x03

#define STOPBIT1    0x00  /* Stop bit */
#define STOPBIT2    0x04

#define NOPARITY    0x00  /* Parity */
#define ODDPARITY   0x08
#define EVENPARITY  0x18

#define BAUD1200    0x80   /* Baud rate */
```

```
#define BAUD2400    0xa0
#define BAUD4800    0xc0
#define BAUD9600    0xe0

/* 8259 PIC(Programmable Interrupt Controller) */

#define IMR  0x21  /* I/O address of OCW1(IMR) of 8259 PIC  */
#define OCW2 0x20  /* I/O address of OCW2 of 8259 PIC       */
                                                    /* OCW: Operation
Command Word          */
                                                    /* IMR: Interrupt
Mask Register         */

#define MASKON  0xe7  /* Mask IRQ3/IRQ4 on  -> IMR */
#define MASKOFF 0x18  /* Mask IRQ3/IRQ4 off -> IMR */

#define EOI 0x20  /* Non-specific End of Interrupt command -> OCW2 */

/* Interrupt Request Numbers */

#define IRQ3 0x0b  /* of COM2 */
#define IRQ4 0x0c  /* of COM1 */

#define IRQNUM ((comport == COM1) ? IRQ4 : IRQ3)

/* Miscellaneos */

#define BUFSIZE 0x4000  /* Size of comm buffer */

#define BUFFEREMPTY (-1)  /* Buffer empty */


int comport = COM2;  /* Default COM port, alterable by user */

void interrupt (*oldvect)(void);   /* Old interrupt vector */
byte far commbuf[BUFSIZE];         /* Comm buffer */
static int bufptr0 = 0, bufptr1 = 0;  /* Pointers to comm buffer */


static void interrupt com_isr(void)
{
                bufptr1 %= BUFSIZE;
                commbuf[bufptr1++] = inportb(RBR);
                outportb(OCW2, EOI);
}

void com_init(int comport)
{
                bioscom(0, DATABIT8 | STOPBIT1 | NOPARITY | BAUD2400, comport);

                outportb(MCR, 0x0b);
                outportb(IER, 0x01);

                oldvect = getvect(IRQNUM);
                setvect(IRQNUM, com_isr);

                outportb(IMR, inportb(IMR) & MASKON);
}

void com_close(void)
{
                outportb(MCR, 0x00);
                outportb(IER, 0x00);

                outportb(IMR, inportb(IMR) | MASKOFF);
                setvect(IRQNUM, oldvect);
}

int com_getc(void)
{
                if (bufptr0 == bufptr1)
                        return BUFFEREMPTY;
                bufptr0 %= BUFSIZE;

                return commbuf[bufptr0++];
}
```

```
void com_putc(int c)
{
                    while (!(inportb(LSR) & 0x20)) ;
                    outportb(THR, (byte)c);
}

void com_puts(char *s)
{
                    while (*s) com_putc(*s++);
}

void clear_com_buffer(void)
{
          bufptr0 = bufptr1 = 0;
}

void open_weatherlink(void)
{
                    int c, y;
                    unsigned char data[147] =
                    {
0x0D,0x57,0x52,0x44,0x84,0x34,0x0D,0x57,0x52,0x44,0x84,0x3C,0x0D,0x57,0x52,0x44
,0x64,0x44,0x0D,0x57,0x52,0x44,0x44,0x9A,0x0D,0x57,0x52,0x44,0x84,0x9E,0x0D,0x57
,0x52,0x44,0x64,0xA6,0x0D,0x57,0x52,0x44,0x82,0x8E,0x0D,0x57,0x52,0x44,0x82,0x96
,0x0D,0x57,0x52,0x44,0x62,0x9E,0x0D,0x57,0x52,0x44,0x84,0x5A,0x0D,0x57,0x52,0x44
,0x84,0x62,0x0D,0x57,0x52,0x44,0x64,0x6A,0x0D,0x57,0x52,0x44,0x44,0x82,0x0D,0x57
,0x52,0x44,0x84,0x86,0x0D,0x57,0x52,0x44,0x64,0x8E,0x0D,0x57,0x52,0x44,0x22,0x60
,0x0D,0x57,0x52,0x44,0x42,0x64,0x0D,0x57,0x52,0x44,0x32,0x68,0x0D,0x57,0x52,0x44
,0x42,0xAC,0x0D,0x57,0x52,0x44,0x42,0xB0,0x0D,0x57,0x52,0x44,0x32,0xB4,0x0D,0x52
,0x52,0x44,0x01,0x2A,0x03,0x0D,0x57,0x52,0x44,0x84,0xCE,0x0D,0x4C,0x4F,0x4F,0x50
                              ,0xE7,0xFF,0x0D
                    };
//                  mou_wait(true);
                    com_init(comport);
                    clear_com_buffer();
                    printf("Testing .");
                    com_putc(13);        delay(500);
                    printf(".");
                    com_putc(13);        delay(500);
                    printf(".");
                    com_putc(13);        delay(500);
                    printf(".");
                    delay(500);
                    printf(".");
                    delay(500);
                    printf(".");
                    delay(500);
                    printf(".");
                    delay(500);
                    printf(".");
                    if((c= com_getc()) == '!')
                    {
                              printf("\nWeather Link was detected on COM%d.\nNow initializing
comport+1);
                              y = wherey();
                              gotoxy(1,y);
                              cputs("쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓쌓");
                              for(c=0; c<140;c++,delay(100))
                              {
                                        com_putc(data[c]);
                                        gotoxy((c) / 140. * 40.+1, y);
                                        cputs(" );
                                        gotoxy(45, y);
                                        printf("%3.0f %% completed.", (c+1.) / 140. * 100.);
                                        if(data[c] == 13) delay(500);
                              }
                    } else
                    {
```

```
                        puts("No \"Weather Link\" was Connected !!!")
                        sound(2000);
                        delay(40);
                        sound(4000);
                        delay(60);
                        sound(3000);
                        delay(40);
                        nosound();
                }
//              get_loop_data();
                clear_com_buffer();
//              mou_wait(false);
}

void close_weatherlink(void)
{
        com_close();
}

void loop(void)
{
        unsigned char data[] = {0x4C,0x4F,0x4F,0x50,0xE7,0xFF,0x0D}, i;
        mou_wait(true);
        status_help("기상스테이션에게 자료를 요청합니다");
        clear_com_buffer();
        for(i=0; i<7;i++, delay(100))
        {
                com_putc(data[i]);
                if(data[i] == 13) delay(200);
        }
        mou_wait(false);
        status_help("");
}


int get_loop_data(void)
{
        int c, i, ok = false;
        while(!ok)
        {
                do
                        c = com_getc();
                while(c != 6 && c != BUFFEREMPTY);
                if(c== BUFFEREMPTY)
                        break;
                c = com_getc();
                if(c == 1)
                {
                        for(i = 1; i < 18; i++)
                                wsensor.dat[i] = (uchar)com_getc();
                        if(i == 18 && wsensor.dat[5] < 100 )
                                ok = true;
                }
                c = com_getc();
                if(c == 1)
                {
                        for(i = 1; i < 18; i++)
                                wsensor.dat[i] = (uchar)com_getc();
                        if(i == 18)
                                ok = true;
                }
        }
//      clear_com_buffer();
        return true;
}

int await(unsigned char code)
{
 unsigned char ch;
 int i, length;
 mou_wait(true);
 for(i = 1000; i > 0; i--) {
        delay(1);
        length = 1;
        ch = 0;
        if(!readcom(port_id, &ch, &length)) {
         if(ch == code)
```

```
                    return true;
            }
    }
    mou_wait(false);
    openmsg("제어기 응답이 없습니다.");
    delay(500);
    closemsg();
// puterror("제어기 응답이 없습니다.");
    return false;
}

int init_mux(void)
{
        int ret, i;
        for(i = 1; i <= MAXPORT; i++) {
                ret = setcom(i, 0x03, 9600);
                if(ret & 0x40) {
                        puterror("멀티포트가 설치되지 않았습니다");
                        break;
                }
                if(ret & 0x80) {
                        puterror("멀티포트 드라이버 화일이 설치되지 않았습니다");
                        break;
                }
                if(ret & 0x20) {
                        puterror("포트가 비활성화 되어 있습니다");
                        break;
                }
                clearcom(i, 0x01);
        }
        return ret;
}


//
// 현재 선택된 온실의 제어자료를 작기에 따라 제어용 자료형태에 넣는다.
//
int load2ctrldata(int haus)
{
        FILE *fin;
        long offset;
        int i, j;
        ctrldata_t ctl;
        mou_wait(true);
        fin = fopen(osjk_env.haus[haus].ctrldata_fn, "r+b");
        if(fin != NULL) {
                if(ctrldata_header[haus].elapsed < ctrldata_header[haus].duration)
                        offset      =      sizeof(ctrldata_header_t)      +      sizeof(ctrldata_t)      *
(ctrldata_header[haus].elapsed - 1);
                else
                        offset      =      sizeof(ctrldata_header_t)      +      sizeof(ctrldata_t)      *
(ctrldata_header[haus].duration - 1);
                fseek(fin, offset, SEEK_SET);
                fread(&ctl,sizeof(ctrldata_t), 1, fin);
                fclose(fin);
                for(i=0; i < 4; i++)
                        for(j=0; j < 48; j++)
                                _ctrldata_.step[i][j] = ctl.ctrlitem[i].step[j];
                memcpy(_ctrldata_.error, ctrldata_header[haus].error, sizeof _ctrldata_.error);
                memcpy(_ctrldata_.mul, ctrldata_header[haus].mul, sizeof _ctrldata_.mul);
                memcpy(_ctrldata_.unit, ctrldata_header[haus].unit, sizeof _ctrldata_.unit);
                memcpy(_ctrldata_.can_open_wind,      ctrldata_header[haus].can_open_wind,      sizeof
_ctrldata_.can_open_wind);
                memcpy(&_ctrldata_.can_open_rain,      &ctrldata_header[haus].can_open_rain,      sizeof
_ctrldata_.can_open_rain);
                _ctrldata_.wind_speed = ctrldata_header[haus].wind_speed;

                mou_wait(false);
                return true;
        }
        mou_wait(false);
        return false;
}

void auto_dnload_ctrldata(void)
```

```
{
        static struct date d[7];
        struct date td;
        static count[7];
        char *rc = "WWRC\r";
        int i, j;

        getdate(&td);
        for(i = 0; i < 7; i++)
        {
                if(!ctrldata_header[i].started || ctrldata_header[i].stoped)
                        continue;
                port_id = i + 1;

                if(td.da_day != d[i].da_day)
                {
                        for(j = 0; j < 5; j++, delay(150))
                                com_putc(rc[j]);

                        if( count[i] > 1 || dnload_ctrldata(i) )
                        {
                                d[i] = td;
                                count[i] = 0;
                        }
                        else
                                count[i]++;
                }
        }
}

//
// commander = pc
//
int dnload_ctrldata(int haus)
{
 int i, item;
 char buf[45];

 if(!ctrldata_header[haus].started || ctrldata_header[haus].stoped)
                return false;

 if(!load2ctrldata(haus))
        return false;

 mou_wait(true);
 status_help("제어기에게 제어자료를 전하고 있습니다");

 if(haus < 4)
 {

        sprintf(buf, "%c%cLD__TEMP", _ESC_, 'L');
        delayedout(haus+1, buf, 10);

        item = 0;
        if(!await(_RS_)) {
                mou_wait(false);
                status_help("");
                return false;
        }
        delay(10);
        sprintf(buf, "%c", _ESC_);
        delayedout(haus+1, buf, 1);

        for(i = 0; i < 48; i++)
        {
                sprintf(buf,"%+04d",_ctrldata_.step[item][i]);
                delayedout(haus+1,buf,4);
        }
        sprintf(buf,"%03d%03d%1d",_ctrldata_.error[item],_ctrldata_.mul[item]

                _ctrldata_.unit[item]);
        delayedout(haus+1, buf, 7);
        sprintf(buf,"%c",_EOF_);
        delayedout(haus+1, buf, 1);

        if(!await(_GS_)) {
                mou_wait(false);
```

```
            status_help("");
            return false;
}

delay(100);

sprintf(buf, "%c%cLD__HUMI", _ESC_, 'H');
delayedout(haus+1, buf, 10);

item = 1;
if(!await(_RS_)) {
            mou_wait(false);
            status_help("");
            return false;
}
delay(10);

sprintf(buf, "%c", _ESC_);
delayedout(haus+1, buf, 1);

for(i = 0; i < 48; i++)
{
            sprintf(buf,"%+04d",_ctrldata_.step[item][i]);
            delayedout(haus+1,buf,4);
}
sprintf(buf,"%03d%03d%1d",_ctrldata_.error[item],_ctrldata_.mul[item],

ctrldata_.unit[item]);
delayedout(haus+1, buf, 7);
sprintf(buf,"%c",_EOF_);
delayedout(haus+1, buf, 1);

if(!await(_GS_)) {
            mou_wait(false);
            status_help("");
            return false;
}

delay(100);

sprintf(buf, "%c%cLD__CO2 ", _ESC_, 'O');
delayedout(haus+1, buf, 10);

item = 2;
if(!await(_RS_)) {
            mou_wait(false);
            status_help("");
            return false;
}
delay(10);

sprintf(buf, "%c", _ESC_);
delayedout(haus+1, buf, 1);

for(i = 0; i < 48; i++)
{
            sprintf(buf,"%+04d",_ctrldata_.step[item][i]);
            delayedout(haus+1,buf,4);
}
sprintf(buf,"%03d%03d%01d",_ctrldata_.error[item],_ctrldata_.mul[item]

,ctrldata_.unit[item]);
delayedout(haus+1, buf, 7);
sprintf(buf,"%c",_EOF_);
delayedout(haus+1, buf, 1);

if(!await(_GS_)) {
            mou_wait(false);
            status_help("");
            return false;
}

delay(100);

sprintf(buf, "%c%cLD_LIGHT", _ESC_, 'G');
delayedout(haus+1, buf, 10);
```

```
        item = 3;
        if(!await(_RS_)) {
                mou_wait(false);
                status_help("");
                return false;
        }
        delay(10);

        sprintf(buf, "%c", _ESC_);
        delayedout(haus+1, buf, 1);

        sprintf(buf,"%+04d",_ctrldata_.step[item][0]);
        delayedout(haus+1,buf,4);
        sprintf(buf,"%03d%03d%1d",_ctrldata_.error[item],_ctrldata_.mul[item],

      _ctrldata_.unit[item]);
        delayedout(haus+1, buf, 7);
        sprintf(buf,"%c",_EOF_);
        delayedout(haus+1, buf, 1);

        if(!await(_GS_)) {
                mou_wait(false);.
                status_help("");
                return false;
        }

        delay(100);

        sprintf(buf, "%c%cLD__WIND", _ESC_, 'N');
        delayedout(haus+1, buf, 10);

        if(!await(_RS_)) {
                mou_wait(false);
                status_help("");
                return false;
        }
        delay(10);

        sprintf(buf, "%c", _ESC_);
        delayedout(haus+1, buf, 1);

        delayedout(haus+1,osjk_env.haus[haus].can_open_wind, 8);
        delayedout(haus+1,&osjk_env.haus[haus].can_open_rain, 1);
        delayedout(haus+1,&osjk_env.haus[haus].wind_speed, 1);

        sprintf(buf,"%c",_EOF_);
        delayedout(haus+1, buf, 1);

        if(!await(_GS_)) {
                mou_wait(false);
                status_help("");
                return false;
        }
} else
{
        //양액
        sprintf(buf, "%c%cLD__HYPH", _ESC_, 'L');
        delayedout(haus+1, buf, 10);

        item = 0;
        if(!await(_RS_)) {
                mou_wait(false);
                status_help("");
                return false;
        }
        delay(10);
        sprintf(buf, "%c", _ESC_);
        delayedout(haus+1, buf, 1);

        sprintf(buf,"%03d%03d%03d%03d%02d%02d%02d%02d",_ctrldata_.step[0][0]

      _ctrldata_.step[0][1],

      _ctrldata_.step[0][2],

      _ctrldata_.step[0][3],
                        _ctrldata_.error[0],
```

```
                        _ctrldata_.error[1],
                        _ctrldata_.error[2]);
        delayedout(haus+1,buf,18); // DO 보함

        for(i = 0; i < 1 /*3*/; i++)
        {
                        for(item = 0; item < 4; item++)
                        {
                                sprintf(buf,"%02d%02d%04d%04d",(_ctrldata_.step[i][4+item]
/ 2),

(_ctrldata_.step[i][4+item] % 2)? 30: 0,

_ctrldata_.step[i][8+item],

_ctrldata_.step[i][12+item]);
                                delayedout(haus+1,buf,12);
                        }
        }
        sprintf(buf,"%c",_EOF_);
        delayedout(haus+1, buf, 1);


        if(!await(_GS_)) {
                mou_wait(false);
                status_help("");
                return false;
        }
 }
 status_help("");
 mou_wait(false);
 return true;
}


int dnload_stationdata(void)
{
 char buf[20], i;
 struct date d;
 int sh, sm, eh, em;
 getdate(&d);
 mou_wait(true);

 status_help("제어기에게 날씨자료를 전하고 있습니다");
 for(i = 0; i < 4; i++)
 {
        if(!ctrldata_header[i].started || ctrldata_header[i].stoped)
                        continue;
        port_id = i + 1;
        sprintf(buf, "%cXLOADST'N",_ESC_);
        delayedout(i+1, buf, 10);

        if(!await(_RS_))
                continue;

        delay(10);
        sprintf(buf, "%c", _ESC_);
        delayedout(i+1, buf, 1);

                                                        // 1   3    3   2   4  = 13
        sprintf(buf, "%c%+03d%+03d%02d%04d",
                                                        dicision_wd(wsensor.sdata.wd)
+ '0',
                                                        (int)(wsensor.sdata.ws*1609.3 /
3600.),
                                                        (int)((wsensor.sdata.out_t      /
10.- 32.)* 5. / 9.),
                                                        (int)wsensor.sdata.out_h,
                                                        wsensor.sdata.light);
        delayedout(i+1, buf, 13);

        // if sunrise and sunset time are download....
        getsunrise(d, &sh, &sm, &eh, &em);
        sprintf(buf,"%02d%02d%02d%02d",sh, sm, eh, em);
        delayedout(i+1, buf, 8);
        //
```

```
            sprintf(buf, "%c", _EOF_);
            delayedout(i+1, buf, 1);
            /*
            if(!await(_GS_)) {
                        mou_wait(false);
                        status_help("");
                        return false;
            } */
 }
 mou_wait(false);
 status_help("");
 return true;
}

/*

//
// commander = pc
//
int current_sensor_status(void)
{
 unsigned char buf[14];
 int i, size, iter;

 mou_wait(true);

 memset(&sint, 0, sizeof sint);

 sprintf(buf,"%c%cSend Sensor", _ESC_, 'S');
 delayedout(port_id, buf, 13);
 if(await(_RS_)) {
            size = 7;
            for(iter = 32767; iter > 0; iter--)
                        if(!readcom(port_id, sint.sdate, &size))
                                    break;
            sint.sdate[7] = 0;

            size = 6;
            for(iter = 32767; iter > 0; iter--)
                        if(!readcom(port_id, sint.stime, &size))
                                    break;

            sint.stime[6] = 0;

            for(i=0; i < MAXSENSOR; i++)
            {
                        if(!await(_RS_))
                                    break;
                        size = 7;
                        *buf=0;
                        for(iter = 32767; iter > 0; iter--)
                                    if(!readcom(port_id, buf, &size))
                                                break;
                        buf[7] = 0;

                        sint.sid[i] = buf[0] - '0';
                        sint.sdata[i] = atoi(&buf[1]);
            }
            if(await(_EOF_))
            {
                        sprintf(buf, "%c", _GS_);
                        delayedout(port_id, buf, 1);
                        mou_wait(false);
                        return true;
            }
 }
 mou_wait(false);
 return false;
}
*/
//
// commander = pc
//
int upload_status_flag(status_flag_t *flag)
{
 char buf[12];
 int size, iter;
```

```
mou_wait(true);

port_id = osjk_env.c_haus + 1;

sprintf(buf, "%c%cRD STATUS",_ESC_, 'R');
delayedout(port_id, buf, 11);
if(!await(_ESC_)) {

        mou_wait(false);
        return false;
}
if(osjk_env.c_haus < 4)
{
        await('C');
        size = 5;
}
else
{
        await('c');
        size = 2;
}
for(iter = 32767; iter > 0; iter--)
                if(!readcom(port_id, flag->sw, &size))
                                                        break;
if(!await(_EOF_)) {
        mou_wait(false);
        return false;
}
sprintf(buf, "%c",_GS_);
delayedout(port_id, buf, 1);
mou_wait(false);
return true;
}

//
// commander = pc
//
int dnload_status_flag(status_flag_t flag)
{
 char buf[12];

 mou_wait(true);

 port_id = osjk_env.c_haus + 1;

 sprintf(buf, "%c%c",_ESC_, 'M');
 delayedout(port_id, buf, 2);
 if(osjk_env.c_haus < 4)
 {
        delayedout(port_id, flag.sw, 5);
        sprintf(buf,"MANUAL%c",_EOF_);
        delayedout(port_id, buf, 7);
 }
 else
 {
        delayedout(port_id, flag.sw, 2);
        sprintf(buf,"_MANUAL%c",_EOF_);
        delayedout(port_id, buf, 8);
 }
 if(await(_GS_)) {
        mou_wait(false);
        return true;
 }
 mou_wait(false);
 return false;
}

//
// commander = pc
//
int set_ctrl_auto(void)
{
 char buf[12];

 mou_wait(true);
```

```
 port_id = osjk_env.c_haus + 1;

 sprintf(buf, "%c%c",_ESC_, 'A');
 delayedout(port_id, buf, 2);
 sprintf(buf,"SET AUTO%c",_EOF_);
 delayedout(port_id, buf, 9);
 if(await(_GS_)) {
  mou_wait(false);
        return true;
 }
 mou_wait(false);
 return false;
}

//
// commander = pc
//
int ctrl_init(void)
{
 char buf[12];
 mou_wait(true);

 port_id = osjk_env.c_haus + 1;

 sprintf(buf,"%c%c",_ESC_, 'I');
 delayedout(port_id, buf, 2);
 sprintf(buf,"SYSRESET%c",_EOF_);
 delayedout(port_id, buf, 9);
 if(await(_GS_)) {
        mou_wait(false);
        return true;
 }
 mou_wait(false);
 return false;
}

int alarm_init(void)
{
 char buf[12];
 mou_wait(true);

 port_id = osjk_env.c_haus + 1;

 sprintf(buf,"%c%c",_ESC_, 'a');
 delayedout(port_id, buf, 2);
 sprintf(buf,"ARMRESET%c",_EOF_);
 delayedout(port_id, buf, 9);
 if(await(_GS_)) {
        mou_wait(false);
        return true;
 }
 mou_wait(false);
 return false;
}

//
// commander = pc
//
int set_ctrl_timer(ctrl_time_t t)
{
 char buf[7], hbu;
 mou_wait(true);
 hbu = osjk_env.c_haus;
 for(osjk_env.c_haus = 0 ; osjk_env.c_haus < 7; osjk_env.c_haus++)
 {
   port_id = osjk_env.c_haus + 1;

   sprintf(buf, "%c%c",_ESC_, 'T');
   delayedout(port_id, buf, 2);

   delayedout(port_id, t.time, 7);

   sprintf(buf,"%c",_EOF_);
   delayedout(port_id, buf, 1);

   await(_GS_);
```

```
 }
 osjk_env.c_haus = hbu;
 mou_wait(false);
 return true;
}

//
// commander = pc
//
int set_ctrl_date(ctrl_date_t d)
{
 char buf[3], hbu;
 mou_wait(true);

 hbu = osjk_env.c_haus;
 for(osjk_env.c_haus = 0 ; osjk_env.c_haus < 7; osjk_env.c_haus++)
 {
   port_id = osjk_env.c_haus + 1;

   sprintf(buf, "%c%c",_ESC_, 'W');
   delayedout(port_id, buf, 2);

   delayedout(port_id, d.date, 7);

   sprintf(buf,"%c",_EOF_);
   delayedout(port_id, buf, 1);

   await(_GS_);
 }
 osjk_env.c_haus = hbu;
 mou_wait(false);
 return true;
}

//
// commander = pc
//
int set_himist_rain(int mist, int rain)
{
 char buf[15];
 int haus;
 mou_wait(true);
 //
 // set rain...
 flag.win_status.rain = rain;

 haus = osjk_env.c_haus;
// osjk_env.c_haus = 1;
// upload_status_flag(&flag);

 for(port_id = 1; port_id <= 4; port_id++)
 {
          sprintf(buf, "%c%c%1dRAINMIST",_ESC_, '!', mist*2+rain)
          delayedout(port_id, buf, 11);
          sprintf(buf,"%c",_EOF_);
          delayedout(port_id, buf, 1);
          await(_GS_);
 }


 port_id = haus + 1;
// osjk_env.c_haus = haus;

// upload_status_flag(&flag);

 mou_wait(false);
 return true;
}


void check_rain(void)
{
          char buf[2]={_GS_,0}, ch;
          int length, p;
          p = 8;
          delayedout(p , buf, 1);
          length = 1;
```

```
        ch = 0;
        readcom(p, &ch, &length);
        if(ch ==_GS_)
                israining = true;
        else
                israining = false;
        flag.win_status.rain = israining;
}


void parse_comm(void)
{
        char ch = 0, buf[10]; msg[80];
        int len, i, haus, j, pos;
        struct time t;
        gettime(&t);
        pos = t.ti_hour * 20 + t.ti_min / 3;

        for(haus = 0; haus < 7; haus++)
        {
                len = 1;

                port_id =  haus + 1;

                if(!ctrldata_header[haus].started || ctrldata_header[haus].stoped)
                        continue;

                if(readcom(haus+1, &ch, &len))
                        continue;


                for(i = 0; i < 1000 ; i++, readcom(haus+1, &ch, &len))
                        if(ch == _ESC_) break;

                if(i == 1000)
                        continue;

                mou_wait(true);

                if(ch == _ESC_)
                {
                        len = 1;
                        for(i = 32767; i > 0; i--)
                                if(!readcom(haus+1, &ch, &len))
                                        break;
                        switch(ch)
                        {
                                case 'c':
                                        status_help("온실 제어기가 보낸 제어자료를 분석합니다")
                                        len = 2;
                                        for(i = 32767; i > 0; i--)
                                                if(!readcom(haus+1, flag.sw, &len))
                                                        break;
                                        if(!await(_EOF_)) {
                                                mou_wait(false);
                                                status_help("");
                                                return;
                                        }
                                        sprintf(buf, "%c",_GS_);
                                        delayedout(haus+1, buf, 1);
                                        watch_ctrl(haus);
                                        log_ctrl_data(haus);
                                        break;
                                case 'C':
                                        sprintf(msg, "%d 제어기가 보낸 제어자료를 분석합니다
haus+1);
                                        status_help(msg);
                                        len = 5;
                                        for(i = 32767; i > 0; i--)
                                                if(!readcom(haus+1, flag.sw, &len))
                                                        break;
                                        if(i == 0) break;
                                        if(!await(_EOF_)) {
                                                mou_wait(false);
                                                status_help("");
                                                return;
```

```
                                        }
                                        sprintf(buf, "%c", _GS_);
                                        delayedout(haus+1, buf, 1);
                                        watch_ctrl(haus);
                                        log_ctrl_data(haus);
                                        break;

                                case 'S':
                                        sprintf(msg, "%d 제어기가 보낸 센서자료를 분석합니다"
haus+1);
                                        status_help(msg);
                                        for(j = 0; j < 8; j++)
                                        {
                                                len = 6;
                                                for(i = 32767; i > 0; i--, delay(10))
                                                        if(!readcom(haus+1, buf, &len))
break;
                                                if(i == 0) break;
                                                buf[6] = 0;
                                                sensordata[haus][j][pos] = atoi(buf);

                                                if(sensordata[haus][3][pos] < 0.)
                                                        sensordata[haus][3][pos] = 0.;
                                                if(sensordata[haus][5][pos] > 10000.)
                                                        sensordata[haus][5][pos]        =
10000.;
                                                if(sensordata[haus][4][pos] > 10000.)
                                                        sensordata[haus][4][pos]        =
10000.;

                                                for(i = pos; i < 480; i++)
                {
                                                        sensordata[haus][j][i]          =
sensordata[haus][j][pos];
                  if(i > 0)
                  {
                     if(fabs((float)sensordata[haus][j][i-1] / ((float)sensordata[haus][j][i-1]+.001)) > 1.5)
                        sensordata[haus][j][i] = sensordata[haus][j][i-1];
                  }
                }
                                        }
                                        //
                                        //  At Ctrl #2(ch1) Light Sensor was attatched....
                                        //
                                        wsensor.sdata.light = sensordata[1/*haus*/][2][pos];

                                        log_sensor_data(haus, pos);
                                        disp_sensor(pos);
                                        break;

                                case 's':
                                        status_help("온실제어기가 보낸 센서자료를 분석합니다");
                                        if(haus > 3)
                                        {
                                                for(j = 0; j < 8; j++)
                                                {
                                                        len = 6;
                                                        for(i = 32767; i > 0; i--)
                                                                if(!readcom(haus+1,
buf, &len))
        break;
                                                        if(i == 0) break;
                                                        buf[6] = 0;
                                                        sensordata[haus][j][pos] = atoi(buf);
                                                        for(i = pos; i < 480; i++)
                {
                                                                sensordata[haus][j][i]  =
sensordata[haus][j][pos];
                  if(i > 0)
                  {
                   if(fabs((float)sensordata[haus][j][i-1] / ((float)sensordata[haus][j][i-1]+.001)) > 1.5)
                      sensordata[haus][j][i] = sensordata[haus][j][i-1];
                  }
                }
                                                }
```

```
                                        log_sensor_data(haus, pos);
                                        disp_sensor(pos);
                                }
                                break;
                        }
                }
                status_help("");
                mou_wait(false);
        }
}

int dicision_wd(int degree)
{
        float d;
        d = (float)degree;
        if(d > 337.5 || d < 22.5)
                return 0;
        if(d > 22.5 && d < 67.5)
                return 1;
        if(d > 67.5 && d < 112.5)
                return 2;
        if(d > 112.5 && d < 157.5)
                return 3;
        if(d > 157.5 && d < 202.5)
                return 4;
        if(d > 202.5 && d < 247.5)
                return 5;
        if(d > 247.5 && d < 292.5)
                return 6;
        if(d > 292.5 && d < 337.5)
                return 7;
        return 0;
}

int load2orchid(void)
{
        FILE *fin;
        long offset;
        int i, j;
        ctrldata_t ctl;
        mou_wait(true);
        fin = fopen("orchid.ctl", "r+b");
        if(fin != NULL) {
                if(ctrldata_header[0].elapsed < ctrldata_header[0].duration)
                        offset = sizeof(ctrldata_header_t) + sizeof(ctrldata_t) *
(ctrldata_header[0].elapsed - 1);
                else
                        offset = sizeof(ctrldata_header_t) + sizeof(ctrldata_t) *
(ctrldata_header[0].duration - 1);
                fseek(fin, offset, SEEK_SET);
                fread(&ctl,sizeof(ctrldata_t), 1, fin);
                fclose(fin);
                for(i=0; i < 4; i++)
                        for(j=0; j < 48; j++)
                                _ctrldata_.step[i][j] = ctl.ctrlitem[i].step[j];
                memcpy(_ctrldata_.error, ctrldata_header[0].error, sizeof _ctrldata_.error);
                memcpy(_ctrldata_.mul, ctrldata_header[0].mul, sizeof _ctrldata_.mul);
                memcpy(_ctrldata_.unit, ctrldata_header[0].unit, sizeof _ctrldata_.unit);
                memcpy(_ctrldata_.can_open_wind, ctrldata_header[0].can_open_wind, sizeof
_ctrldata_.can_open_wind);
                memcpy(&_ctrldata_.can_open_rain, &ctrldata_header[0].can_open_rain, sizeof
_ctrldata_.can_open_rain);
                _ctrldata_.wind_speed = ctrldata_header[0].wind_speed;

                mou_wait(false);
                return true;
        }
        mou_wait(false);
        return false;
}


int dnload_orchid(void)
{
 int i, item;
 char buf[15];
```

```
if(!ctrldata_header[0].started || ctrldata_header[0].stoped)
                return false;

if(!load2orchid())
        return false;

mou_wait(true);
status_help("제어기에게 제어자료를 전하고 있습니다");

sprintf(buf, "%c%cLD__TEMP", _ESC_, '@');
delayedout(1, buf, 10);

item = 0;
if(!await(_RS_)) {
        mou_wait(false);
        status_help("");
        return false;
}
delay(10);
sprintf(buf, "%c", _ESC_);
delayedout(1, buf, 1);

for(i = 0; i < 48; i++)
{
        sprintf(buf,"%+04d",_ctrldata_.step[item][i]);
        delayedout(1,buf,4);
}
sprintf(buf,"%03d%03d%1d",_ctrldata_.error[item],_ctrldata_.mul[item],

        _ctrldata_.unit[item]);
delayedout(1, buf, 7);
sprintf(buf,"%c",_EOF_);
delayedout(1, buf, 1);

if(!await(_GS_))
{
        mou_wait(false);
        status_help("");
        return false;
}
status_help("");
mou_wait(false);
return true;
}


/*
int check_security(void)
{
        int ok = 0, i, len;
        char key[10] = {'M'-'0','R'-'0','E'-'0','2'-'0','A'-'0','G'-'0','&'-'0','C'-'0'};
        char send[10] = {'Z'-'0','G'-'0','&'-'0','C'-'0','2'-'0','M'-'0','R'-'0','E'-'0'};
        char buf[15], cmp[12];
        sprintf(buf,"%c%c%c%c%c%c%c%c%c.",_ESC_,
                                                        send[0]+'0', send[1]+'0', send[2]+'0
                                                        send[3]+'0', send[4]+'0', send[5]+'0
                                                        send[6]+'0', send[7]+'0');

        for(i = 0; i < 8; i++)
                key[i] += '0';
        key[8] = 0;
        for(i = 0; i < 3; i++)
        {
                port_id = 3;
                delayedout(3, buf, 10);
                for(len = 0; len < 32767; len++) delay(0);
                len = 9;
                if(!readcom(3, cmp, &len))
                {
                                cmp[9] = 0;
                                if(!strcmp(cmp+1, key))
                                {
                                        ok++;
                                        break;
                                }
                }
        }
        return ok;
```

```
}
*/
```

[온실 복합환경 제어시스템 프로그램중 List에서 생략된 파일]

1) SETENV.C : 시스템 운영환경을 정의하는 파일

2) ACCESARY.C : 계산기, 만년달력, 도스PROMPT, SYSTEM CLOCK 등의 운영을 위한 파일

3) BPLUS.C : MACRO기능을 위한 파일

4) MUX.C : MULTI-PORT사용을 지원하는 파일

5) MULTPORT.DRV : MULTI-PORT DRIVER 파일

5) MAINMENU.C : 주메뉴 운영 프로그램 화일

6) *.H : 각 소스프로그램의 HEADER화일(10종)

7) *.SCX : 윈도우-95환경에서 운영하기 위한 각종 Form파일류

8) 제어논리 생성 루틴 파일

9) 인공지능형 NN시스템 파일

```
/* ************************************** *
 *        온실제어 프로그램 version 0.0
 * ************************************** */

#include <dos.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys\stat.h>
#include <io.h>
#include <alloc.h>
#include <bios.h>

#include "hanbgi.h"

#include "ascii.h"
#include "extkey.h"
#include "hanbgi.h"
#include "hanin.h"
#include "hanlib.h"
#include "mywindow.h"
#include "mymenu.h"
#include "setenv.h"
#include "mystdio.h"
//#include "asynch.h"
#include "mux.h"
#include "accesary.h"
#include "ctrldata.h"
#include "pc2ctrl.h"
#include "setctrl.h"

extern char *bangbang[30];
extern char *state[30][3];
extern int printsw;

extern int fire_hi_mist_temp; // default 30 centi degree.

extern ctrl_data_t _ctrldata_;
extern status_flag_t flag;

void disp_temp_21(int win);


ctrldata_header_t ctrldata_header[MAX_HAUS] =
{
 { 0, //1
           "Control Logic Data File v1.0",
           false, //start
           false, //stop
           false, //ed
           false, //ind
           "",    //crop
           0,0,0, //sd
           0,0,0, //ed
           0, //d8ur
           0, //ela
           0, //exc
           { "","","","","","","","","","",""},
           { 2, 10, 5, 5},
           { 1,  1, 10, 100},
 },
 { 0, //2
           "Control Logic Data File v1.0",
           false, //start
           false, //stop
           false, //ed
           false, //ind
           "",    //crop
           0,0,0, //sd
           0,0,0, //ed
           0, //d8ur
           0, //ela
           0, //exc
           { "","","","","","","","","","",""},
           { 2, 10, 5, 5},
           { 1,  1, 10, 100},
```

```
},
{ 0, //3
        "Control Logic Data File v1.0",
        false,
        false,
        false,
        false,  // ind
        "",
        0,0,0,
        0,0,0,
        0,
        0,
        0,
        { "","","","","","","","" },
        { 2, 10, 5, 5},
        { 1,  1, 10, 100},
},
{ 0, //4
        "Control Logic Data File v1.0",
        false,
        false,
        false,
        false,  // ind
        "",
        0,0,0,
        0,0,0,
        0,
        0,
        0,
        { "","","","","","","","" },
        { 2, 10, 5, 5},
        { 1,  1, 10, 100},
},
{ 0, //5
        "Control Logic Data File v1.0",
        false,
        false,
        false,
        false,  // ind
        "",
        0,0,0,
        0,0,0,
        0,
        0,
        0,
        { "","","","","","","","" },
        { 2, 10, 5, 5},
        { 1,  1, 10, 100},
},
{ 0, //6
        "Control Logic Data File v1.0",
        false,
        false,
        false,
        false,  // ind
        "",
        0,0,0,
        0,0,0,
        0,
        0,
        0,
        { "","","","","","","","" },
        { 2, 10, 5, 5},
        { 1,  1, 10, 100},
},
{ 0, // 7
        "Control Logic Data File v1.0",
        false,
        false,
        false,
        false,  // ind
        "",
        0,0,0,
        0,0,0,
        0,
        0,
        0,
```

```
        { "","","","","","","",""},
        { 2, 10, 5, 5},
        { 1,  1, 10, 100},
 }
};

int far sensordata[MAX_HAUS][8][480];
int disp_data_pos;
int disable_disp_sensor = false;
//void siren(void);
int edit_waterctrldata(void);
void disp_temp_hy(int win);

extern button_t status[90];
extern int isautomatic;

int upload_status_flag(status_flag_t *flag);
int dnload_status_flag(status_flag_t flag);
int dnload_stationdata(void);
void agc_mre(void);
int dnload_orchid(void);

struct {
        char crop[40];
        unsigned int  duration;  //max 65535 days
        int excess;
        int  syear, smon, sday; // 1994 22 22
        int  eyear, emon, eday;
        unsigned char ctrldata_fn[MAX_PATH];
        unsigned char data_log_fn[MAX_PATH];
        unsigned char ctrl_log_fn[MAX_PATH];
        char memo[8][60];
} temp;


void disp_cntl(void);
void log_sensor_data(int haus, int pos);  // 센서 자료 로그
void log_ctrl_data(int haus);

char *WEEK[] ={"일","월","화","수","목","금","토"};
char *wind_direction[8]={" 북 ","북동"," 동 ","남동"," 남 ","남서"," 서 ", "북서"};
char *hausname[] = {"온실1","온실2","온실3","온실4","양액1","양액2", "양액3"};

config_t osjk_env = {
       37.0,//float latitude;  // 위도
       127.0,//float longitude; // 경도
       0, //current haus 현재 표시되는 하우스
       {
        {
                0, // char edited; // control data is edited?
                false, // is auto
                "OSJK_H1.CTL", //unsigned char ctrldata_fn[MAX_PATH];
                "SENS_H1",//unsigned char data log_fn[MAX_PATH];
                "CTRL_H1",//unsigned char ctrl_log_fn[MAX_PATH];
                {0,0,0,0,0,0,0,0},//unsigned char can_open_wind[8];
                0, //unsigned char can_open_rain;
                0
        },
        {
                0, // char edited; // control data is edited?
                false,  // is auto
                "OSJK_H2.CTL", //unsigned char ctrldata_fn[MAX_PATH]
                "SENS_H2",//unsigned char data_log_fn[MAX_PATH];
                "CTRL_H2",//unsigned char ctrl_log_fn[MAX_PATH];
                {0,0,0,0,0,0,0,0},//unsigned char can_open_wind[8];
                0, //unsigned char can_open_rain;
                0
        },
        {
                0, // char edited; // control data is edited?
                false,  // is auto
                "OSJK_H3.CTL", //unsigned char ctrldata_fn[MAX_PATH]
                "SENS_H3",//unsigned char data_log_fn[MAX_PATH];
                "CTRL_H3",//unsigned char ctrl_log_fn[MAX_PATH];
                {0,0,0,0,0,0,0,0},//unsigned char can_open_wind[8];
                0, //unsigned char can_open_rain;
```

```
                0
        },
        {
                0, // char edited; // control data is edited?
                false,  // is auto
                "OSJK_H4.CTL", //unsigned char ctrldata_fn[MAX_PATH];
                "SENS_H4",//unsigned char data_log_fn[MAX_PATH];
                "CTRL_H4",//unsigned char ctrl_log_fn[MAX_PATH];
                {0,0,0,0,0,0,0,0},//unsigned char can_open_wind[8];
                0, //unsigned char can_open_rain;
                0
        },
        {
                0, // char edited; // control data is edited?
                false,  // is auto
                "OSJK_W1.CTL", //unsigned char ctrldata_fn[MAX_PATH]
                "SENS_W1",//unsigned char data_log_fn[MAX_PATH];
                "CTRL_W1",//unsigned char ctrl_log_fn[MAX_PATH];
                {0,0,0,0,0,0,0,0},//unsigned char can_oper_wind[8];
                0, //unsigned char can_open_rain;
                0
        },
        {
                0, // char edited, // control data is edited?
                false,  // is auto
                "OSJK_W2.CTL", //unsigned char ctrldata_fn[MAX_PATH]
                "SENS_W2",//unsigned char data_log_fn[MAX_PATH];
                "CTRL_W2",//unsigned char ctrl_log_fn[MAX_PATH];
                {0,0,0,0,0,0,0,0},//unsigned char can_open_wind[8];
                0, //unsigned char can_open_rain;
                0
        },
        {
                0, // char edited; // control data is edited?
                false,  // is auto
                "OSJK_W3.CTL", //unsigned char ctrldata_fn[MAX_PATH]
                "SENS_W3",//unsigned char data_log_fn[MAX_PATH];
                "CTRL_W3",//unsigned char ctrl_log_fn[MAX_PATH];
                {0,0,0,0,0,0,0,0},//unsigned char can_open_wind[8];
                0, //unsigned char can_open_rain;
                0
        },
       }
};

int set_on_off_rain(void);
int _set_item(void);
int set_pos(void);
int setprint(void);

win_popupmenu_t _environment_ =
{
 { 8, 18, 36, 24,
          WHITE, BLACK, CYAN, CYAN, BLACK, BLACK, LIGHTGRAY,
          0, 0,
          0,
          true,
          true,
          NULL,
          {"온실환경설정"},
          false
 },
 4,
 1,
 false,
 0,
 BLUE, BLACK, LIGHTGRAY, CYAN,
 {
         {"강우시 천/측창 제어          ^UC^u",
          "강우시 천/측창 제어치 설정",
          1,
          0,
          'C',
          NULL,
          set_rain_status,
         },
         {"풍향에 따른 천/측창 제어  ^UW^u",
```

```
            "풍향에 따른 천/측창 제어치",
            1,
            0,
            'W',
            NULL,
            set_wind_status
           },
           {"온실위치 설정(위도/경도)  ^UP^u",
            "지방의 위도/경도 설정",
            1,
            0,
            'P',
            NULL,
            set_pos
           },
           {"온실센서상황 인쇄              ",
            "지방의 위도/경도 설정",
            1,
            0,
            0,
            NULL,
            setprint
           },
 }
};

win_popupmenu_t _rain_ =
{
 { 12, 12, 34, 22,
           WHITE, BLACK, CYAN, CYAN, BLACK, BLACK, LIGHTGRAY
           0, 0,
           0,
           true,
           true,
           NULL,
           {"비올때 창 제어설정"},
           false
 },
 8,
 1,     //escout
 false, //close & exec
 0,
 BLUE, BLACK, LIGHTGRAY, CYAN,
 {
           {"천창 1                  ^U1^u",
            "1번 천창 열림/닫힘 선택",
            1,
            0,
            '1',
            NULL,
            set_on_off_rain
           },
           {"천창 2                  ^U2^u",
            "2번 천창 열림/닫힘 선택",
            1,
            0,
            '2',
            NULL,
            set_on_off_rain
           },
           {"천창 3                  ^U3^u",
            "3번 천창 열림/닫힘 선택",
            1,
            0,
            '3',
            NULL,
            set_on_off_rain
           },
           {"천창 4                  ^U4^u",
            "1번 천창 열림/닫힘 선택",
            1,
            0,
            '4',
            NULL,
            set_on_off_rain
           },
           {"측창 1                  ^U5^u",
```

```
            "1번 천창 열림/닫힘 선택",
            1,
            0,
            '5',
            NULL,
            set_on_off_rain
          },
          {"측창 2                ^U6^u",
            "2번 측창 열림/닫힘 선택",
            1,
            0,
            '6',
            NULL,
            set_on_off_rain
          },
          {"측창 3                ^U7^u",
            "3번 측창 열림/닫힘 선택",
            1,
            0,
            '7',
            NULL,
            set_on_off_rain
          },
          {"측창 4                ^U8^u",
            "4번 측창 열림/닫힘 선택",
            1,
            0,
            '8',
            NULL,
            set_on_off_rain
          },
  }
};

int disp_pos[6][4] =
          {
                    {19, 2, 107, 14},
                    {19, 15, 107, 27},
                    {19, 28, 107, 40},

                    {216,192,500, 0},
                    {216,400,500, 0},
                    {216,608,500, 0}
          };


int setprint(void)
{
          printsw = !printsw;
          if(printsw)
                    putmsg("프린터로 센서자료를 출력합니다", true);
          else
                    putmsg("프린터로 센서자료를 출력하지 않습니다", true);
          sleep(1);
          closemsg();
          return false;
}


int set_environment(void)
{
          putmsg("환경을 임의로 변경하면 치명적인 손해를 볼 수 있습니다", true);
          delay(500);
          closemsg();
          win_popupmenu(&_environment_);

          return false;
}


int set_on_off_rain(void)
{
          if(osjk_env.haus[osjk_env.c_haus].can_open_rain
                    & (0x01 << _rain_.lastsel))
          {
           strncpy(&_rain_.menu[_rain_.lastsel].menu[12]," ° 닫힘",7);
           osjk_env.haus[osjk_env.c_haus].can_open_rain &= (~(0x01 << _rain_.lastsel))
```

```
        } else {
          strncpy(&_rain_.menu[_rain_.lastsel].menu[12],"○ 열림",7);
          osjk_env.haus[osjk_env.c_haus].can_open_rain |= (0x01 << _rain_.lastsel);
        }
        osjk_env.haus[osjk_env.c_haus].edited = true;
        return false;
}

int set_rain_status(void)
{
 int i;
 for(i = 0; i < 8; i++)
        if(osjk_env.haus[osjk_env.c_haus].can_open_rain & (0x01 << i))
          strncpy(&_rain_.menu[i].menu[12],"○ 열림",7);
        else
          strncpy(&_rain_.menu[i].menu[12],"  ˇ 닫힘",7);

 win_popupmenu(&_rain_);
 return false;
}

int set_wind_status(void)
{
 int i, j, done = false, key;
 char *direc[8] = {
                        " 북 ",
                        "북동",
                        " 동 ",
                        "남동",
                        " 남 ",
                        "남서",
                        " 서 ",
                        "북서"
 };
 char *top[9] = {
                        "풍향",
                        "천창1","천창2","천창3","천창4",
                        "측창1","측창2","측창3","측창4"
 },
 char tbl[8][8][7], buf[12];

 mywindow_t wind;
 button_t btn[17],speed;

 setmywindow(&wind, 13, 10, 68, 23, WHITE, BLACK, CYAN, LIGHTGRAY,
                                                BLUE, BLACK, LIGHTGRAY, true, true,
                                                "바람불 때 창 제어설정",false);

 for(i=0; i < 9; i++)
        set_button(&btn[i], 14 + 6*i, 12, 20 + 6*i, 13,
                                                        WHITE, BLACK, GREEN,
WHITE, BLACK,
                                                        false, false, top[i]);
 for(i=0; i < 8; i++)
        set_button(&btn[9+i], 14, 13+i, 20, 14+i,
                                                        WHITE, BLACK, CYAN,
YELLOW, BLACK,
                                                        false, false, direc[i]);
 for(j = 0; j < 8; j++)
        for(i = 0; i < 8; i++)
          if(osjk_env.haus[osjk_env.c_haus].can_open_wind[j]      & (0x01 << i))
                strcpy(tbl[j][i],"  ○  ");
          else
                strcpy(tbl[j][i],"     ˚  ");

 pushcursor();
 _showcursor = false;
 if(open_window(&wind)) {
          // disp pannel
          set_button(&speed, 14, 22, 50, 23,
                                                        WHITE, BLACK, GREEN,
WHITE, BLACK,
                                                        false, false, "제한풍속설정
TAB");
          open_button(&speed);

          for(i=0; i < 17; i++)
```

```
        open_button(&btn[i]);

       for(i = 0; i < 8; i++)
        for(j = 0; j < 8; j++)
                hputsxy(7+6*j, i + 2, tbl[i][j]);
       i = j = 0;
       disp_button(&btn[j+1], true);
       disp_button(&btn[9+i], true);

       allow_darg(false);
       do {
        //disp cursor
        hprintfxy(40,15,"%3d m/sec", osjk_env.haus[osjk_env.c_haus].wind_speed);
        sprintf(buf,"%3d", osjk_env.haus[osjk_env.c_haus].wind_speed);
        if(key == LEFTARROW || key==RIGHTARROW || key == MOUKEY ||
                        key == PGUPKEY || key == PGDNKEY || key == HOMEKEY || key
ENDKEY)
                disp_button(&btn[j+1], true);
        if(key == UPARROW || key == DOWNARROW || key == MOUKEY ||
                        key == PGUPKEY || key == PGDNKEY || key == HOMEKEY || key
ENDKEY)
                disp_button(&btn[9+i], true);
        hsetreverse(true);
        hputsxy(7+6*j, i + 2, tbl[i][j]);
        hsetreverse(false);
        //getkey
        key = hgetch();
        // erase cursor
        if(key == LEFTARROW || key==RIGHTARROW || key == MOUKEY ||
                        key == PGUPKEY || key == PGDNKEY || key == HOMEKEY || key
ENDKEY)
                disp_button(&btn[j+1], false);
        if(key == UPARROW || key == DOWNARROW || key == MOUKEY ||
                        key == PGUPKEY || key == PGDNKEY || key == HOMEKEY || key
ENDKEY)
                disp_button(&btn[9+i], false);
        hputsxy(7+6*j, i + 2, tbl[i][j]);
        switch(key) {
                case TAB:
                 win_getdata("한계 풍속을 입력하세요(m/sec)", buf, "999", ENGONLY);
                 buf[3] = 0;
                 osjk_env.haus[osjk_env.c_haus].wind_speed = atoi(buf);
                 break;
                case PGUPKEY:
                 i = 0;
                 break;
                case PGDNKEY:
                 i = 7;
                 break;
                case HOMEKEY:
                 j = 0;
                 break;
                case ENDKEY:
                 j = 7;
                 break;
                case UPARROW:
                 i = (i+7) % 8;
                 break;
                case DOWNARROW:
                 i = (i+1) % 8;
                 break;
                case LEFTARROW:
                 j = (j+7) % 8;
                 break;
                case RIGHTARROW:
                 j = (j+1) % 8;
                 break;
                case ' ':
                 ungetxch(DOWNARROW);
                 break;
                case CR:
                 osjk_env.haus[osjk_env.c_haus].edited = true;
                 if(osjk_env.haus[osjk_env.c_haus].can_open_wind[i] & (0x01 << j)) {
                        strcpy(tbl[i][j],"  ●  ");
                        osjk_env.haus[osjk_env.c_haus].can_open_wind[i] &= (~(0x01 << j));
                 } else {
                        strcpy(tbl[i][j],"  ○  ");
```

```
                        osjk_env.haus[osjk_env.c_haus].can_open_wind[i] |= (0x01 << j);
                    }
                    break;
                  case ESC:
                    done = true;
                    break;
                  case MOUKEY:
                    if(mou_ax > 19 && mou_ax < 67 && mou_ay > 12 && mou_ay < 21) {
                        j = (mou_x-7) / 6;
                        i = mou_y-2;
                        ungetxch(CR);
                    } else {
                        done = true;
                        ungetxch(MOUKEY);
                    }
                    break;
                } // end switch
            } while(!done);
            for(i=0; i < 17; i++)
              close_button(&btn[i]);
            close_button(&speed);
            close_window(&wind);
    }
    popcursor();
    return false;
}

int set_pos(void)
{
        char buf[10];
        sprintf(buf,"%3d ° %02.0f\'",(int)osjk_env.longitude,

(osjk_env.longitude-(int)osjk_env.longitude)*100);
        if(win_getdata("현 위치의 경도는(### ° ##\')?", buf, "999 ° 99\'", ENGONLY))
        {
                buf[3] = buf[7] = 0;
                osjk_env.longitude = atof(buf)+ atof(&buf[5])/100.;
                sprintf(buf,"%2d ° %02.0f\'",(int)osjk_env.latitude,

(osjk_env.latitude-(int)osjk_env.latitude)*100);
                if(win_getdata("현 위치의 위도는(## ° ##\')?", buf, "99 ° 99\'", ENGONLY)) {
                        buf[2] = buf[6] = 0;
                        osjk_env.latitude = atof(buf)+atof(&buf[4])/100.;
                }
        }
        return false;
}

int read_config(void)
{
    int file;
    if((file = open("OSJK_12.CFG",O_RDONLY | O_BINARY)) < 0)
            return false;
    read(file,&osjk_env, sizeof osjk_env);
    read(file, sensordata, sizeof sensordata);
    read(file,&printsw, sizeof printsw);
    read(file, &fire_hi_mist_temp, sizeof fire_hi_mist_temp);

    close(file);
    return true;
}

int write_config(void)
{
    int file;
    if((file = open("OSJK_12.CFG",O_WRONLY | O_BINARY | O_CREAT, S_IWRITE)) < 0)
            return false;
    write(file,&osjk_env, sizeof osjk_env);
    write(file, sensordata, sizeof sensordata);
    write(file,&printsw, sizeof printsw);
    write(file, &fire_hi_mist_temp, sizeof fire_hi_mist_temp);
    close(file);
    return true;
}

void disp_crop_season(void)
{
```

```
 int j;

 ctrlprintfxy(2,1,"^C4편집대상 온실번호:^C1 %s %02d.",
                                                    osjk_env.c_haus < 4 ? "기상":"양액",
                                                    osjk_env.c_haus      <      4      ?
osjk_env.c_haus+1:osjk_env.c_haus-3);
 ctrlprintfxy(2,2,"^O=========================================^o");
 ctrlprintfxy(2,3,"^CA재 배 작 물 :^CF %-.39s",

ctrldata_header[osjk_env.c_haus].crop);
 ctrlprintfxy(2,4,"^CA예 상 작 기 :^CF %05d일",

ctrldata_header[osjk_env.c_haus].duration);
 ctrlprintfxy(2,5,"^C1예상 제어 시작일 :^CF %4d년 %02d월 %02d일",

ctrldata_header[osjk_env.c_haus].syear,

ctrldata_header[osjk_env.c_haus].smon,

ctrldata_header[osjk_env.c_haus].sday);
 ctrlprintfxy(2,6,"^C1예상 제어 마침일 :^CF %4d년 %02d월 %02d일",

ctrldata_header[osjk_env.c_haus].eyear,

ctrldata_header[osjk_env.c_haus].emon,

ctrldata_header[osjk_env.c_haus].eday);
 ctrlprintfxy(2,7,"^C0경과 작기 : %5d일", ctrldata_header[osjk_env.c_haus].elapsed);
 ctrlprintfxy(2,8,"^C0초과 작기 : %5d일", ctrldata_header[osjk_env.c_haus].excess);
 ctrlprintfxy(2,9,"^C4제어 논리 기록 화일명 :^CF %-.12s",

osjk_env haus[osjk_env.c_haus].ctrldata_fn);
 ctrlprintfxy(2,10,"^C4실행 명령 기록 화일명 :^CF %-.12s",

osjk_env haus[osjk_env.c_haus].ctrl_log_fn);
 ctrlprintfxy(2,11,"^C4센서 자료 기록 화일명 :^CF %-.12s",

osjk_env.haus[osjk_env.c_haus].data_log_fn);
 ctrlprintfxy(2,12,"^C1^O=======================^o 메      모 ^O=======================^o^CF");
 for(j=0; j < 8; j++)
          hprintfxy(2,13+j,"%-.58s", ctrldata_header[osjk_env.c_haus].memo[j]);
}

#pragma warn -par
bool crop_hookhgetdata(int *keycodep, byte *string,

                                          int *curpp, int *lastpp, size_t n)
{
 switch(*keycodep)
 {
          case UPARROW:
          case DOWNARROW:
          case F2:
          case MOUKEY:
                    *keycodep = CR;
                    break;
 }
 return true;
}
#pragma warn +par

void calc_edays(int year, int mon, int day, long duration,
                                                                int  *eyear,  int  *emon,  int
*eday)
{
          duration += day;
          while(duration - Year[leap(year)][mon] > 0)
          {
                    duration -= Year[leap(year)][mon];
                    mon++;
                    if(mon > 12) {
                              year++;
                              mon = 1;
                    }
          }
          *eyear = year;
          *emon = mon;
```

```
        *eday = (int)duration;
}

int edit_crop_season(void)
{
 mywindow_t crop_season;
 char buf[20], *ch;
 FILE *fp;
 int i, done = false;
 long fsize;
 int pos[14][2]=
 { {16,  3},
         {21,  5},
         {21,  6},
         {26,  9},{26, 10},{26, 11},
         { 2, 13},{ 2, 14},{ 2, 15},{ 2, 16},{ 2, 17},{ 2, 18},{ 2, 19},{ 2, 20}
 };
 button_t btn[2];
 char *text[2] =
                         {"취소 ESC",
                          "저장  F2"
                         };
 struct date d, fr, to;
 getdate(&d);
 setmywindow(&crop_season, 34, 4, 95, 26, WHITE, BLACK, GREEN, LIGHTGRAY,
                                                 BLUE, BLACK, LIGHTGRAY, false, true,
                                                 "온실별 작기(Crop Season) 편집",false);

 savescreen();
 if(open_window(&crop_season)) {

         for(i = 0; i < 2; i++)
         {
                 set_button(&btn[i], 76+9*i, 6, 85+9*i, 7,
                                                                          WHITE,  BLACK,
GREEN, WHITE, BLACK,
                                                                        false, false, text[i]);
                 open_button(&btn[i]);
         }
         /* copy to temp */
         strcpy(temp.crop,ctrldata_header[osjk_env.c_haus].crop);
//       _fmemcpy(temp.memo,ctrldata_header[osjk_env.c_haus].memo,480);
         if(!ctrldata_header[osjk_env.c_haus].edited)
         {
                 ctrldata_header[osjk_env.c_haus].duration = 0;
                 ctrldata_header[osjk_env.c_haus].excess = 0;
         }
         temp.duration = ctrldata_header[osjk_env.c_haus].duration;
         temp.excess = ctrldata_header[osjk_env.c_haus].excess;
         temp.syear = ctrldata_header[osjk_env.c_haus].syear;
         temp.smon = ctrldata_header[osjk_env.c_haus].smon;
         temp.sday = ctrldata_header[osjk_env.c_haus].sday;

         temp.eyear = ctrldata_header[osjk_env.c_haus].eyear;
         temp.emon = ctrldata_header[osjk_env.c_haus].emon;
         temp.eday = ctrldata_header[osjk_env.c_haus].eday;

         strcpy(temp.ctrldata_fn, osjk_env.haus[osjk_env.c_haus].ctrldata_fn);
         strcpy(temp.ctrl_log_fn, osjk_env.haus[osjk_env.c_haus].ctrl_log_fn);
         strcpy(temp.data_log_fn, osjk_env.haus[osjk_env.c_haus].data_log_fn);
         _hookhgetdata = crop_hookhgetdata;
         pushcursor();
         pushhanmode();
         disp_crop_season();
         _showcursor = true;

         for(i = 0; i < 14 && !done;) {
                 hgotoxy(pos[i][0],pos[i][1]);
                 switch(i) {
                         case 0://재배작물
                                 _hangulmode = true;

hgetdata(temp.crop,"xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
                                 break;
                         case 1://예상제어시작일
                                 sprintf(buf, "%4d년 %02d월 %02d일", temp.syear, temp.smon
temp.sday);
```

```
                                        if(hgetdata(buf, "9999년 99월 99일") == CR && _lastkey == CR) {
                                                buf[4] = buf[9] = buf[14];
                                                temp.syear = atoi(buf);
                                                if(temp.syear < 1994)
                                                        temp.syear = d.da_year;
                                                temp.smon = atoi(&buf[7]);
                                                if(temp.smon > 12 || temp.smon < 1)
                                                        temp.smon = d.da_mon;
                                                temp.sday = atoi(&buf[12]);
                                                 if(temp.sday   >   Year[leap(temp.syear)][temp.smon]   ||
temp.sday < 1)
                                                        temp.sday = d.da_day;

//                                                  ctrlprintfxy(2,7,"^C0경과        작기        :        %5d일",
ctrldata_header[osjk_env.c_haus].elapsed);
                                                if(temp.excess < 0 || temp.excess > 1000)
                                                        temp.excess = 0;
                                                ctrlprintfxy(2,8,"^C0초과 작기 : %5d일", temp.excess);

//                                                      calc_edays(temp.syear,temp.smon,          temp.sday,
temp.duration,
//
&temp.eyear, &temp.emon, &temp.eday);
                                        }
                                        //
                                        // 제어 마침일 기록...
                                        //
                                        ctrlprintfxy(2,5,"^C1예상 제어 시작일 :^CF %4d년 %02d월 %02d일",

                                temp.syear, temp.smon, temp.sday);
                                        break;
                                case 2://예상제어마침일
                                         sprintf(buf,  "%4d년  %02d월  %02d일",  temp.eyear,  temp.emon,
temp.eday);
                                        if(hgetdata(buf, "9999년 99월 99일") == CR && _lastkey == CR) {
                                                buf[4] = buf[9] = buf[14];
                                                temp.eyear = atoi(buf);
                                                if(temp.eyear < temp.syear)
                                                        temp.eyear = temp.syear;
                                                temp.emon = atoi(&buf[7]);
                                                if(temp.emon > 12 || temp.emon < 1)
                                                        temp.emon = temp.smon;
                                                temp.eday = atoi(&buf[12]);
                                                 if(temp.eday   >   Year[leap(temp.eyear)][temp.smon]   ||
temp.eday < 1)
                                                        temp.eday = temp.sday;

                                                fr.da_year = temp.syear;
                                                fr.da_mon = temp.smon;
                                                fr.da_day = temp.sday;

                                                to.da_year = temp.eyear;
                                                to.da_mon = temp.emon;
                                                to.da_day = temp.eday;
                                                temp.duration =(int) diff_days(fr, to);
                                                temp.excess = 0;
                                                ctrlprintfxy(2,8,"^C0초과 작기 : %5d일", temp.excess);

/*
                                                      calc_edays(temp.eyear,temp.emon,          temp.eday
temp.duration,

&temp.eyear, &temp.emon, &temp.eday);
*/
                                        }
                                        //
                                        // 제어 마침일 기록...
                                        //
                                        ctrlprintfxy(2,6,"^C1예상 제어 마침일 :^CF %4d년 %02d월 %02d일"

                                temp.eyear, temp.emon, temp.eday);

                                        ctrlprintfxy(2,4,"^CA예 상 작 기 :^CF %05d일", temp.duration);
                                        break;
                                case 3://제어논리기록화일명
                                        _hangulmode = false;
                                        hgetdata(temp.ctrldata_fn,"xxxxxxxxxxxx");
```

```
                                        break;
                                case 4://실행명령기록화일명
                                        _hangulmode = false;
                                        hgetdata(temp.ctrl_log_fn,"xxxxxxxx");
                                        ch=strchr(temp.ctrl_log_fn,'.');
                                        if(ch) *ch = 0;
                                        break;
                                case 5://센서자료기록화일명
                                        _hangulmode = false;
                                        hgetdata(temp.data_log_fn,"xxxxxxxx");
                                        ch=strchr(temp.data_log_fn,'.');
                                        if(ch) *ch = 0;
                                        break;
                                case 6: case 7:case 8: case 9: case 10: case 11: case 12: case 13:
                                //메모
                                        _hangulmode = true;

hgetdata(temp memo[i-6],"xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
                                        break;
                        }
                        switch(_lastkey)
                        {
                                case ESC:
                                        done = true;
                                        break;
                                case F2:
                                        //save;
                                        strcpy(ctrldata_header[osjk_env.c_haus].crop,temp.crop);
                                        _fmemcpy(ctrldata_header[osjk_env.c_haus].memo,temp.memo,480);
                                        ctrldata_header[osjk_env.c_haus].duration = temp.duration;
                                        ctrldata_header[osjk_env.c_haus].syear = temp.syear;
                                        ctrldata_header[osjk_env.c_haus].smon = temp.smon;
                                        ctrldata_header[osjk_env.c_haus].sday = temp.sday;

                                        ctrldata_header[osjk_env.c_haus].eyear = temp.eyear;
                                        ctrldata_header[osjk_env.c_haus].emon = temp.emon;
                                        ctrldata_header[osjk_env.c_haus].eday = temp.eday;

                                        ctrldata_header[osjk_env.c_haus] excess = temp.excess;

                                        strcpy(osjk_env.haus[osjk_env.c_haus].ctrldata_fn,temp.ctrldata_fn);
                                        strcpy(osjk_env.haus[osjk_env.c_haus].ctrl_log_fn,temp ctrl_log_fn);
                                        strcpy(osjk_env haus[osjk_env.c_haus].data_log_fn,temp.data_log_fn),

                                        ////////////////////////////////////////////////
                                        ctrldata_header[osjk_env.c_haus].mul[0] = 1;
                                        ctrldata_header[osjk_env.c_haus].mul[1] = 1;
                                        ctrldata_header[osjk_env.c_haus].mul[2] = 10,


                                        osjk_env.haus[osjk_env.c_haus].edited = true;
                                        ctrldata_header[osjk_env.c_haus].edited = true;
                                        ctrldata_header[osjk_env.c_haus].init = true;
                                        //
                                        // update each file
                                        //
                                        if(!access(osjk_env.haus[osjk_env.c_haus] ctrldata_fn, 0))
                                                fp = fopen(osjk_env.haus[osjk_env.c_haus].ctrldata_fn,
"r+b");
                                        else
                                                fp = fopen(osjk_env.haus[osjk_env.c_haus].ctrldata_fn,
"w+b");
                                        if(fp != NULL) {
                                                fseek(fp, 0, SEEK_SET);
                                                        fwrite(&ctrldata_header[osjk_env.c_haus],
sizeof(ctrldata_header_t), 1, fp);

                                                fsize = sizeof(ctrldata_header_t) + sizeof(ctrldata_t) *
ctrldata_header[osjk_env.c_haus].duration;
                                                chsize(fileno(fp), fsize);
                                                fclose(fp);
                                        } else
                                                puterror("화일에 쓸 수가 없습니다");
                                        done = true;
                                        break;
                                case UPARROW:
                                        i = (i + 13) % 14;
```

```
                        break;
                case MOUKEY:
                        switch(process_button(btn, 2)) {
                                case 0:
                                 ungetxch(ESC);
                                 break;
                                case 1:
                                 ungetxch(F2);
                                 break;
                        }
                        break;
                case DOWNARROW:
                default:
                        i = (i + 1) % 14;
                        break;
                }
        }
        for(i = 0 ; i < 2; i++) {
                close_button(&btn[i]);
        }
        close_window(&crop_season);

        popcursor();
        pophanmode();
}
restorescreen();
_hookhgetdata = NULL;


return false;
}

int show_ctrldata_info(void)
{
 char week;
 mywindow_t info_w;
 struct date d;
 float freem;

 if(!ctrldata_header[osjk_env.c_haus].started)
        puterror("제어가 시작되지 않았습니다");
 else if(ctrldata_header[osjk_env.c_haus].stoped)
        puterror("제어가 끝났습니다");


 setmywindow(&info_w, 34, 4, 95, 24, WHITE, BLACK, DARKGRAY, LIGHTGRAY,
                                                BLUE, BLACK, LIGHTGRAY, false, true,
                                                "온실제어정보",false);
 savescreen();
 freem = farcoreleft();
 if(open_window(&info_w)) {
        getdate(&d);
        week = (getweek(d.da_year, d.da_mon) + d.da_day - 1) % 7;
        ctrlprintfxy(2,2,"^C4온 실 번 호:^C1 %s %02d.",

osjk_env.c_haus < 4 ? "기상":"양액",

osjk_env.c_haus < 4? osjk_env.c_haus + 1 : osjk_env.c_haus-3);
        ctrlprintfxy(2,3,"^C0재 배 작 물:^CF %-s.",ctrldata_header[osjk_env.c_haus].crop);
        ctrlprintfxy(2,4,"^C0현 재 날 짜:^CF %4d년 %02d월 %02d일 (%s요일)", d.da_year, d.da_mon
d.da_day, WEEK[week]);
        ctrlprintfxy(2,5,"^O=============================================================^o");
        ctrlprintfxy(2,7,"^C1제어시작일:^CF %4d년 %02d월 %02d일  ^C1제어마침일:^CF %4d년 %02d월
%02d일",

ctrldata_header[osjk_env.c_haus].syear,

ctrldata_header[osjk_env.c_haus].smon,

ctrldata_header[osjk_env.c_haus].sday,

ctrldata_header[osjk_env.c_haus] eyear,

ctrldata_header[osjk_env.c_haus].emon,

ctrldata_header[osjk_env.c_haus].eday);
        ctrlprintfxy(2,9,"^C0예상 재배 작기:^CF %5d 일",
```

```
ctrldata_header[osjk_env.c_haus].duration);
        ctrlprintfxy(2,10,"^C0경과 재배 작기:^CF %5d 일",

ctrldata_header[osjk_env.c_haus].elapsed);
        ctrlprintfxy(2,11,"^C0초과 재배 작기:^CF %5d 일",

ctrldata_header[osjk_env.c_haus].excess);

        ctrlprintfxy(2,13,"^C4온실의 지리적 위치:^C1 [동경] %3.0f° %02.0f\'        [북위] %2.0f
%02.0f\'",
                                                        osjk_env.longitude,
(osjk_env.longitude-(int)osjk_env.longitude)*100,
                                                        osjk_env.latitude,
(osjk_env.latitude-(int)osjk_env.latitude)*100);
        ctrlprintfxy(2,15,"^C4제어기의 통신 포트: IRQ10, 8 포트, 500H, 500bytes buffer");
        ctrlprintfxy(2,16,"^C4사용 가능한 메모리:^C1 %.0f(%.1f k) bytes",freem,freem/1024.);

        ctrlprintfxy(1,18,"^C1^R 온실지기  제 1.2 판.  Copyright 1994,95 농업과 컴퓨터 Co. ^r");
        pushcursor();
        _showcursor = false;
        hgetch();
        popcursor();
        close_window(&info_w);
 }
 restorescreen();
 return false;
}

int edit_weater_water_ctrldata(void)
{
        switch(osjk_env.c_haus)
        {
                case 0: case 1: case 2: case 3:
                        edit_ctrldata();
                        break;
                case 4: case 5: case 6:
                        edit_waterctrldata();
                        break;
        }
        return false;
}


int edit_ctrldata(void)
{
 ctrldata_t tmpdata, tmpdata2;
 FILE *fin;
 mywindow_t info_w;
 button_t btn[20];
 int hkey[20] =
        { LEFTARROW, RIGHTARROW, UPARROW, DOWNARROW,
                '-', '+', PGUPKEY, PGDNKEY, HOMEKEY, ENDKEY,
                F5, F6, F10, ESC, F2, CR, ALT_1, ALT_2, ALT_3, ALT_4};
 int ii, items = 0, key, done = false, redraw = true, step = 0, day = 0;
 int last_item = -1;
 int pos[20][4] =
                {
                        {27,  24, 31, 25},
                        {31,  24, 35, 25},
                        {35, 24, 39, 25},
                        {39, 24, 43, 25},

                        {43, 24, 53, 25},
                        {53, 24, 63, 25},
                        {63, 24, 73, 25},
                        {73, 24, 83, 25},
                        {83, 24, 93, 25},
                        {93, 24, 103, 25},

                        {27, 25, 40, 27},
                        {40, 25, 53, 27},

                        {53, 25, 63, 27},

                        {63, 25, 73, 27},
                        {73, 25, 83, 27},
```

```
                         {83, 25, 103, 27},
                         // select item
                         {92, 5,  103, 7 },
                         {92, 7,  103, 9 },
                         {92, 9,  103, 11 },
                         {92,11,  103, 13 }
               };
int scale[4] = {4, 2, 2, 4}, scrollbar, cyr, cmon, cday;
float  max, min, mean;
struct date d;
int sh, sm, eh, em;
char *text[20] =
        {"Ц","Ш","▲","▼",
         "어제 -","내일 +", "앞쪽 PgUp","뒤쪽 PgDn","처음 Home","끝 End",
         "일괄지정 F5","개별지정 F6","오차 F10","취소 ESC","완료 F2", "저장 ⊟",
         "온도 @1", "습도 @2", "CO2  @3", "광도 @4"};
char buf[80];
char *report[4][4] =
{
                {"^c1^CC▼^CF    최고온도^c7","^c1^C0▼^CF    최저온도^c7","^c1^C2▼^CF    평균온도
^c7","^c1^CB▼^CF 설정온도^c7"},
                {"^c1^CC▼^CF    최고습도^c7","^c1^C0▼^CF    최저습도^c7","^c1^C2▼^CF    평균습도
^c7","^c1^CB▼^CF 설정습도^c7"},
                {"^c1^CC▼^CF   최고   CO2^c7","^c1^C0▼^CF    최저   CO2^c7","^c1^C2▼^CF   평균
CO2^c7","^c1^CB▼^CF 설정 CO2^c7"},
                {"^c1^CC▼^CF    최고광도^c7","^c1^C0▼^CF    최저광도^c7","^c1^C2▼^CF    평균광도
^c7","^c1^CB▼^CF 설정광도^c7"}
};

        if(!ctrldata_header[osjk_env.c_haus].init) {
                puterror("작기가 설정되어 있지 않습니다. 먼저 작기를 설정하세요");
                return false;
        }

        for(ii = 0 ; ii < 20; ii++) {
                set_button(&btn[ii], pos[ii][0],pos[ii][1],pos[ii][2],pos[ii][3],
                WHITE, BLACK, ii < 16 ? LIGHTGRAY:GREEN,ii < 16 ? LIGHTGREEN:WHITE,
BLACK,
                false, false, text[ii]);
        }

        sprintf(buf,"온실제어정보 편집기 : %02d 번 온실", osjk_env.c_haus + 1);
        setmywindow(&info_w, 25, 3, 104, 27, WHITE, BLACK, GREEN, BLUE,
                                        WHITE, BLACK, LIGHTGRAY, false, true,
                                        buf, false);
        memset(&tmpdata, 0, sizeof(ctrldata_t));
        memset(&tmpdata2, 0, sizeof(ctrldata_t));
        savescreen();
        if(open_window(&info_w)) {
                pushcolor();
                hsetbkcolor(LIGHTGRAY);
                hsetcolor(BLACK);
                allow_darg(true);

                hsolidbarpxy(-208,-63, -511, -280, LIGHTGRAY);

                rectangle_3d(208, 63, 511, 280, WHITE, BLACK);
                // file open
                mou_wait(true);
                copyfile(osjk_env.haus[osjk_env.c_haus].ctrldata_fn, "_ctrl_.$$$");
                mou_wait(false);

                fin = fopen(osjk_env.haus[osjk_env.c_haus].ctrldata_fn, "r+b");
                if(fin != NULL)
                {
                        for(ii = 0; ii < 20; ii++)
                                open_button(&btn[ii]);
                        ctrlputspxy(8,282,"^C3^cF^RЦ^r■■■■■■■■■■■■■■■■■■■■■
■■■■■■■■■^RШ^r^C0^c7");
                        if(ctrldata_header[osjk_env.c_haus].elapsed)
                                day = ctrldata_header[osjk_env.c_haus].elapsed-1;
                        else
                                day = 0;
                        while(!done)
                        {
                                if(redraw) {
                                        fseek(fin,      (long)sizeof(ctrldata_header_t)      +
```

```
(long)sizeof(ctrldata_t) * (long)day, SEEK_SET);
                                        fread(&tmpdata2, sizeof(ctrldata_t), 1, fin);
                                        if(tmpdata2.init[items])
                                                tmpdata = tmpdata2;

                                        if(last_item != items|| key == F10)
                                        {
                                                hsolidbarpxy(-209,-80,      -509,      -263,
LIGHTGRAY);
                                        disp_button(&btn[16+items], true);

                                        hhline(80,260,420, BLACK);
                                        hvline(80,20,240, BLACK);
                                        hprintfpxy(30, 262,"시각 →");
                                        for(ii = 0; ii < 48; ii++) {
                                                hvline(ii*8+115, ii % 2 ? 255:250,ii % 2
? 5:10, BLACK);
                                                if(!(ii % 4))
                                                        hprintfpxy(ii*8+108,
262,"%02d", ii / 2);
                                        }
                                        switch (items)
                                        {
                                                case 0: // 온도
                                                        for(ii = 0; ii <= 50; ii+=10) {
                                                                hhline(80, ii * 4 +
24, 10, BLACK);
                                                                hprintfpxy(30, ii * 4
+ 16, "%2d ° C", 50-ii);
                                                        }
                                                        ctrlprintfpxy(530, 282,"^c2^CF
[± %3d ° C]^C0^c7",
                                ctrldata_header[osjk_env.c_haus].error[items]
                               * ctrldata_header[osjk_env.c_haus].mul[items]);
                                                        break;
                                                case 1: // 습도
                                                        for(ii = 0; ii <= 100; ii+=10)
{
                                                                hhline(80, ii * 2 +
24, 10, BLACK);
                                                                hprintfpxy(30, ii * 2
+ 16, "%3d %%", 100-ii);
                                                        }
                                                        ctrlprintfpxy(530, 282,"^c2^CF
[± %3d  %%]^C0^c7",
                                ctrldata_header[osjk_env.c_haus].error[items]
                                * ctrldata_header[osjk_env.c_haus].mul[items]);
                                                        break;
                                                case 2: // 이산화탄소
                                                        for(ii = 0; ii <= 100; ii+=10)
{
                                                                hhline(80, ii * 2 +
24, 10, BLACK);
                                                                hprintfpxy(22, ii * 2
+ 16, "%4dppm", 1000-ii*10);
                                                        }
                                                        ctrlprintfpxy(530, 282,"^c2^CF
[± %4dppm]^C0^c7",
                                ctrldata_header[osjk_env.c_haus].error[items]
                                * ctrldata_header[osjk_env.c_haus].mul[items]);
                                                        break;
                                                case 3: // 과앙
                                                        for(ii = 0; ii <= 50; ii+=10) {
                                                                hhline(80, ii * 4 +
24, 10, BLACK);
                                                                hprintfpxy(10, ii * 4
+ 16, "%2dkW/m₂ ", (5000-ii*100)/1000);
                                                        }
                                                        ctrlprintfpxy(530, 282,"^c2^CF
±%4dW/m₂ ^C0^c7",
```

```
                    ctrldata_header[osjk_env.c_haus].error[items]
                * ctrldata_header[osjk_env.c_haus].mul[items]);
                                                    break;
                                    }
                                    for(ii = 0; ii < 4; ii++)
                                            ctrlprintfxy(67,    ii    *    2    +    10
report[items][ii]);
                                    last_item = items;
                            }
                            redraw = false;
                    }

                    // draw cursor
                    for(ii = 0; ii < 48; ii++)
                            hprintfpxy(8*(14+ii), 216-tmpdata.ctrlitem[items].step[ii] *
scale[items],"-");

                    for(ii=0, max = 0; ii < 48; ii++) {
                            if(tmpdata.ctrlitem[items].step[ii] > max)
                                    max = tmpdata.ctrlitem[items].step[ii];
                    }
                    for(ii=0, min = tmpdata.ctrlitem[items].step[0]; ii < 48; ii++) {
                            if(tmpdata.ctrlitem[items].step[ii] < min)
                                    min = tmpdata.ctrlitem[items].step[ii];
                    }
                    for(ii=0, mean = 0; ii < 48; ii++) {
                            mean += tmpdata.ctrlitem[items].step[ii] / 48.;
                    }
                    switch(items)
                    {
                            case 0:
                                    ctrlprintfxy(68, 11, "^c1^CF%4.1f ° C   ^C0^c7",
max *  ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(68, 13, "^c1^CF%4.1f ° C   ^C0^c7",
min* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(68, 15, "^c1^CF%4.1f ° C   ^C0^c7",
mean* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(68, 17, "^c1^CF%4.1f ° C   ^C0^c7",
            (float)tmpdata.ctrlitem[items].step[step]* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    break;
                            case 1:
                                    ctrlprintfxy(68,   11,   "^c1^CF%4.1f        %%
^C0^c7", max* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(68,   13,   "^c1^CF%4.1f        %%
^C0^c7", min* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(68,   15,   "^c1^CF%4.1f        %%
^C0^c7", mean* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(68,   17,   "^c1^CF%4.1f        %%
^C0^c7",
            (float)tmpdata.ctrlitem[items].step[step]* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    break;
                            case 2:
                                    ctrlprintfxy(68,   11,   "^c1^CF%6.1f   ppm^C0^c7",
max* ctrldata_header[osjk_env.c_haus].mul[items],
                                    ctrlprintfxy(68,   13,   "^c1^CF%6.1f   ppm^C0^c7",
min* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(68,   15,   "^c1^CF%6.1f   ppm^C0^c7",
mean* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(68, 17, "^c1^CF%6.1f ppm^C0^c7",
            (float)tmpdata ctrlitem[items].step[step]* ctrldata_header[osjk_env c_haus].mul[items]);
                                    break;
                            case 3:
                                    ctrlprintfxy(66,   11,   "^c1^CF%6.0fW/m₂  ^C0^c7",
max* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(66,   13,   "^c1^CF%6.0fW/m₂  ^C0^c7",
min* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(66,   15,   "^c1^CF%6.0fW/m₂  ^C0^c7",
mean* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    ctrlprintfxy(66, 17, "^c1^CF%6.0fW/m₂  ^C0^c7",
            (float)tmpdata.ctrlitem[items].step[step]* ctrldata_header[osjk_env.c_haus].mul[items]);
                                    break;
                    }
```

```
                                        ctrlprintfpxy(8*(14+step),216 - tmpdata.ctrlitem[items].step[step] *
scale[items] ,
tmpdata2.init[items] ? "^CA^A-^a^C0":"^CC^A-^a^C0");
                                scrollbar = (int)(464. / (ctrldata_header[osjk_env.c_haus].duration-1)
* (day))+24;
                                ctrlputspxy(scrollbar,282, "^CF^c0^A■^a^C0^c7");
                                calc_edays(ctrldata_header[osjk_env.c_haus].syear,
ctrldata_header[osjk_env.c_haus].smon,
ctrldata_header[osjk_env.c_haus].sday,
                                                                        day,  &cyr,
&cmon, &cday);
                                d.da_year = cyr;
                                d.da_mon = cmon;
                                d.da_day = cday;
                                getsunrise(d, &sh, &sm, &eh, &em);
                                ctrlprintfpxy(sh*16+115+sm*16/60-48,232,"^CE      ☼      ^C0");
                                ctrlprintfpxy(eh*16+115+em*16/60-48,232,"^C0      ☼      ^C0");
                                hsetcolor(BLACK);
                                ctrlprintfxy(3, 1, "^U%2d시 %02d분  < %s지정 >  [ %3d일 / %3d
일 ]  [%4d년 %2d월 %2d일]^u",
                                                step / 2, step % 2 ? 30:0,
                                                ctrldata_header[osjk_env.c_haus].individual  ?  "개
별":"일괄",
                                                day + 1,ctrldata_header[osjk_env.c_haus].duration,
d.da_year, d.da_mon, d.da_day);
                                // get key
                                key = hgetch();
                                // earse cursor
                                for(ii = 0; ii < 48; ii++)
                                        ctrlprintfpxy(8*(14+ii),216 - tmpdata.ctrlitem[items].step[ii]
* scale[items] ," ");
                                switch (key)
                                {
                                                case '+': case PGDNKEY: case '-': case
PGUPKEY: case HOMEKEY: case ENDKEY:
                                                case MOUKEY:
                                                        ctrlputspxy(scrollbar,282, "^C3■^C0");
                                                        break;
                                                case CR: case F2:
                                                        tmpdata.init[items] = true;
                                                        tmpdata2.init[items] = true;
                                                        fseek(fin,  (long)sizeof(ctrldata_header_t)
+ (long)sizeof(ctrldata_t) * (long)day, SEEK_SET);
                                                        fwrite(&tmpdata,  sizeof(ctrldata_t),  1,
fin);
                                                        osjk_env.haus[osjk_env.c_haus].edited  =
true;
                                                        break;
                                }
                                switch(key)
                                {
                                        case ESC:
                                                if(osjk_env.haus[osjk_env.c_haus].edited)
                                                        if(askanswer("현재 입력한 것을 취소하
십니까", false) != true)
                                                                break;
                                                fclose(fin);
copyfile("_ctrl_.$$$",osjk_env.haus[osjk_env.c_haus].ctrldata_fn);
                                                remove("_ctrl_.$$$");
                                                done = true;
                                                break;
                                        case '-':
                                                if(day > 0)
                                                        day --;
                                                redraw = true;
                                                break;
                                        case '+':
                                                                if(day                  <
```

```
ctrldata_header[osjk_env.c_haus].duration - 1)
                                        day++;
                                redraw = true;
                                break;
                        case UPARROW:
                                if(tmpdata.ctrlitem[items].step[step]  <  200.  /
scale[items] )
tmpdata.ctrlitem[items].step[step]++;
                                osjk_env.haus[osjk_env.c_haus].edited = true;
                                break;
                        case DOWNARROW:
                                if(tmpdata.ctrlitem[items].step[step] > 0)
tmpdata.ctrlitem[items].step[step]--;
                                osjk_env.haus[osjk_env.c_haus].edited = true;
                                break;
                        case LEFTARROW: case '4':
                                if(step > 0)
                                        step--;
                                break;
                        case RIGHTARROW: case '6':
                                if(step < 48-1)
                                        step++;
                                break;
                        case PGUPKEY:
                                if(day - 7 > 0)
                                        day -= 7;
                                else
                                        day = 0;
                                redraw = true;
                                break;
                        case PGDNKEY:
                                if(day          +          7          <
ctrldata_header[osjk_env.c_haus].duration)
                                        day += 7;
                                else
                                                day                     =
ctrldata_header[osjk_env.c_haus].duration - 1;
                                redraw = true;
                                break;
                        case HOMEKEY:
                                day = 0;
                                redraw = true;
                                break;
                        case ENDKEY:
                                day  =  ctrldata_header[osjk_env.c_haus].duration
- 1;
                                redraw = true;
                                break;
                        case '8':
                                for(ii=0; ii < 48; ii++)
                                        if(tmpdata.ctrlitem[items].step[ii] < 200.
/ scale[items] )
tmpdata.ctrlitem[items].step[ii]++;
                                osjk_env.haus[osjk_env.c_haus].edited = true;
                                break;
                        case '2':
                                for(ii=0; ii < 48; ii++)
                                        if(tmpdata.ctrlitem[items].step[ii] > 0)
tmpdata.ctrlitem[items].step[ii]--;
                                osjk_env.haus[osjk_env.c_haus].edited = true;
                                break;
                        case '7':
                                for(ii=0; ii <= step; ii++)
                                        if(tmpdata.ctrlitem[items].step[ii] < 200.
/ scale[items] )
tmpdata.ctrlitem[items].step[ii]++;
                                osjk_env.haus[osjk_env.c_haus].edited = true;
                                break;
                        case '1':
                                for(ii=0; ii <= step; ii++)
                                        if(tmpdata.ctrlitem[items].step[ii] > 0)
```

```
tmpdata.ctrlitem[items].step[ii]--;
                                                osjk_env.haus[osjk_env.c_haus].edited = true;
                                                break;
                                        case '9':
                                                for(ii=step; ii < 48; ii++)
                                                        if(tmpdata.ctrlitem[items].step[ii] < 200.
/ scale[items] )
tmpdata.ctrlitem[items].step[ii]++;
                                                osjk_env.haus[osjk_env.c_haus].edited = true;
                                                break;
                                        case '3':
                                                for(ii=step; ii < 48; ii++)
                                                        if(tmpdata.ctrlitem[items].step[ii] > 0)
tmpdata.ctrlitem[items].step[ii]--;
                                                osjk_env haus[osjk_env.c_haus].edited = true;
                                                break;
                                        case MOUKEY:
                                                ii = process_button(btn, 20);
                                                if(ii >= 0)
                                                        ungetxch(hkey[ii]);
                                                else
                                                if((mou_x > 14 && mou_x < 63) && (mou_py
< 224 && mou_py > 22)) {
                                                                step = mou_x - 15;
                                                                ctrlprintfpxy(8*(14+step),216 -
tmpdata.ctrlitem[items].step[step] * scale[items] ," ");
tmpdata.ctrlitem[items].step[step] = (224 - mou_py) / scale[items];
osjk_env.haus[osjk_env.c_haus].edited = true;
                                                } else
                                                if(mou_py > 282 && mou_py < 298)
                                                {
                                                        if(mou_px > 8 && mou_px < 24)
                                                                ungetxch('-');
                                                        else if(mou_px > 504 && mou_px <
520)
                                                                ungetxch('+');
                                                        else if(mou_px > 24 && mou_px <
504) {
                                                                day =(int)((float)(mou_px -
24) * (float)(ctrldata_header[osjk_env.c_haus].duration) / 480 );
                                                                redraw = true;
                                                        }
                                                }
                                                break;
                                        case F10:
sprintf(buf,"%3d",ctrldata_header[osjk_env c_haus].error[items]),
                                                if(win_getdata("오차의 범위를 입력하세요", buf,
"999", ENGONLY))
ctrldata_header[osjk_env.c_haus].error[items] = atoi(buf);
                                                redraw = true;
                                                break;
                                        case F2:
                                                // save
                                                if(!ctrldata_header[osjk_env.c_haus].individual) {
                                                        putmsg("[일괄지정] 나머지 작기를 모두
동일하게 설정합니다", true);
                                                        tmpdata.init[0] = tmpdata.init[1] = true;
                                                        fseek(fin, 0, SEEK_SET);
fwrite(&ctrldata_header[osjk_env.c_haus], sizeof(ctrldata_header_t), 1, fin);
                                                        for(ii = day; ii <
ctrldata_header[osjk_env.c_haus].duration; ii++)
                                                        {
fseek(fin,(long)sizeof(ctrldata_header_t) + (long)sizeof(ctrldata_t) * (long)ii, SEEK_SET);
                                                                fwrite(&tmpdata,
sizeof(ctrldata_t), 1, fin);
                                                        }
                                                        closemsg();
                                                }
```

```
                                        fclose(fin);
                                        remove("_ctrl_.$$$");
                                        osjk_env.haus[osjk_env.c_haus].edited = true;
                                        port_id = osjk_env.c_haus + 1;
                                        dnload_ctrldata(osjk_env.c_haus);
                                        done = true;
                                        break;
                                case F5:
                                        ctrldata_header[osjk_env.c_haus].individual
false;
                                        break;
                                case F6:
                                        ctrldata_header[osjk_env.c_haus].individual
true;
                                        break;
                                case ALT_1:
                                case ALT_2:
                                case ALT_3:
                                case ALT_4:
                                        disp_button(&btn[16+items], false);
                                        switch(key)
                                        {
                                                case ALT_1: items = 0
break;
                                                case ALT_2: items = 1
break;
                                                case ALT_3: items = 2
break;
                                                case ALT_4: items = 3
break;
                                        }
                                        redraw = true;
                                        break;
                                }
                        }
                } else
                        puterror("작기화일을 열 수가 없습니다");

                allow_darg(false);

                for(ii = 0; ii < 20; ii++)
                        close_button(&btn[ii]);
                popcolor();
                close_window(&info_w);
        }
        restorescreen();
 return false;
}

int edit_waterctrldata(void)
{
 ctrldata_t tmpdata;
 FILE *fin;
 mywindow_t info_w;
 button_t btn[16];
 int hkey[16] =
        { LEFTARROW, RIGHTARROW, UPARROW, DOWNARROW,
                '-', '+', PGUPKEY, PGDNKEY, HOMEKEY, ENDKEY,
                F5, F6, TAB, ESC, F2, CR},
 int ii, items = 0, key, done = false, redraw = true, step = 0, day = 0,
                system = 0;
 int pos[16][4] =
                {
                        {27, 26, 31, 27},
                        {31, 26, 35, 27},
                        {35, 26, 39, 27},
                        {39, 26, 43, 27},

                        {43, 26, 53, 27},
                        {53, 26, 63, 27},
                        {63, 26, 73, 27},
                        {73, 26, 83, 27},
                        {83, 26, 93, 27},
                        {93, 26, 103, 27},

                        {27, 27, 40, 29},
                        {40, 27, 53, 29},
```

```
                    {53, 27, 63, 29},

                    {63, 27, 73, 29},
                    {73, 27, 83, 29},
                    {83, 27, 103, 29},
        };
int /*scale[4] = {4, 2, 2, 4},*/ scrollbar, cyr, cmon, cday;
struct date d;
int sh, sm, eh, em;
char *text[17] =
{"Ц","Ш","▲","▼",
        "어제 -","내일 +", "앞쪽 PgUp","뒤쪽 PgDn","처음 Home","끝 End",
        "일괄지정 F5","개별지정 F6","켬/끔 TAB","취소 ESC","완료 F2", "저장 ⊟",
};

char buf[80];

if(!ctrldata_header[osjk_env.c_haus].init) {
        puterror("작기가 설정되어 있지 않습니다. 먼저 작기를 설정하세요");
        return false;
}

for(ii = 0; ii < 16; ii++) {
        set_button(&btn[ii], pos[ii][0],pos[ii][1],pos[ii][2],pos[ii][3],
        WHITE, BLACK, LIGHTGRAY, LIGHTGREEN, BLACK,
        false, false, text[ii]);
}
sprintf(buf, "제 %d 양액제어시스템 제어정보 편집기", osjk_env.c_haus-3);
setmywindow(&info_w, 25, 3, 104, 31, WHITE, BLACK, GREEN, LIGHTGRAY,
                                WHITE, BLACK, LIGHTGRAY, false, true,
                                buf, false);
memset(&tmpdata, 0, sizeof tmpdata);
savescreen();
if(open_window(&info_w)) {
        pushcolor();
//      hsetbkcolor(BLUE);
//      hsetcolor(WHITE);
        allow_darg(true);

//      hsolidbarpxy(-208,-63, -511, -280, LIGHTGRAY);

//      rectangle_3d(208, 63, 511, 280, WHITE, BLACK);
        // file open
        mou_wait(true);
        copyfile(osjk_env.haus[osjk_env.c_haus].ctrldata_fn, "_ctrl_.$$$");
        mou_wait(false);

        fin = fopen(osjk_env.haus[osjk_env.c_haus].ctrldata_fn, "r+b");
        if(fin != NULL)
        {
                for(ii = 0; ii < 16; ii++)
                        open_button(&btn[ii]);

                ctrlputspxy(8,282+32,"^C3^cF^RЦ^r[redacted]^RШ^r^C0^c7");
                if(ctrldata_header[osjk_env.c_haus].elapsed)
                        day = ctrldata_header[osjk_env.c_haus].elapsed-1;
                else
                        day = 0;

                hhline(30,260+32,470, BLACK);
                hprintfpxy(30, 262+32,"시각 →");
                for(ii = 0; ii < 48; ii++) {
                        hvline(ii*8+115, ii % 2 ? 255+32:250+32,ii % 2 ? 5:10, BLACK);
                        if(!(ii % 4))
                                hprintfpxy(ii*8+108, 262+32,"%02d", ii / 2);
                }

                ctrlprintfxy(2,2,"^C1^cF     pH  산[redacted]열
                ^C0^c7");
                ctrlprintfxy(2,4,"^C1^cF     EC  저[redacted]고
            mS ^C0^c7");
                ctrlprintfxy(2,6,"^C1^cF   TEMP  저[redacted]고
            ˚C ^C0^c7");
                ctrlprintfxy(2,8,"^C1^cF     DO  저[redacted]고
           ppm ^C0^c7");
```

```
                for(ii = 0; ii < 4; ii++)
                {
//                              ctrlprintfxy(2,8+2*ii,"^C1^cFPROGRAM %d  ■■■■■■■■■■
■■■■■■■■■■■■■■■■^C0^c7",ii+1);
                              ctrlprintfxy(2,10+2*ii,"^C1^cFPROGRAM %d  ■■■■■■■■■■
■■■■■■■■■■■■■■■■^C0^c7",ii+1);
                }

                 fseek(fin,  (long)sizeof(ctrldata_header_t) + (long)sizeof(ctrldata_t) * (long)day,
SEEK_SET);
                fread(&tmpdata, sizeof(ctrldata_t), 1, fin);
                while(!done)
                {
                        if(redraw) {
                                ctrlprintfxy(67,2,"^C1^cF%4.1f",  tmpdata.ctrlitem[0].step[0] / 10.);
//MAX 14
                                ctrlputspxy(88+tmpdata.ctrlitem[0].step[0] * 2.85, 16,
                                                                                items ==
0? "^CC^c1^A | ^a^C0^c7":"^CF^c1^A | ^a^C0^c7");
                                ctrlprintfxy(67,4,"^C1^cF%4.1f",tmpdata.ctrlitem[0].step[1] / 10.);
//MAX 5
                                ctrlputspxy(88+tmpdata.ctrlitem[0].step[1] * 8., 48,
                                                                                items ==
1? "^CC^c1^A | ^a^C0^c7":"^CF^c1^A | ^a^C0^c7");
                                ctrlprintfxy(67,6,"^C1^cF%4.1f",tmpdata.ctrlitem[0].step[2] / 10.);
//MAX 50
                                ctrlputspxy(88+tmpdata.ctrlitem[0].step[2] * .8, 80,
                                                                                items ==
2? "^CC^c1^A | ^a^C0^c7":"^CF^c1^A | ^a^C0^c7");
                                ctrlprintfxy(67,8,"^C1^cF%4.1f",tmpdata.ctrlitem[0].step[3] / 10.);
//MAX 14
                                ctrlputspxy(88+tmpdata.ctrlitem[0].step[3] * 2.85, 112,
                                                                                items ==
3? "^CC^c1^A | ^a^C0^c7":"^CF^c1^A | ^a^C0^c7");

                                for(ii = 0; ii < 4; ii++)
                                {

ctrlprintfxy(64,10+2*ii,"^C1^cF%02d:%02d>%04d-%04d",

        tmpdata.ctrlitem[system].step[ii+4] / 2,

        tmpdata.ctrlitem[system].step[ii+4] % 2 ? 30:0,

        tmpdata.ctrlitem[system].step[ii+8],

        tmpdata.ctrlitem[system].step[ii+12] );
                                                        ctrlputspxy(108+tmpdata.ctrlitem[system].step[ii+4]
* 8, /*112*/144+ 32*ii,

items == ii+4? "^CC^c1^A | ^a^C0^c7":"^CF^c1^A | ^a^C0^c7");
                                }

        hprintfxy(2,26,"[pH]-[.%1d]-[EC]-[.%02d]-[Temp]-[%2d]    Ц F10",
                        ctrldata_header[osjk_env.c_haus].error[0],
                        ctrldata_header[osjk_env.c_haus].error[1],
                        ctrldata_header[osjk_env.c_haus].error[2]);
                        }
                        // draw cursor
                         scrollbar = (int)(464. / (ctrldata_header[osjk_env.c_haus].duration-1) *
(day))+24;
                        ctrlputspxy(scrollbar/*282*/314, "^CF^c0^A■^a^C0^c7");

                        calc_edays(ctrldata_header[osjk_env.c_haus].syear,

ctrldata_header[osjk_env.c_haus].smon,

ctrldata_header[osjk_env.c_haus].sday,
                                                                        day, &cyr, &cmon,
&cday);
                        d.da_year = cyr;
                        d.da_mon = cmon;
                        d.da_day = cday;
                        getsunrise(d, &sh, &sm, &eh, &em);
                        ctrlprintfpxy(sh*16+115+sm*16/60-48,/*232*/264,"^CE      ☼      ^C0");
                        ctrlprintfpxy(eh*16+115+em*16/60-48,/*232*/264,"^C0      ☼      ^C0");
```

```
                        hsetcolor(BLACK);

                        if(items > 3)
                                step = tmpdata.ctrlitem[system].step[items];

                        ctrlprintfxy(3, 1, "%2d시 %02d분  < %s지정 >  [ %3d일 / %3d일 ]  [%4d
년 %2d월 %2d일]",
                                step / 2, step % 2 ? 30:0,
                                ctrldata_header[osjk_env.c_haus].individual ? "개별":"일관",
                                day                    ],ctrldata_header[osjk_env.c_haus].duration,
d.da_year, d.da_mon, d.da_day);

                        // get key
                        key = hgetch();

                        ctrlputspxy(88+tmpdata.ctrlitem[0].step[0] * 2.85, 16,
                                                                        "^C1^cF■^C0^c7");
                        ctrlputspxy(88+tmpdata.ctrlitem[0].step[1] * 8., 48,
                                                                        "^C1^cF■^C0^c7");
                        ctrlputspxy(88+tmpdata.ctrlitem[0].step[2] * .8, 80,
                                                                        "^C1^cF■^C0^c7");
                        ctrlputspxy(88+tmpdata.ctrlitem[0].step[3] * 2.85, 112,
                                                                        "^C1^cF■^C0^c7");
                        for(ii = 0; ii < 4; ii++)
                        {
                                ctrlputspxy(108+tmpdata.ctrlitem[system].step[ii+4]*8,
/*112*/144+ 32*ii,
                                                                        "^C1^cF■^C0^c7");
                        }

                        switch (key)
                        {
                                case '+': case PGDNKEY: case '-': case PGUPKEY: case
HOMEKEY: case ENDKEY:
                                case MOUKEY:
                                        ctrlputspxy(scrollbar,/*282*/314, "^C3■^C0"),
                                        break;
                                case CR:
                                        tmpdata.init[0] = true;
                                        fseek(fin,      (long)sizeof(ctrldata_header_t)
(long)sizeof(ctrldata_t) * (long)day, SEEK_SET);
                                        fwrite(&tmpdata, sizeof(ctrldata_t), 1, fin);
                                        osjk_env.haus[osjk_env.c_haus].edited = true;
                                        break;
                                case TAB:
                                        if(items > 3)
                                        {
                                                sprintf(buf,"[%04d-%04d]",
tmpdata.ctrlitem[system].step[items+4],
                tmpdata.ctrlitem[system].step[items+8]);
                                                win_getdata("동작시간/휴지시간을  분단
위로 입력하세요",
                        buf,"[9999-9999]",ENGONLY);
                                                buf[5] = buf[10] = 0;
                                                tmpdata.ctrlitem[system].step[items+4] =
atoi(buf+1);
                                                tmpdata.ctrlitem[system].step[items+8] =
atoi(buf+6);
                                        }
                                        break;
                        }
                        switch(key)
                        {
                                case ESC:
                                        if(osjk_env.haus[osjk_env.c_haus].edited)
                                                if(askanswer("현재 입력한 것을 취소하십니까",
false) != true)
                                                        break;
                                        fclose(fin);

copyfile("_ctrl_.$$$",osjk_env.haus[osjk_env.c_haus].ctrldata_fn);
                                        remove("_ctrl_.$$$");
                                        done = true;
                                        break;
                                case '-':
```

```
                        if(day > 0)
                                day --;
                        redraw = true;
                        break;
                case '1':
                        if(day < ctrldata_header[osjk_env.c_haus].duration - 1)
                                day++;
                        redraw = true;
                        break;
                case UPARROW:
                        items = (items + 7) % 8;
                        osjk_env.haus[osjk_env.c_haus].edited = true;
                        break;
                case DOWNARROW:
                        items = (items + 1) % 8;
                        osjk_env.haus[osjk_env.c_haus].edited = true;
                        break;
                case LEFTARROW: case '4':
                        switch(items)
                        {
                                case 0:
                                        if(tmpdata.ctrlitem[0].step[items] > 0)
tmpdata.ctrlitem[0].step[items]--;
                                        break;
                                case 1:
                                        if(tmpdata.ctrlitem[0].step[items] > 0)
tmpdata.ctrlitem[0].step[items]--;
                                        break;
                                case 2:
                                        if(tmpdata.ctrlitem[0].step[items] > 4)
                                                tmpdata.ctrlitem[0].step[items]
-= 5;
                                        break;
                                case 3:
                                        if(tmpdata.ctrlitem[0].step[items] > 0)
tmpdata.ctrlitem[0].step[items]--;
                                        break;
                                case 4: case 5: case 6: case 7:
                                        if(tmpdata.ctrlitem[system].step[items]
> 0)
tmpdata.ctrlitem[system].step[items]--;
                                        break;
                        }
                        redraw = true;
                        break;
                case RIGHTARROW: case '6':
                        switch(items)
                        {
                                case 0:
                                        if(tmpdata.ctrlitem[0].step[items] < 140)
tmpdata.ctrlitem[0].step[items]++;
                                        break;
                                case 1:
                                        if(tmpdata.ctrlitem[0].step[items] < 50)
tmpdata.ctrlitem[0] step[items]++;
                                        break;
                                case 2:
                                        if(tmpdata.ctrlitem[0].step[items] < 500)
                                                tmpdata.ctrlitem[0].step[items]
+= 5;
                                        break;
                                case 3:
                                        if(tmpdata.ctrlitem[0].step[items] < 140)
tmpdata.ctrlitem[0] step[items]++;
                                        break;
                                case 4: case 5: case 6: case 7:
                                        if(tmpdata.ctrlitem[system].step[items]
< 47)
tmpdata ctrlitem[system].step[items] ++.
```

```
                                                    break;
                                            }
                                            redraw = true;
                                            break;
                                    case PGUPKEY:
                                            if(day - 7 > 0)
                                                    day -= 7;
                                            else
                                                    day = 0;
                                            redraw = true;
                                                fseek(fin,      (long)sizeof(ctrldata_header_t)      +
(long)sizeof(ctrldata_t) * (long)day, SEEK_SET);
                                            fread(&tmpdata, sizeof(ctrldata_t), 1, fin);
                                            break;
                                    case PGDNKEY:
                                            if(day + 7 < ctrldata_header[osjk_env.c_haus].duration)
                                                    day += 7;
                                            else
                                                    day  =  ctrldata_header[osjk_env.c_haus].duration
- 1;
                                            redraw = true;
                                                fseek(fin,      (long)sizeof(ctrldata_header_t)      +
(long)sizeof(ctrldata_t) * (long)day, SEEK_SET);
                                            fread(&tmpdata, sizeof(ctrldata_t), 1, fin);
                                            break;



                                    case F10: // edit error
        sprintf(buf, "[pH]-[.%1d]-[EC]-[.%02d]-[Temp]-[%02d]",
                        ctrldata_header[osjk_env.c_haus].error[0],
                        ctrldata_header[osjk_env.c_haus].error[1],
                        ctrldata_header[osjk_env.c_haus].error[2]);
        hgotoxy(2,26);
        _showcursor = true;
        _insertmode = false;
        if(hgetdata(buf,"[**]-[.9]-[**]-[.99]-[****]-[99]") == CR)
        {               //  01234567890123456789012345678901234567890
          ctrldata_header[osjk_env.c_haus].error[0] = atoi(buf+7);
          ctrldata_header[osjk_env.c_haus].error[1] = atoi(buf+17);
          ctrldata_header[osjk_env.c_haus].error[2] = atoi(buf+29);
                                        redraw = true;
        }
        _insertmode = true;
        _showcursor = false;
                                            break;



                                    case HOMEKEY:
                                            day = 0;
                                            redraw = true;
                                                fseek(fin,      (long)sizeof(ctrldata_header_t)      +
(long)sizeof(ctrldata_t) * (long)day, SEEK_SET);
                                            fread(&tmpdata, sizeof(ctrldata_t), 1, fin);
                                            break;
                                    case ENDKEY:
                                            day = ctrldata_header[osjk_env.c_haus].duration - 1;
                                            redraw = true;
                                                fseek(fin,      (long)sizeof(ctrldata_header_t)      +
(long)sizeof(ctrldata_t) * (long)day, SEEK_SET);
                                            fread(&tmpdata, sizeof(ctrldata_t), 1, fin);
                                            break;
                                    case MOUKEY:
                                            ii = process_button(btn, 16);
                                            if(ii >= 0)
                                                    ungetxch(hkey[ii]);
                                            else
                                            if(mou_py > /*282*/314 && mou_py < /*298*/330)
                                            {
                                                    if(mou_px > 8 && mou_px < 24)
                                                            ungetxch('-');
                                                    else if(mou_px > 504 && mou_px < 520)
                                                            ungetxch('+');
```

```
                                        else if(mou_px > 24 && mou_px < 504) {
                                                day  =(int)((float)(mou_px  -  24)  *
(float)(ctrldata_header[osjk_env.c_haus].duration) / 480.);
                                                redraw = true;
                                        }
                                } else
                                if(mou_y % 2 == 0 && mou_y < /*15*/17)
                                        items = mou_y / 2 - 1;
                                break;

                        case F2:
                                // save
                                if(!ctrldata_header[osjk_env.c_haus].individual) {
                                        putmsg("[일괄지정] 나머지 작기를 모두 동일하게
신성합니다", true);
                                        tmpdata.init[0] = tmpdata.init[1] = true;
                                        tmpdata.init[2] = tmpdata.init[3] = true;
                                        fseek(fin, 0, SEEK_SET);
                                                fwrite(&ctrldata_header[osjk_env.c_haus],
sizeof(ctrldata_header_t), 1, fin);

                                                for(ii  =  day;  ii  <
ctrldata_header[osjk_env c_haus] duration; ii++)
                                        {
                                                fseek(fin,(long)sizeof(ctrldata_header_t)
+ (long)sizeof(ctrldata_t) * (long)ii, SEEK_SET);
                                                fwrite(&tmpdata,  sizeof(ctrldata_t),  1,
fin);
                                        }
                                        closemsg();
                                }
                                fclose(fin);
                                remove("_ctrl_.$$$");
                                osjk_env.haus[osjk_env.c_haus].edited = true;
                                port_id = osjk_env.c_haus + 1;
                                dnload_ctrldata(osjk_env.c_haus);
                                done = true;
                                break;
                        case F5:
                                ctrldata_header[osjk_env c_haus].individual = false;
                                break;
                        case F6:
                                ctrldata_header[osjk_env.c_haus].individual = true;
                                break;
                        case ALT_1:
                                items = 0;
                                redraw = true;
                                break;
                        case ALT_2:
                                items = 1;
                                redraw = true;
                                break;
                        case ALT_3:
                                items = 2;
                                redraw = true;
                                break;
                        case ALT_4:
                                items = 3;
                                redraw = true;
                                break;
                        case ALT_5:
                                items = 4;
                                redraw = true;
                                break;
                        case ALT_6:
                                items = 5;
                                redraw = true;
                                break;
                        case ALT_7:
                                items = 6;
                                redraw = true;
                                break;
                        case ALT_8:
                                items = 7;
                                redraw = true;
                                break;
/*
```

```
                                        case ALT_S:
                                                disp_button(&btn[system+16], false);
                                                system = 0;
                                                disp_button(&btn[system+16], true);
                                                break;
                                        case ALT_N:
                                                disp_button(&btn[system+16], false);
                                                system = 1;
                                                disp_button(&btn[system+16], true);
                                                break;
                                        case ALT_R:
                                                disp_button(&btn[system+16], false);
                                                system = 2;
                                                disp_button(&btn[system+16], true);
                                                break;
*/
                                }
                        }
                } else
                        puterror("작기화일을 열 수가 없습니다");

                allow_darg(false);

                for(ii = 0; ii < 16; ii++)
                        close_button(&btn[ii]);
                popcolor();
                close_window(&info_w);
        }
        restorescreen();
        return false;
}


void disp_sensor(int pos)
{
        int i,j;
        char buf[80];

        if(disable_disp_sensor)
                return;

        disp_data_pos = pos;

        if(!ctrldata_header[osjk_env.c_haus].init)
        {
                sprintf(buf,"%d 온실은 작기설정이 안 되었습니다", osjk_env.c_haus+1);
                putmsg(buf, 1);
                sleep(1);
                closemsg();
                return;
        }
        if(!ctrldata_header[osjk_env.c_haus].started)
        {
                sprintf(buf,"%d 온실은 제어를 시작하지 않았습니다.", osjk_env.c_haus+1)
                putmsg(buf, 1);
                sleep(1);
                closemsg();
                for(i=0; i < MAXITEM; i++)
                        for(j = 0; j < 48; j++)
                                _ctrldata_.step[i][j] = 0;
        } else if(ctrldata_header[osjk_env.c_haus].stoped) {
                sprintf(buf,"%d 온실은 제어가 이미 끝났습니다.", osjk_env.c_haus+1);
                putmsg(buf, 1);
                sleep(1);
                closemsg();
                for(i=0; i < MAXITEM; i++)
                        for(j = 0; j < 48; j++)
                                _ctrldata_.step[i][j] = 0;
        } else
        {
                pushcolor();
                hsetcolor(WHITE);
                hsetbkcolor(DARKGRAY);

                if(osjk_env.c_haus < 4)
                {
```

```
        if(osjk_env.c_haus > 0)
        {
                                disp_temp(0);
                                disp_humi(1);
                                disp_co2(2);
        } else
        {
                                disp_temp_21(0);
                                disp_humi(1);
                                disp_co2(2);
        }
                        } else
                        {
                                disp_ec(0);
                                disp_ph(1);
                                disp_temp_hy(2);
                        }
                        popcolor();
                }
}

void disp_weather_station(void)
{
        int i;
        static min = 0;
        struct time t;
        gettime(&t);
        if(t.ti_min != min)
        {
                min = t.ti_min;
                if(min % 2)
                        loop();
                else
                        get_loop_data();
        } else return;

        if(display_disable)
                return;

        pushcolor();
        hsetcolor(WHITE);
        hsetbkcolor(BLUE);
        ctrlprintfxy(-109, -3,"^R        풍  향      ^r");
        setcolor(WHITE);
        circle(940, 140, 79);
        setcolor(LIGHTGRAY);
        for(i = 18; i > 0; i--)
                circle(940, 140, 60+i);
        setcolor(WHITE);
        circle(940, 140, 60);
        hprintfpxy(-932, -62, "북");
        hprintfpxy(-862, -132, "서");
        hprintfpxy(-1002, -132, "동");
        hprintfpxy(-932, -202, "남");
        ctrlprintfxy(-109, -19,"^R        풍  속      ^r");
        ctrlprintfxy(-109, -22,"^R        온  도      ^r");
        ctrlprintfxy(-109, -25,"^R        습  도      ^r");
        ctrlprintfxy(-109, -28,"^R        기  압      ^r");
        ctrlprintfxy(-109, -31,"^R        일  사      ^r");
        ctrlprintfxy(-109, -34,"^R        강수량      ^r");
        ctrlprintfxy(-109, -37,"^R        현재날씨    ^r");
        hsetbkcolor(CYAN);
        setfillstyle(SOLID_FILL, DARKGRAY);
        floodfill(940, 140, WHITE);
//      pieslice(940, 140, (450- last_wd) % 360, (450-last_wd + 1) % 360, 55)
        wsensor.sdata.wd %= 360;
        setcolor(LIGHTGREEN);
        setfillstyle(SOLID_FILL, LIGHTGREEN);
        setbkcolor(LIGHTRED);
        pieslice(940, 140, (449-wsensor.sdata.wd) % 360,

(450-wsensor.sdata.wd + 1)%360, 55);
        pieslice(940, 140, (440-wsensor.sdata.wd) % 360,

(460-wsensor.sdata.wd + 1)%360, 20);

        hprintfxy(-111,-16,"풍향:%-4s (%3d ° )",
```

```
                                                wind_direction[dicision_wd(wsensor.sdata.wd)],
                                                wsensor.sdata.wd);
        hprintfxy(-113,-20,"%6.1f m/sec",wsensor.sdata.ws*1609.3 / 3600.);
        hprintfxy(-113,-23,"%6.1f °C",(wsensor.sdata.out_t /10.- 32.)* 5.0 / 9.0);
        hprintfxy(-113,-26,"%8d %%",wsensor.sdata.out_h);
        hprintfxy(-113,-29,"%8.2f mb",wsensor.sdata.atm * .03386);
        hprintfxy(-113,-32,"%8d W/m² ",wsensor.sdata.light);
        hprintfxy(-113,-35,"%8d mm",(int)(wsensor.sdata.rain * 0.254));
        hsetcolor(BLACK);
        hprintfxy(-114,-38,"%8s", flag.win_status.rain ? "비가와요":".....");

        dnload_stationdata();
/*
        hprintfxy(-110,-37,"%c%c%c%c%c%c%c%c %3d",
                                                wsensor.sdata.dmy5 & 0x80 ? '1':'0',
                                                wsensor.sdata.dmy5 & 0x40 ? '1':'0',
                                                wsensor.sdata.dmy5 & 0x20 ? '1':'0',
                                                wsensor.sdata.dmy5 & 0x10 ? '1':'0',
                                                wsensor.sdata.dmy5 & 0x08 ? '1':'0',
                                                wsensor.sdata.dmy5 & 0x04 ? '1':'0',
                                                wsensor.sdata.dmy5 & 0x02 ? '1':'0',
                                                wsensor.sdata.dmy5 & 0x01 ?
'1':'0',wsensor.sdata.dmy5 );
        hprintfxy(-110,-38,"%c%c%c%c%c%c%c%c %3d",
                                                wsensor.sdata.dmy6 & 0x80 ? '1':'0',
                                                wsensor.sdata.dmy6 & 0x40 ? '1':'0',
                                                wsensor.sdata.dmy6 & 0x20 ? '1':'0',
                                                wsensor.sdata.dmy6 & 0x10 ? '1':'0',
                                                wsensor.sdata.dmy6 & 0x08 ? '1':'0',
                                                wsensor.sdata.dmy6 & 0x04 ? '1':'0',
                                                wsensor.sdata.dmy6 & 0x02 ? '1':'0',
                                                wsensor.sdata.dmy6 & 0x01 ? '1':'0'
,wsensor.sdata.dmy6 );
*/
//      hprintfxy(109,41,"%6d",wsensor.sdata.dmy6 );
        popcolor();
}


void disp_temp(int win)
{
        int i;
        char buf[50];
//      if(draw_new)
        if(ctrldata_header[osjk_env.c_haus].init &&
                        ctrldata_header[osjk_env.c_haus].started &&
                        !ctrldata_header[osjk_env.c_haus].stoped)
        {
                sprintf(buf,"%-s 온 도", hausname[osjk_env.c_haus]);
                msgbox(disp_pos[win][0],disp_pos[win][1],disp_pos[win][2],disp_pos[win][3],
                                                WHITE, BLACK, LIGHTGRAY, DARKGRAY,
WHITE,CYAN, buf, 0);

                pushcolor();
                hsetbkcolor(DARKGRAY);
                hsetcolor(WHITE);
                hhline(disp_pos[win+3][0],disp_pos[win+3][1],disp_pos[win+3][2], WHITE);
                hvline(disp_pos[win+3][0],disp_pos[win+3][1]-120,120, WHITE);

                for(i = 0; i < 480; i += 20) {
                        hvline(-(disp_pos[win+3][0]+i),
                                                -(disp_pos[win+3][1]), i % 40 ? 5:10,
WHITE);
                        if(!(i % 40))
                                hprintfpxy(-(disp_pos[win+3][0]+i-8),

-(disp_pos[win+3][1]+4) ,"%02d", i / 20);
                }

                for(i = 0; i <= 50; i += 10) {
                        hhline(-(disp_pos[win+3][0] - 8),
                                                -(disp_pos[win+3][1] - i * 2), 10,
WHITE);
                        hprintfpxy(-(disp_pos[win+3][0]-64),
                                                -(disp_pos[win+3][1]
- i * 2-8), "%2d ° C", i);
                }
```

```
                load2ctrldata(osjk_env.c_haus);
                for(i=0; i < 48 && ctrldata_header[osjk_env.c_haus].init; i++)
                {
                        hhline(-(disp_pos[win+3][0]+(float)i * 10.),
                                                        -(disp_pos[win+3][1]
-_ctrldata_.step[0][i] * 2), 10,
                                                        YELLOW);
                }

                hprintfxy(-(disp_pos[win][0]+74), -(disp_pos[win][1]+2 ),"%5.2f ° C",
                                        (float)sensordata[osjk_env.c_haus][7][disp_data_pos]    /
100.);

                setcolor(LIGHTRED);
                for(i=1; i <= disp_data_pos; i++)
                        line( disp_pos[win+3][0] + i,
                                                                disp_pos[win+3][1]
-(int)(sensordata[osjk_env.c_haus][7][i-1] / 50.),
                                                        disp_pos[win+3][0] + i+1,
                                                        disp_pos[win+3][1]
-(int)(sensordata[osjk_env.c_haus][7][i] / 50.));
                popcolor();
        } else
                agc_mre();
}

void disp_temp_21(int win)
{
        int i;
        char buf[50];
//      if(draw_new)
        if(ctrldata_header[osjk_env.c_haus].init &&
                        ctrldata_header[osjk_env.c_haus].started &&
                        !ctrldata_header[osjk_env.c_haus].stoped)
        {
                sprintf(buf,"%-s  온도1(하늘색)/온도2(보라색)", hausname[osjk_env.c_haus]);
                msgbox(disp_pos[win][0],disp_pos[win][1],disp_pos[win][2],disp_pos[win][3],
                                                WHITE,  BLACK,  LIGHTGRAY,  DARKGRAY,
WHITE,CYAN, buf, 0);

                pushcolor();
                hsetbkcolor(DARKGRAY);
                hsetcolor(WHITE);
                hhline(disp_pos[win+3][0],disp_pos[win+3][1],disp_pos[win+3][2], WHITE);
                hvline(disp_pos[win+3][0],disp_pos[win+3][1]-120,120, WHITE);

                for(i = 0; i < 480; i += 20) {
                        hvline(-(disp_pos[win+3][0]+i),
                                                        -(disp_pos[win+3][1]), i % 40 ? 5:10,
WHITE);
                        if(!(i % 40))
                                hprintfpxy(-(disp_pos[win+3][0]+i-8),

-(disp_pos[win+3][1]+4) ,"%02d", i / 20);
                }

                for(i = 0; i <= 50; i += 10) {
                        hhline(-(disp_pos[win+3][0] - 8),
                                                        -(disp_pos[win+3][1]  -  i  *  2),  10,
WHITE);
                        hprintfpxy(-(disp_pos[win+3][0]-64),
                                                                -(disp_pos[win+3][1]
- i * 2-8), "%2d ° C", i);
                }

                load2ctrldata(osjk_env.c_haus);
                for(i=0; i < 48 && ctrldata_header[osjk_env.c_haus].init; i++)
                {
                        hhline(-(disp_pos[win+3][0]+(float)i * 10.),
                                                        -(disp_pos[win+3][1]
-_ctrldata_.step[0][i] * 2), 10,
                                                        MAGENTA);
                }
    hsetcolor(LIGHTMAGENTA);

                hprintfxy(-(disp_pos[win][0]+74), -(disp_pos[win][1]+2 ),"T1:%5.2f ° C",
```

```
                                                    (float)sensordata[osjk_env.c_haus][7][disp_data_pos]    /
100.);

                    setcolor(LIGHTMAGENTA);
                    for(i=1; i <= disp_data_pos; i++)
                            line( disp_pos[win+3][0] + i,
                                                                    disp_pos[win+3][1]                -
(int)(sensordata[osjk_env.c_haus][7][i-1] / 50.),
                                                            disp_pos[win+3][0] + i+1,
                                                            disp_pos[win+3][1]                        -
(int)(sensordata[osjk_env.c_haus][7][i] / 50.));

        load2orchid();
//                  load2ctrldata(osjk_env.c_haus);
                    for(i=0; i < 48 && ctrldata_header[osjk_env.c_haus].init; i++)
                    {
                            hhline(-(disp_pos[win+3][0]+(float)i * 10.),
                                                                    -(disp_pos[win+3][1]              -
_ctrldata_.step[0][i] * 2), 10,
                                                                    CYAN);
                    }
        hsetcolor(LIGHTCYAN);
                    hprintfxy(-(disp_pos[win][0]+74), -(disp_pos[win][1]+4 ),"T2:%5.2f ˚C",
                                                            (float)sensordata[osjk_env.c_haus][6][disp_data_pos]    /
100 );

                    setcolor(LIGHTCYAN);
                    for(i=1; i <= disp_data_pos; i++)
                            line( disp_pos[win+3][0] + i,
                                                                    disp_pos[win+3][1]                -
(int)(sensordata[osjk_env.c_haus][6][i-1] / 50.),
                                                            disp_pos[win+3][0] + i+1,
                                                            disp_pos[win+3][1]                        -
(int)(sensordata[osjk_env.c_haus][6][i] / 50.));

                    popcolor();
            } else
                    agc_mre();
}


void disp_temp_hy(int win)
{
            int i;
            char buf[50];
//          if(draw_new)
            if(ctrldata_header[osjk_env c_haus].init &&
                            ctrldata_header[osjk_env.c_haus].started &&
                            !ctrldata_header[osjk_env.c_haus].stoped)
            {
                    sprintf(buf,"%-s 온 도", hausname[osjk_env.c_haus]);
                    msgbox(disp_pos[win][0],disp_pos[win][1],disp_pos[win][2],disp_pos[win][3],
                                                            WHITE,  BLACK,  LIGHTGRAY,  DARKGRAY,
WHITE,CYAN, buf, 0);

                    pushcolor();
                    hsetbkcolor(DARKGRAY);
                    hsetcolor(WHITE);

                    hhline(disp_pos[win+3][0],disp_pos[win+3][1],disp_pos[win+3][2], WHITE);
                    hvline(disp_pos[win+3][0],disp_pos[win+3][1]-120,120, WHITE);

                    for(i = 0; i < 480; i += 20) {
                            hvline(-(disp_pos[win+3][0]+i),
                                                                    -(disp_pos[win+3][1]), i % 40 ? 5:10,
WHITE);
                            if(!(i % 40))
                                    hprintfpxy(-(disp_pos[win+3][0]+i-8),

-(disp_pos[win+3][1]+4) ,"%02d", i / 20);
                    }

                    for(i = 0; i <= 50; i += 10) {
                            hhline(-(disp_pos[win+3][0] - 8),
                                                                    -(disp_pos[win+3][1]  -  i  *  2),  10,
WHITE);
                            hprintfpxy(-(disp_pos[win+3][0]-64),
```

```
                                                                    -(disp_pos[win+3][1]
- i * 2-8), "%2d ° C", i);
                }


                load2ctrldata(osjk_env.c_haus);
                for(i=0; i < 48 && ctrldata_header[osjk_env.c_haus].init; i++)
                {
                        hhline(-(disp_pos[win+3][0]+(float)i * 10.),
                                                                    -(disp_pos[win+3][1]                -
_ctrldata_.step[0][2] / 5.), 10,
                                                                    YELLOW);
                }

                hprintfxy(-(disp_pos[win][0]+74), -(disp_pos[win][1]+2 ),"%5.2f ° C",
                                                (float)sensordata[osjk_env.c_haus][7][disp_data_pos]    /
100 ),

                setcolor(LIGHTRED);
                for(i=1; i <= disp_data_pos; i++)
                        line( disp_pos[win+3][0] + i,
                                                                    disp_pos[win+3][1]                  -
(int)(sensordata[osjk_env.c_haus][7][i-1] / 50.),
                                                        disp_pos[win+3][0] + i+1,
                                                                    disp_pos[win+3][1]                  -
(int)(sensordata[osjk_env.c_haus][7][i] / 50.));
                popcolor();
        } else
                agc_mre();
}


void disp_humi(int win)
{
        int i;
//      if(draw_new)
        char buf[50];
//      if(draw_new)
        if(ctrldata_header[osjk_env.c_haus].init &&
                        ctrldata_header[osjk_env.c_haus].started &&
                        !ctrldata_header[osjk_env.c_haus].stoped)
        {
                sprintf(buf,"%-s 습 도", hausname[osjk_env.c_haus]);
                msgbox(disp_pos[win][0],disp_pos[win][1],disp_pos[win][2],disp_pos[win][3],
                                                        WHITE,  BLACK,  LIGHTGRAY,  DARKGRAY,
WHITE,CYAN, buf, 0);

                pushcolor();
                hsetbkcolor(DARKGRAY);
                hsetcolor(WHITE);

                hhline(disp_pos[win+3][0],disp_pos[win+3][1],disp_pos[win+3][2], WHITE);
                hvline(disp_pos[win+3][0],disp_pos[win+3][1]-120,120, WHITE);

                for(i = 0; i < 480; i += 20) {
                        hvline(-(disp_pos[win+3][0]+i),
                                                                    -(disp_pos[win+3][1]), i % 40 ? 5:10,
WHITE);
                        if(!(i % 40))
                                hprintfpxy(-(disp_pos[win+3][0]+i-8),

-(disp_pos[win+3][1]+4) ,"%02d", i / 20);
                }

                for(i = 0; i <= 50; i += 10) {
                        hhline(-(disp_pos[win+3][0] - 8),
                                                                    -(disp_pos[win+3][1]  -  i  *  2),  10,
WHITE);
                        hprintfpxy(-(disp_pos[win+3][0]-64),
                                                                                -(disp_pos[win+3][1]
- i * 2-8), "%3d %%", i*2);
                }

                load2ctrldata(osjk_env.c_haus);
                for(i=0; i < 48 && ctrldata_header[osjk_env.c_haus].init; i++)
                {
                        hhline(-(disp_pos[win+3][0]+(float)i * 10.),
```

```
                                                        -(disp_pos[win+3][1]                    -
_ctrldata_.step[1][i]), 10,
                                                        YELLOW);
                }

    hsetcolor(LIGHTRED);
                hprintfxy(-(disp_pos[win][0]+74), -(disp_pos[win][1]+4 ),"%6.1f %%(2)",
                                                (float)sensordata[osjk_env.c_haus][5][disp_data_pos]    /
100.);
    hsetcolor(LIGHTMAGENTA);
                hprintfxy(-(disp_pos[win][0]+74), -(disp_pos[win][1]+2 ),"%6.1f %%(1)",
                                                (float)sensordata[osjk_env.c_haus][5][disp_data_pos]    /
100.);

                setcolor(LIGHTRED);
                for(i=1; i <= disp_data_pos; i++)
                        line( disp_pos[win+3][0] + i,
                                                                disp_pos[win+3][1]                      -
(int)(sensordata[osjk_env.c_haus][5][i-1]/100.),
                                                        disp_pos[win+3][0] + i+1,
                                                        disp_pos[win+3][1]                              -
(int)(sensordata[osjk_env.c_haus][5][i]/100.));

                setcolor(LIGHTMAGENTA);
                for(i=1; i <= disp_data_pos; i++)
                        line( disp_pos[win+3][0] + i,
                                                                disp_pos[win+3][1]                      -
(int)(sensordata[osjk_env.c_haus][4][i-1]/100.),
                                                        disp_pos[win+3][0] + i+1,
                                                        disp_pos[win+3][1]                              -
(int)(sensordata[osjk_env.c_haus][4][i]/100.));
                popcolor();
        } // else
                // agc_mre();
}

void disp_co2(int win)
{
        int i;
        char buf[50];
        sprintf(buf,"%-s 이산화탄소", hausname[osjk_env.c_haus]);
        if(ctrldata_header[osjk_env.c_haus].init &&
                        ctrldata_header[osjk_env.c_haus].started &&
                        !ctrldata_header[osjk_env.c_haus].stoped)
        {
//      if(draw_new)
                msgbox(disp_pos[win][0],disp_pos[win][1],disp_pos[win][2],disp_pos[win][3],
                                                        WHITE,  BLACK,  LIGHTGRAY,  DARKGRAY,
WHITE,CYAN, buf, 0);
                pushcolor();
                hsetbkcolor(DARKGRAY);
                hsetcolor(WHITE);

                hhline(disp_pos[win+3][0],disp_pos[win+3][1],disp_pos[win+3][2], WHITE);
                hvline(disp_pos[win+3][0],disp_pos[win+3][1]-120,120, WHITE);

                for(i = 0; i < 480; i += 20) {
                        hvline(-(disp_pos[win+3][0]+i),
                                                        -(disp_pos[win+3][1]), i % 40 ? 5:10
WHITE);
                        if(!(i % 40))
                                hprintfpxy(-(disp_pos[win+3][0]+i-8),

-(disp_pos[win+3][1]+4) ,"%02d", i / 20);
                }

                for(i = 0; i <= 50; i += 10) {
                        hhline(-(disp_pos[win+3][0] - 8),
                                                        -(disp_pos[win+3][1] - i * 2), 10
WHITE);
                        hprintfpxy(-(disp_pos[win+3][0]-64),
                                                                -(disp_pos[win+3][1]
- i * 2-8), "%4d ppm", i*40);
                }

                load2ctrldata(osjk_env.c_haus);
                for(i=0; i < 48 && ctrldata_header[osjk_env.c_haus].init; i++)
```

```
                {
                        hhline(-(disp_pos[win+3][0]+(float)i * 10.),
                                                        -(disp_pos[win+3][1]
_ctrldata_.step[2][i]), 10,
                                                        YELLOW);
                }

                hprintfxy(-(disp_pos[win][0]+74), -(disp_pos[win][1]+2 ),"%4.0f ppm",
                                        (float)sensordata[osjk_env.c_haus][3][disp_data_pos] / 10.)

                setcolor(LIGHTRED);
                for(i=1; i <= disp_data_pos; i++)
                        line( disp_pos[win+3][0] + i,
                                                                disp_pos[win+3][1]
(int)(sensordata[osjk_env.c_haus][3][i-1] / 200.),
                                                        disp_pos[win+3][0] + i+1,
                                                        disp_pos[win+3][1]
(int)(sensordata[osjk_env.c_haus][3][i] / 200.));
                popcolor();
        } // else
                // agc_mre();
}

/*
void disp_light(int win)
{
        int i;
        msgbox(disp_pos[win][0],disp_pos[win][1],disp_pos[win][2],disp_pos[win][3],
                                                        WHITE, BLACK, LIGHTGRAY, DARKGRAY
WHITE,CYAN, "광", 0);
        hhline(170,(win ? 390:198),400, WHITE);
        hvline(170,(win ? 280:88),110, WHITE);

        for(i = 0, i < 360; i += 15) {
                hvline(i+180,(win ? 390:198) ,i % 80 ? 5:10, WHITE);
                if(!(i % 30))
                        hprintfpxy((i+172), (win ? 392:200),"%02d", i / 15);
        }
        for(i = 0, i <= 50; i += 10) {
                hhline(160, (win ? 384:192) - i * 2, 10, WHITE);
                hprintfpxy(100, (win ? 376 184) - i * 2, "%3d,000", i * 4);
        }
        hprintfxy(15, (win ? 17 5), "LUX");
        hprintfxy(70, win ? 17 5,"%6d LUX", (float)sensordata[osjk_env.c_haus][1][disp_data_pos] *
10);
        for(i=0; i <= disp_data_pos; i++)
                hputpixel(180 + i, (win ? 384:192) - sensordata[osjk_env.c_haus][5][i] / 200,
                                LIGHTGREEN);
}
*/

void disp_ec(int win)
{
        int i;
        char buf[50];
        sprintf(buf,"%-s  EC", hausname[osjk_env.c_haus]);
//      if(draw_new)
        if(ctrldata_header[osjk_env.c_haus].init &&
                        ctrldata_header[osjk_env.c_haus].started &&
                        !ctrldata_header[osjk_env.c_haus].stoped)
        {
                msgbox(disp_pos[win][0],disp_pos[win][1],disp_pos[win][2],disp_pos[win][3],
                                                        WHITE, BLACK, LIGHTGRAY, DARKGRAY,
WHITE,CYAN, buf, 0);
                pushcolor();
                hsetbkcolor(DARKGRAY);
                hsetcolor(WHITE);

                hhline(disp_pos[win+3][0],disp_pos[win+3][1],disp_pos[win+3][2], WHITE);
                hvline(disp_pos[win+3][0],disp_pos[win+3][1]-120,120, WHITE);

                for(i = 0; i < 480; i += 20) {
                        hvline(-(disp_pos[win+3][0]+i),
                                                        -(disp_pos[win+3][1]), i % 40 ? 5:10,
WHITE);
                        if(!(i % 40))
                                hprintfpxy(-(disp_pos[win+3][0]+i-8),
```

```
(disp_pos[win+3][1]+4) ,"%02d", i / 20);
                }

                for(i = 0; i <= 50; i += 10) {
                        hhline(-(disp_pos[win+3][0] - 8),
                                                        -(disp_pos[win+3][1] - i * 2), 10
WHITE);
                        hprintfpxy(-(disp_pos[win+3][0]-64),
                                                                -(disp_pos[win+3][1]
- i * 2-8), "%5.2f uS", i / 10.);
                }

                load2ctrldata(osjk_env.c_haus);
                for(i=0; i < 48 && ctrldata_header[osjk_env.c_haus].init; i++)
                {
                        hhline(-(disp_pos[win+3][0]+(float)i * 10.),
                                                        -(disp_pos[win+3][1]
-_ctrldata_.step[0][1] * 2), 10,
                                                        YELLOW);
                }

                hprintfxy(-(disp_pos[win][0]+74), -(disp_pos[win][1]+2 ),"%5.2f uS",
                                        (float)sensordata[osjk_env.c_haus][4][disp_data_pos]    /
1000.);

                setcolor(LIGHTRED);
                for(i=1; i <= disp_data_pos; i++)
                        line( disp_pos[win+3][0] + i,
                                                                disp_pos[win+3][1]
-(int)(sensordata[osjk_env.c_haus][4][i-1] /100.* 2),
                                                        disp_pos[win+3][0] + i+1,
                                                        disp_pos[win+3][1]
-(int)(sensordata[osjk_env.c_haus][4][i] /100.* 2));

                popcolor();
        }//  else
                // agc_mre();
}

//
// DO와 겹쳐그려야 함.....
//
void disp_ph(int win)
{
        int i;
        char buf[50];
        sprintf(buf,"%-s  pH & DO", hausname[osjk_env.c_haus]);
//      if(draw_new)
        if(ctrldata_header[osjk_env.c_haus].init &&
                        ctrldata_header[osjk_env.c_haus].started &&
                        !ctrldata_header[osjk_env.c_haus].stoped)
        {
                msgbox(disp_pos[win][0],disp_pos[win][1],disp_pos[win][2],disp_pos[win][3],
                                                        WHITE,  BLACK, LIGHTGRAY,  DARKGRAY,
WHITE,CYAN, buf, 0);
                pushcolor();
                hsetbkcolor(DARKGRAY);
                hsetcolor(WHITE);

                hhline(disp_pos[win+3][0],disp_pos[win+3][1],disp_pos[win+3][2], WHITE);
                hvline(disp_pos[win+3][0],disp_pos[win+3][1]-120,120, WHITE);

                for(i = 0; i < 480; i += 20) {
                        hvline(-(disp_pos[win+3][0]+i),
                                                        -(disp_pos[win+3][1]), i % 40 ? 5:10,
WHITE);
                        if(!(i % 40))
                                hprintfpxy(-(disp_pos[win+3][0]+i-8),

-(disp_pos[win+3][1]+4) ,"%02d", i / 20);
                }

                for(i = 0; i <= 50; i += 10) {
                        hhline(-(disp_pos[win+3][0] - 8),
                                                        -(disp_pos[win+3][1] - i * 2), 10,
WHITE);
```

```
                        hprintfpxy(-(disp_pos[win+3][0]-64),
                                                                -(disp_pos[win+3][1]
- i * 2-8), "%4.1f ", i / 3.571);
                }

                load2ctrldata(osjk_env.c_haus);
                for(i=0; i < 48 && ctrldata_header[osjk_env.c_haus].init; i++)
                {
                        //pH ----------------------------------------------------------
                        hhline(-(disp_pos[win+3][0]+(float)i * 10 ),
                                                                -(disp_pos[win+3][1]
_ctrldata_.step[0][0] * .714), 10,
                                                                RED);
                        //DO ----------------------------------------------------------
                        hhline(-(disp_pos[win+3][0]+(float)i * 10.),
                                                                -(disp_pos[win+3][1]
_ctrldata_.step[0][3] * .714), 10,
                                                                GREEN);
                }
                pushcolor();
                hsetcolor(LIGHTRED);
                hprintfxy(-(disp_pos[win][0]+68), -(disp_pos[win][1]+2 ),"[붉은선] pH %4.1f",
                                        (float)sensordata[osjk_env.c_haus][5][disp_data_pos]    /
1000.);
                hsetcolor(LIGHTGREEN);
                hprintfxy(-(disp_pos[win][0]+68), -(disp_pos[win][1]+3 ),"[녹색선] DO %4.1fppm",
                                        (float)sensordata[osjk_env.c_haus][3][disp_data_pos]    /
1000.);
                popcolor();
                for(i=1; i <= disp_data_pos; i++)
                {
                        setcolor(LIGHTRED);
                        line( disp_pos[win+3][0] + i,
                                                                disp_pos[win+3][1]
(int)(sensordata[osjk_env.c_haus][5][i-1] /1000.*7.14),
                                                        disp_pos[win+3][0] + i+1,
                                                        disp_pos[win+3][1]
(int)(sensordata[osjk_env.c_haus][5][i] /1000.*7.14));
        //
        //
        //
        //
                        setcolor(LIGHTGREEN); //DO
                        line( disp_pos[win+3][0] + i,
                                                                disp_pos[win+3][1]
(int)(sensordata[osjk_env.c_haus][3][i-1] /1000.*7.14),
                                                        disp_pos[win+3][0] + i+1,
                                                        disp_pos[win+3][1]
(int)(sensordata[osjk_env.c_haus][3][i] /1000.*7.14));
        //
        //
        //
        //
                }
                popcolor();
        }       else
                agc_mre();
}
/*
void siren(void)
{
        int i = 0;
        for(i = 0; i < 500; i++)
        {
                sound(i + 500);
                delay(1);
        }
        nosound();
}
*/
/*

ESC C
2 byte ; status

7  cheon close
6        open
```

```
5  cheik close
4        open
3  curtain  close
2           open
1  boiler   on ==1 / off
0  fan       on /off
7  mist      on /off
6
5
4
3
2
1  alarm ==1
0


1 byte ; limit

7  cheon close
6        open
5  chik  cls
4        opn
3  manual/auto==0
2
1
0
*/

void watch_ctrl(int haus)
{
        char msg[60];
        int j;
        static int lasthaus = -1;

        if(haus != osjk_env.c_haus)
                return;

        if(lasthaus != osjk_env.c_haus)
                for(j=0; j < 90; j+=3)
                {
                        disp_button(&status[j], false);
                        lasthaus = osjk_env.c_haus;
                }
        if(haus < 4)
        {
                for(j = 0; j < 30; j++)
                {
                        switch(j)
                        {
                                case 0:
                                        disp_button(&status[j*3+flag.win_status.top1],
flag.win_status.top1);
                                        break;
                                case 1:
                                        disp_button(&status[j*3+flag.win_status.top2],
flag.win_status.top2/*==2? false:true*/);
                                        break;
                                case 2:
                                        disp_button(&status[j*3+flag.win_status.side1],
flag.win_status.side1);
                                        break;
                                case 3:
                                        disp_button(&status[j*3+flag.win_status.side2],
flag.win_status.side2);
                                        break;
                                case 4:
                                        disp_button(&status[j*3+flag.win_status.curtain]
flag.win_status.curtain);
                                        break;
                                case 5:
                                        disp_button(&status[j*3+flag.win_status.boiler],
```

```
flag.win_status.boiler);
                                        break;
                                case 6:
                                        disp_button(&status[j*3+flag.win_status.vent1],
flag.win_status.vent1);
                                        break;
                                case 7:
                                        disp_button(&status[j*3+flag.win_status.vent2],
flag.win_status vent2);
                                        break;
                                case 8:
                                        disp_button(&status[j*3+flag.win_status.driftf],
flag.win_status.driftf);
                                        break;
                                case 9:
                                        disp_button(&status[j*3+flag.win_status.himist]
flag.win_status himist);
                                        break;
                                case 10:
                                        disp_button(&status[j*3+flag.win_status.shelt1],
flag.win_status.shelt1);
                                        break;
                                case 11:
                                        disp_button(&status[j*3+flag.win_status.shelt2]
flag.win_status.shelt2);
                                        break;
                                case 12:
                                        disp_button(&status[j*3+flag.win_status.mist1],
flag.win_status.mist1);
                                        break;
                                case 13:
                                        disp_button(&status[j*3+flag.win_status.mist2],
flag win_status.mist2);
                                        break;
                                case 14:
                                        disp_button(&status[j*3+flag.win_status.mist3],
flag.win_status.mist3);
                                        break;
                                case 15:
                                        disp_button(&status[j*3+flag.win_status.irig1],
flag.win_status.irig1);
                                        break;
                                case 16:
                                        disp_button(&status[j*3+flag.win_status.irig2],
flag.win_status.irig2);
                                        break;
                                case 17:
                                        disp_button(&status[j*3+flag.win_status.irig3],
flag.win_status.irig3);
                                        break;
                                case 18:
                                        disp_button(&status[j*3+flag.win_status.co2],
flag.win_status.co2);
                                        break;
                                case 19:
                                        disp_button(&status[j*3+flag.win_status.alarm]
flag.win_status.alarm);
                                        break;
                                case 29:
                                        disp_button(&status[j*3+flag.win_status.froze],
flag.win_status.froze);
                                        break;
```

```
                        }
                }
        } else //hyph
        {
                for(j = 19; j < 29; j++)
                {
                        switch(j)
                        {
                                case 19:
                                        disp_button(&status[j*3+flag.win_status.alarm]
flag.win_status.alarm);
                                        break;
                                case 20:
                                        disp_button(&status[j*3+flag.water.ab],
flag.water.ab);
                                        break;
                                case 21:
                                        disp_button(&status[j*3+flag.water.acid],
flag water.acid);
                                        break;
                                case 22:
                                        disp_button(&status[j*3+flag.water.base],
flag.water.base);
                                        break;
                                case 23:
                                        disp_button(&status[j*3+flag.water.spray],
flag.water.spray);
                                        break;
                                case 24:
                                        disp_button(&status[j*3+flag.water.nft],
flag.water.nft);
                                        break;
                                case 25:
                                        disp_button(&status[j*3+flag.water.rockwool],
flag water.rockwool);
                                        break;
                                case 26:
                                        disp_button(&status[j*3+flag.water.wonsoo],
flag water wonsoo),
                                        break;
                                case 27:
                                        disp_button(&status[j*3+flag.water.heater],
flag water.heater),
                                        break;
                                case 28:
                                        disp_button(&status[j*3+flag.water.stirrer],
flag water.stirrer);
                                        break;
                        }
                }
        }
        if(osjk_env.c_haus < 4)
                if(osjk_env.haus[osjk_env.c_haus].isauto)
                        isautomatic = true;
                else
                        isautomatic = false;

        if(flag win_status.alarm)
        {
                if(haus < 4)
                {
                        sprintf(msg,"%d번 온실 이상 검출", haus+1);
                        status_help(msg);
                        sprintf(msg,"\"%d번 온실 경보!\" 점검하세요", haus+1);
                } else
                {
                        sprintf(msg,"%d번 온실제어기 이상 검출", haus-3);
```

```
                        status_help(msg);
                        sprintf(msg,"\"%d번 온실제어기 경보!\" 점검하세요", haus-3);
                }
                puterror(msg);
//              siren();
                flag.win_status.alarm = false;
        }
}


void log_ctrl_data(int haus)
{
        FILE *fout;
        struct date d;
        struct time t;
        char *door[] = {"정지","열림","닫힘"};
        char *onoff[] = {"꺼짐","켜짐"};
        char fn[80];
        getdate(&d);
        gettime(&t);
        mou_wait(true);
        status_help("제어자료를 디스크에 보관합니다");

        sprintf(fn,"%s %1d%02d", osjk_env haus[haus].ctrl_log_fn,

                d.da_year % 10,   d.da_mon);

        if(haus < 4)
        {

                if(access(fn, 0))
                {
                        // make new file
                        fout = fopen(fn, "wb");
                        fprintf(fout, "온실지기 제어자료 보관 화일 기상제어기 %d\r\n", haus+1);
                        fprintf(fout,
"=============================================================================
=============================================================================\r\n")
;
                        fprintf(fout, "날    짜 시    간 경과일 창 온풍기 환기 관수 CO2 발 제어 경
보\r\n");
                        fprintf(fout,
"-----------------------------------------------------------------------------
-----------------------------------------------------------------------------\r\n")
;
                } else
                {
                        fout = fopen(fn, "ab");
                }
                //                                                  천 천 측 측 커 보 환 환 유
하 차 차 안 안 안 관 관 관 탄
                        fprintf(fout," %2d/%2d/%2d  %02d:%02d:%02d %6d %6s %6s %6s %6s %6s
%6s %6s %6s %6s %6s %6s %6s %6s %6s %6s %6s %6s %6s %6s %6s %6s %4s\r\n",
                                        d.da_year % 100, d.da_mon, d.da_day,
                                        t.ti_hour, t.ti_min, t.ti_min, ctrldata_header[haus].elapsed,
                                        door[flag.win_status.top1],
                                        door[flag.win_status.top2],
                                        door[flag.win_status.side1],
                                        door[flag.win_status.side2],
                                        door[flag.win_status.curtain],
                                        onoff[flag.win_status.boiler],
                                        onoff[flag.win_status.vent1],
                                        onoff[flag.win_status.vent2],
                                        onoff[flag.win_status.driftf],
                                        onoff[flag win_status.himist],
                                        door[flag.win_status.shelt1],
                                        door[flag.win_status.shelt2],
                                        onoff[flag.win_status.mist1],
                                        onoff[flag.win_status.mist2],
                                        onoff[flag.win_status.mist3],
                                        onoff[flag.win_status.irig1],
                                        onoff[flag.win_status.irig2],
                                        onoff[flag.win_status.irig3],
                                        onoff[flag.win_status.froze],
                                        (flag.win_status.auto_man) ? "자동":"수동",
                                        flag.win_status.alarm ? "발생":" -- "
                );
```

```
                fclose(fout);
        } else //양액
        {
                if(access(fn, 0))
                {
                        // make new file
                        fout = fopen(fn, "wb");
                        fprintf(fout, "온실지기 제어자료 보관 화일 온실제어기 %d\r\n", haus-3);
                        fprintf(fout,
"================================================================================\r\n");
                        fprintf(fout,
"--------------------------------------------------------------------------------\r\n");
                } else
                {
                        fout = fopen(fn, "ab");
                }
                        fprintf(fout," %2d/%2d/%2d  %02d:%02d:%02d %6d %-4s %-4s %-4s %-4s
%-4s %-4s %-6s %-6s %-6s\r\n",
                                                        d.da_year % 100, d.da_mon, d.da_day,
                                                        t.ti_hour,      t.ti_min,       t.ti_min,
ctrldata_header[haus].elapsed,
                                                        flag.water.ab ? "공급":"정지",
                                                        flag.water.acid ? "공급":"정지",
                                                        flag.water.base ? "공급":"정지",
                                                        flag.water.heater ? "가열":"정지",
                                                        flag.water.stirrer ? "교반":"정지",
                                                        flag.water.wonsoo ? "공급":"정지",
                                                        flag.water.spray ? "공급":"정지",
                                                        flag.water.nft ? "공급":"정지",
                                                        flag.water.rockwool ? "공급":"정지" );
                fclose(fout);
        }
        status_help("");
        mou_wait(false);

} // 제어 로그


void log_sensor_data(int haus, int pos)  // 센서 자료 로그
{
        FILE *fout;
        char fn[80];
        struct date d;
        struct time t;
        static int m;
        getdate(&d);
        gettime(&t);

        if(m == t.ti_min)
                return;
        if(t.ti_min % 30 == 0)
        {
                m = t.ti_min;
        } else
                return;

        mou_wait(true);
        sprintf(fn,"%s.%1d%02d", osjk_env.haus[haus].data_log_fn,

                d.da_year % 10,   d.da_mon);
        status_help("센서 자료를 디스크에 보관합니다");
        if(haus < 4) //보통
        {
                if(access(fn, 0))
                {
                        // make new file
                        fout = fopen(fn, "wb");
                        fprintf(fout,"온실지기 센서 자료 보관 화일 기상제어기 %d\r\n", haus+1);

fprintf(fout,"================================================================================
==========================\r\n");
                        fprintf(fout," 날    짜  시    각  경과일 온도1 온도2 습도1 습도2   CO2
일사  지면 풍향 풍속 외온 외습 강수 기  압\r\n");

fprintf(fout,"--------------------------------------------------------------------------------
--------------------------\r\n");
                } else
```

```
                {
                        fout = fopen(fn, "ab");
                }
                        fprintf(fout," %2d/%2d/%2d   %02d:%02d:%02d   %6d %5.1f %5.1f %5.1f
%5.1f %5.0f %5d %5.1f %4s %4.0f %4.1f %4d %4.1f %6.2f\r\n",
                                d.da_year % 100, d.da_mon, d.da_day,
                                        t.ti_hour,          t.ti_min,          t.ti_min,
(int)ctrldata_header[haus].elapsed,
                                sensordata[haus][7][pos] / 100., //온도
                                sensordata[haus][6][pos] / 100., //온도
                                sensordata[haus][5][pos] / 100., //습도
                                sensordata[haus][4][pos] / 100., //습도
                                sensordata[haus][3][pos] / 10.,  //씨오투
                                sensordata[haus][2][pos] ,       //일사
                                sensordata[haus][0][pos] / 100., //지면온도
                                wind_direction[dicision_wd(wsensor.sdata.wd)],
                                wsensor.sdata.ws*1609.3 / 3600.,
                                (wsensor.sdata.out_t /10.- 32.)* 5.0 / 9.0,
                                wsensor.sdata.out_h,
                                wsensor.sdata.rain * 0.254,
                                wsensor.sdata.atm * .03386
                );
                fclose(fout);
        } else
        {
                if(access(fn, 0))
                {
                        // make new file
                fout = fopen(fn, "wb");

fprintf(fout,"=================================================================================
\r\n");

fprintf(fout,"---------------------------------------------------------------------------------
\r\n");
                } else
                {
                        fout = fopen(fn, "ab");
                }
                fprintf(fout," %2d/%2d/%2d   %02d:%02d:%02d   %6d   %4.1f      %3.1f      %5.1f
%4.1f    %3.1f    %5.1f\r\n",
                                d.da_year % 100, d.da_mon, d.da_day,
                                t.ti_hour, t.ti_min, t.ti_min, ctrldata_header[haus].elapsed,
                                sensordata[haus][5][pos] / 1000.,
                                sensordata[haus][4][pos] / 1000.,
                                sensordata[haus][7][pos] / 100.,
                                sensordata[haus][3][pos] / 1000.,
                                sensordata[haus][2][pos] / 1000.,
                                sensordata[haus][6][pos] / 100.
                );
                fclose(fout);
        }
        status_help("");
        mou_wait(false);
}


void print_file(char *filename)
{
        FILE *fp;
        char buf[1024];
        fp = fopen(filename,"rb");
        if(fp == NULL)
                        return;
        mou_wait(true);
                while(fgets(buf, 1024, fp))
                {
                        if(pputs(buf))
                                break;
                }
        mou_wait(false);
                fclose(fp);
}

int view_logfile(void)
{
        char filename[80]="*.*";
```

```
        dirbox("보고자 하는 로그화일이름을 고르세요",
                                        filename, NULL, 0);
        if(*filename)
                view_file("온실지기 로그화일 보기", filename, 1);
        return false;
}

int print_logfile(void)
{
        char filename[80]="*.*";
        dirbox("인쇄하고자 하는 화일을 고르세요", filename, NULL, 0);
        mou_wait(true);
        if(*filename)
                print_file(filename);
        mou_wait(false);
        return false;
}

void print_to_printer(void)
{
        static min = -1;
        struct time t;
        struct date d;
        int i;
        gettime(&t);
        getdate(&d);
        if(min != t.ti_min && !(t.ti_min % 30))
        {
                if(printsw == 0)
                {
                        //puterror("센서자료를 인쇄하지 안도록 설정되어 있습니다");
                        return;
                }
                min = t.ti_min;
                //
                //  print data
                //
                if(pprintf("◆ %4d년 %2d월 %2d일 %02d:%02d [%4s] 현재 온실상황보고    [온실지기
V1.20 농업과컴퓨터]\r\n\n",
                                                        d.da_year,    d.da_mon,    d.da_day,
t.ti_hour, t.ti_min,
                                                        flag.win_status.rain ? "비옴":"----"))
                        return;
                if(pprintf("☼ 풍향: [%4s] 풍속: %3.1f m/sec 외부온도: %+3.0f °C 외부습도: %3d
%%\r\n☼ 강수량: %3.1f mm 기압: %6.1f mb 일사량: %5d W/m₂ \r\n",
                                        wind_direction[dicision_wd(wsensor.sdata.wd)],
                                        wsensor.sdata.ws*1609.3 / 3600.,
                                        (wsensor.sdata.out_t /10.- 32.)* 5.0 / 9.0,
                                        wsensor.sdata.out_h,
                                        wsensor.sdata.rain * 0.254,
                                        wsensor.sdata.atm * .03386,
                                        wsensor.sdata.light ))
                        return;
                for(i = 0; i < 4; i++)
                        if(ctrldata_header[i].elapsed)
                                if(pprintf("[온실%d] 경과작기: %3d일 온도1: %+5.1f °C 온도2:
%+5.1f °C 습도1: %3d %% 습도2: %3d %% 이산화탄소: %4d ppm.\r\n",
                                                                i+1,
                                                                ctrldata_header[i].elapsed,
                                                                (float)(sensordata[i][7][disp_data_pos] /
100.),
                                                                (float)(sensordata[i][6][disp_data_pos] /
100.),
                                                                (int)(sensordata[i][5][disp_data_pos] /
100.),
                                                                (int)(sensordata[i][4][disp_data_pos] /
100.),
                                                                (int)(sensordata[i][3][disp_data_pos] /
10.))
                                        )
                                        return;
                for(i = 4; i < 7; i++)
                        if(ctrldata_header[i].elapsed)
                                pprintf("[양액%d] 경과작기: %3d일 pH: %5.2f EC: %5.2fuS DO:
%5.2fppm 수온: %4.1f °C.\r\n",
                                                                i-3,
                                                                ctrldata_header[i].elapsed,
```

```
                                                        (float)sensordata[i][5][disp_data_pos] /
1000.,
                                                        (float)sensordata[i][4][disp_data_pos] /
1000.,
                                                        (float)sensordata[i][3][disp_data_pos] /
1000.,
                                                        (float)(sensordata[i][7][disp_data_pos] /
100.)),
    pprintf("\r\n");
        }
}


void agc_mre(void)
{
                pushcolor();
                hsetbkcolor(150);
//              hsetbkcolor(CYAN);

                hclrscrxy(-19,-2, 107, 40);

                setcolor(WHITE);
                settextstyle(TRIPLEX_FONT, HORIZ_DIR, 6);
                outtextxy(450, 240, "Ag&C" );
                settextstyle(TRIPLEX_FONT, HORIZ_DIR, 6);
//              outtextxy(300, 300, "Mirae Electronics" );
                outtextxy(350, 350, "1994,1995" );

                hsetoutputmode(OVERWRITE_ENLARGE);
                hsetenlarge(1,5);
                hprintfpxy(-320, -120,"서울대농생명과학대학 온실 제어시스템");
                hsetthreed(false);
                hsetoutputmode(OVERWRITE);
                popcolor();
}


int edit_orchid(void)
{
 ctrldata_t tmpdata, tmpdata2;
 FILE *fin;
 mywindow_t info_w;
 button_t btn[20];
 int hkey[20] =
        { LEFTARROW, RIGHTARROW, UPARROW, DOWNARROW,
                '-', '+', PGUPKEY, PGDNKEY, HOMEKEY, ENDKEY,
                F5, F6, F10, ESC, F2, CR, ALT_1, ALT_2, ALT_3, ALT_4};
 int ii, items = 0, key, done = false, redraw = true, step = 0, day = 0;
 int last_item = -1;
 int pos[20][4] =
                {
                        {27,  24, 31, 25},
                        {31,  24, 35, 25},
                        {35, 24, 39, 25},
                        {39, 24, 43, 25},

                        {43, 24, 53, 25},
                        {53, 24, 63, 25},
                        {63, 24, 73, 25},
                        {73, 24, 83, 25},
                        {83, 24, 93, 25},
                        {93, 24, 103, 25},

                        {27, 25, 40, 27},
                        {40, 25, 53, 27},

                        {53, 25, 63, 27},

                        {63, 25, 73, 27},
                        {73, 25, 83, 27},
                        {83, 25, 103, 27},
                        // select item
                        {92, 5,  103, 7 },
                        {92, 7,  103, 9 },
                        {92, 9,  103, 11 },
                        {92,11,  103, 13 }
```

```
        };
int scale[4] = {4, 2, 2, 4}, scrollbar, cyr, cmon, cday;
float  max, min, mean;
struct date d;
int sh, sm, eh, em;
char *text[20] =
        {"Ⅱ","Ⅲ","▲","▼",
        "어제 -","내일 +", "앞쪽 PgUp","뒤쪽 PgDn","처음 Home","끝 End",
        "일괄지정 F5","개별지정 F6","-","취소 ESC","완료 F2", "저장 ⊟",
        "온도 @1", "-", "-", "-"};
char buf[80];
char *report[4][4] =
{
                {"^c1^CC▼^CF    최고온도^c7","^c1^C0▼^CF    최저온도^c7","^c1^C2▼^CF    평균온도
^c7","^c1^CB▼^CF 설정온도^c7"},
                {"^c1^CC▼^CF    최고습도^c7","^c1^C0▼^CF    최저습도^c7","^c1^C2▼^CF    평균습도
^c7","^c1^CB▼^CF 설정습도^c7"},
                {"^c1^CC▼^CF   최고   CO2^c7","^c1^C0▼^CF   최저   CO2^c7","^c1^C2▼^CF   평균
CO2^c7","^c1^CB▼^CF 설정 CO2^c7"},
                {"^c1^CC▼^CF    최고광도^c7","^c1^C0▼^CF    최저광도^c7","^c1^C2▼^CF    평균광도
^c7","^c1^CB▼^CF 설정광도^c7"}
};

        if(!ctrldata_header[0].init) {
                puterror("작기가 설정되어 있지 않습니다. 먼저 작기를 설정하세요");
                return false;
        }

        for(ii = 0 ; ii < 20; ii++) {
                set_button(&btn[ii], pos[ii][0],pos[ii][1],pos[ii][2],pos[ii][3],
                WHITE, BLACK, ii < 16 ? LIGHTGRAY:GREEN,ii < 16 ? LIGHTGREEN:WHITE,
BLACK,
                false, false, text[ii]);
        }

        sprintf(buf,"온실 제어정보 편집기");
        setmywindow(&info_w, 25, 3, 104, 27, WHITE, BLACK, GREEN, BLUE,
                                        WHITE, BLACK, LIGHTGRAY, false, true,
                                        buf, false);
        memset(&tmpdata, 0, sizeof(ctrldata_t));
        memset(&tmpdata2, 0, sizeof(ctrldata_t));
        savescreen();
        if(open_window(&info_w)) {
                pushcolor();
                hsetbkcolor(LIGHTGRAY);
                hsetcolor(BLACK);
                allow_darg(true);

                hsolidbarpxy(-208,-63, -511, -280, LIGHTGRAY);

                rectangle_3d(208, 63, 511, 280, WHITE, BLACK);
                // file open
                mou_wait(true);
                copyfile("ORCHID.CTL", "ORCHID.$$$");
                mou_wait(false),

                fin = fopen("ORCHID.CTL", "r+b");
                if(fin == NULL)
                {
                        fin = fopen("ORCHID.CTL", "w+b");
                        chsize(fileno(fin),
                                                sizeof(ctrldata_header_t)
                                                +           sizeof(ctrldata_t)          *
ctrldata_header[0].duration);
                }
                if(fin != NULL)
                {
                        for(ii = 0; ii < 20; ii++)
                                open_button(&btn[ii]);
                        ctrlputspxy(8,282,"^C3^cF^RⅡ^r[redacted]^RⅢ^r^C0^c7");
                        if(ctrldata_header[0].elapsed)
                                day = ctrldata_header[0].elapsed-1;
                        else
                                day = 0;
                        while(!done)
                        {
```

```
                        if(redraw) {
                                fseek(fin, (long)sizeof(ctrldata_header_t)
                                                + (long)sizeof(ctrldata_t) *
(long)day, SEEK_SET);
                                fread(&tmpdata2, sizeof(ctrldata_t), 1, fin);
                                if(tmpdata2.init[items])
                                        tmpdata = tmpdata2;

                                if(last_item != items|| key == F10)
                                {
                                        hsolidbarpxy(-209,-80,      -509,      -263,
LIGHTGRAY);
                                disp_button(&btn[16+items], true);

                                hhline(80,260,420, BLACK);
                                hvline(80,20,240, BLACK);
                                hprintfpxy(30, 262,"시각 →");
                                for(ii = 0; ii < 48; ii++) {
                                        hvline(ii*8+115, ii % 2 ? 255:250,ii % 2
? 5:10, BLACK);
                                        if(!(ii % 4))
                                                hprintfpxy(ii*8+108,
262,"%02d", ii / 2);
                                }
                                switch (items)
                                {
                                        case 0: // 온도
                                                for(ii = 0; ii <= 50; ii+=10) {
                                                        hhline(80, ii * 4 +
24, 10, BLACK);
                                                        hprintfpxy(30, ii * 4
+ 16, "%2d ° C", 50-ii);
                                                }
                                                ctrlprintfpxy(530, 282,"^c2^CF
[± %3d ° C]^C0^c7",
ctrldata_header[0].error[items]
* ctrldata_header[0].mul[items]);
                                                break;
                                }
                                        ctrlprintfxy(67, 10, report[items][0]);
                        }
                        redraw = false;
                }

                // draw cursor
                for(ii = 0; ii < 48; ii++)
                        hprintfpxy(8*(14+ii),
                        216-tmpdata.ctrlitem[items].step[ii] * scale[items],"-");

                for(ii=0, max = 0; ii < 48; ii++) {
                        if(tmpdata.ctrlitem[items].step[ii] > max)
                                max = tmpdata.ctrlitem[items].step[ii];
                }
                for(ii=0, min = tmpdata.ctrlitem[items].step[0]; ii < 48; ii++) {
                        if(tmpdata.ctrlitem[items].step[ii] < min)
                                min = tmpdata.ctrlitem[items].step[ii];
                }
                for(ii=0, mean = 0; ii < 48; ii++) {
                        mean += tmpdata.ctrlitem[items].step[ii] / 48.;
                }
                switch(items)
                {
                        case 0:
                                ctrlprintfxy(68, 11, "^c1^CF%4.1f ° C  ^C0^c7",
                                                                max             *
ctrldata_header[0].mul[items]);
                                ctrlprintfxy(68, 13, "^c1^CF%4.1f ° C  ^C0^c7",
                                                                        min*
ctrldata_header[0].mul[items]);
                                ctrlprintfxy(68, 15, "^c1^CF%4.1f ° C  ^C0^c7",
                                                                      mean*
ctrldata_header[0].mul[items]);
                                ctrlprintfxy(68, 17, "^c1^CF%4.1f ° C  ^C0^c7",

        (float)tmpdata.ctrlitem[items].step[step]
```

```
*ctrldata_header[0].mul[items]);
                                                    break;
                                    }
                                    ctrlprintfpxy(8*(14+step),216

tmpdata.ctrlitem[items].step[step] * scale[items] ,

tmpdata2.init[items] ? "^CA^A-^a^C0":"^CC^A-^a^C0");
                                    scrollbar = (int)(464. / (ctrldata_header[0].duration-1) * (day))+24;
                                    ctrlputspxy(scrollbar,282, "^CF^c0^A■^a^C0^c7");

                                    calc_edays(ctrldata_header[0].syear,

ctrldata_header[0].smon,

ctrldata_header[0].sday,
                                                                                    day,    &cyr,

&cmon, &cday),
                                    d.da_year = cyr;
                                    d.da_mon = cmon;
                                    d.da_day = cday;
                                    getsunrise(d, &sh, &sm, &eh, &em);
                                    ctrlprintfpxy(sh*16+115+sm*16/60-48,232,"^CE      ☼      ^C0");
                                    ctrlprintfpxy(eh*16+115+em*16/60-48,232,"^C0      ☼      ^C0");

                                    hsetcolor(BLACK);
                                    ctrlprintfxy(3, 1, "^U%2d시 %02d분  < %s지정 >  [ %3d일 / %3d
일 ]  [%4d년 %2d월 %2d일]^u",
                                                    step / 2, step % 2 ? 30:0,
                                                    ctrldata_header[0].individual ? "개별":"일괄",
                                                    day + 1,ctrldata_header[0] duration,
                                                    d.da_year, d.da_mon, d.da_day);

                                    // get key
                                    key = hgetch();

                                    for(ii = 0; ii < 48; ii++)
                                            ctrlprintfpxy(8*(14+ii),216
                                                                                    -
tmpdata.ctrlitem[items].step[ii] * scale[items] ," ");

                                    switch (key)
                                    {
                                                    case '+': case PGDNKEY: case '-': case
PGUPKEY: case HOMEKEY: case ENDKEY:
                                                    case MOUKEY:
                                                            ctrlputspxy(scrollbar,282, "^C3■^C0");
                                                            break;
                                                    case CR:
                                                            tmpdata.init[items] = true;
                                                            tmpdata2.init[items] = true;
                                                            fseek(fin, (long)sizeof(ctrldata_header_t)
                                                                                                    +
(long)sizeof(ctrldata_t) * (long)day, SEEK_SET);
                                                            fwrite(&tmpdata,    sizeof(ctrldata_t),    1,

fin);
                                                            osjk_env.haus[0].edited = true;
                                                            break;
                                    }
                                    switch(key)
                                    {
                                            case ESC:
                                                    if(osjk_env.haus[0].edited)
                                                            if(askanswer("현재 입력한 것을 취소하
십니까", false) != true)
                                                                    break;
                                                    fclose(fin);
                                                    copyfile("ORCHID.$$$","ORCHID.CTL");
                                                    remove("ORCHID.$$$");
                                                    done = true;
                                                    break;
                                            case '-':
                                                    if(day > 0)
                                                            day --;
                                                    redraw = true;
                                                    break;
```

```
case '+':
        if(day < ctrldata_header[0].duration - 1)
                day++;
        redraw = true;
        break;
case UPARROW:
        if(tmpdata.ctrlitem[items].step[step] < 200. /
scale[items] )
tmpdata ctrlitem[items].step[step]++;
        osjk_env.haus[0].edited = true;
        break;
case DOWNARROW.
        if(tmpdata.ctrlitem[items] step[step] > 0)
tmpdata.ctrlitem[items].step[step]--;
        osjk_env.haus[0].edited = true;
        break;
case LEFTARROW: case '4':
        if(step > 0)
                step--;
        break;
case RIGHTARROW: case '6':
        if(step < 48-1)
                step++;
        break;
case PGUPKEY:
        if(day - 7 > 0)
                day -= 7;
        else
                day = 0;
        redraw = true;
        break;
case PGDNKEY:
        if(day + 7 < ctrldata_header[0].duration)
                day += 7;
        else
                day = ctrldata_header[0].duration - 1;
        redraw = true;
        break;
case HOMEKEY:
        day = 0;
        redraw = true;
        break;
case ENDKEY:
        day = ctrldata_header[0].duration - 1;
        redraw = true;
        break;
case '8':
        for(ii=0; ii < 48; ii++)
                if(tmpdata.ctrlitem[items].step[ii] < 200.
/ scale[items] )
tmpdata.ctrlitem[items].step[ii]++;
        osjk_env.haus[0].edited = true;
        break;
case '2':
        for(ii=0; ii < 48; ii++)
                if(tmpdata.ctrlitem[items].step[ii] > 0)
tmpdata.ctrlitem[items].step[ii]--;
        osjk_env.haus[0].edited = true;
        break;
case '7':
        for(ii=0; ii <= step; ii++)
                if(tmpdata.ctrlitem[items].step[ii] < 200.
/ scale[items] )
tmpdata.ctrlitem[items].step[ii]++;
        osjk_env.haus[0].edited = true;
        break;
case '1':
        for(ii=0; ii <= step; ii++)
                if(tmpdata.ctrlitem[items].step[ii] > 0)
tmpdata.ctrlitem[items].step[ii]--;
        osjk_env.haus[0].edited = true;
```

```
                                        break;
                                case '9':
                                        for(ii=step; ii < 48; ii++)
                                                if(tmpdata.ctrlitem[items].step[ii] < 200.
/ scale[items] )
tmpdata.ctrlitem[items].step[ii]++;
                                        osjk_env.haus[0].edited = true;
                                        break;
                                case '3':
                                        for(ii=step; ii < 48; ii++)
                                                if(tmpdata.ctrlitem[items].step[ii] > 0)
tmpdata.ctrlitem[items].step[ii]--;
                                        osjk_env.haus[0].edited = true;
                                        break;
                                case MOUKEY:
                                        ii = process_button(btn, 20);
                                        if(ii >= 0)
                                                ungetxch(hkey[ii]);
                                        else
                                        if((mou_x > 14 && mou_x < 63) && (mou_py
< 224 && mou_py > 22)) {
                                                step = mou_x - 15;
                                                ctrlprintfpxy(8*(14+step),216 -
tmpdata.ctrlitem[items].step[step] * scale[items] ," ");
tmpdata.ctrlitem[items].step[step] = (224 - mou_py) / scale[items];
                                                osjk_env.haus[0].edited = true;
                                        } else
                                        if(mou_py > 282 && mou_py < 298)
                                        {
                                                if(mou_px > 8 && mou_px < 24)
                                                        ungetxch('-');
                                                else if(mou_px > 504 && mou_px <
520)
                                                        ungetxch('+');
                                                else if(mou_px > 24 && mou_px <
504) {
                                                        day =(int)((float)(mou_px -
24) * (float)(ctrldata_header[0].duration) / 480.);
                                                        redraw = true;
                                                }
                                        }
                                        break;
//                              case F10:
//                                      sprintf(buf,"%3d",ctrldata_header[0].error[items]);
//                                      if(win_getdata("오차의 범위를 입력하세요", buf,
"999", ENGONLY))
                                                ctrldata_header[0].error[items] =
//                                              atoi(buf);
                                        redraw = true;
//                                      break;
                                case F2:
                                        // save
                                        if(!ctrldata_header[0].individual) {
                                                putmsg("[일괄지정] 나머지 작기를 모두
동일하게 설정합니다", true);
                                                tmpdata.init[0] = tmpdata.init[1] = true;
                                                fseek(fin, 0, SEEK_SET);
                                                        fwrite(&ctrldata_header[0],
sizeof(ctrldata_header_t), 1, fin);
                                                for(ii = day; ii <
ctrldata_header[0].duration; ii++)
                                                {
fseek(fin,(long)sizeof(ctrldata_header_t) + (long)sizeof(ctrldata_t) * (long)ii, SEEK_SET);
                                                        fwrite(&tmpdata
sizeof(ctrldata_t), 1, fin);
                                                }
                                                closemsg();
                                        }
                                        fclose(fin);
                                        remove("ORCHID.$$$");
                                        //*****************************
```

```
                                        dnload_orchid();
                                        //*****************************

                                        done = true;
                                        break;
                                case F5:
                                        ctrldata_header[0].individual = false
                                        break;
                                case F6:
                                        ctrldata_header[0].individual = true;
                                        break;
                                }
                        }
                } else
                        puterror("작기화일을 열 수가 없습니다");

                allow_darg(false);

                for(ii = 0; ii < 20; ii++)
                        close_button(&btn[ii]);
                popcolor();
                close_window(&info_w);
        }
        restorescreen();
 return false;
}
```

```
//
//
//        MULTIPORT  Device  Driver Interface
//
//

#include  <dos.h>
#include  "mux.h"

int u_peek(int seg_,int off_)
{
          char      far *farread;
          farread = (char far *) ((long)seg_* 0x10000l + (long)off_);
          return( (*farread & 0xff) + (*(farread+1) << 8) );
}

int setcom(int port_no,int byte_spec,int bps) /*  Set Port Address        */
{                                             /*  MULTIPORT INITIALIZE    */
        union  REGS  r;
        int    seg,val;

        seg      =   u_peek( 0,   0x142 );  /* 0x50 vector address is 0:0x140 */
        val      =   u_peek( seg, 0x103 );

        if( val != 0x434D ) return( 0x80 );    /* device not found          */

        r.h.ah   =   0x00;                      /*  number of Function      */
        r.h.bl   =   port_no;                   /*  Port Number             */
        r.h.bh   =   byte_spec;                 /*  Port initialize parm    */
        r.x.cx   =   bps;

        int86(0x50, &r, &r);
        return (r.h.al);
}

int clearcom(int port_no, int svr_code)  /*  clear buffer & XON/XOFF */
{                                        /*  MULTIPORT INITIALIZE    */
        union  REGS  r;
        int    seg,val;

        seg      =   u_peek( 0,   0x142 );   /* 0x50 vector address is 0:0x140 */
        val      =   u_peek( seg, 0x103 );

        if( val != 0x434D ) return( 0x80 );    /* device not found          */

        r.h.ah   =   0x01;                      /*  number of Function      */
        r.h.bl   =   port_no;                   /*  Port Number             */
        r.h.bh   =   svr_code;
        int86(0x50, &r, &r);
        return (r.h.al);
}

int writecom (int port_no, char *data, int data_size) /* Write data */
{

        char      far   *fptr;              /* send character in user buffer */
        union     REGS      r;
        struct    SREGS  sregs;
        int    seg,val;

        seg      =   u_peek( 0,   0x142 );   /* 0x50 vector address is 0:0x140 */
        val      =   u_peek( seg, 0x103 );

        if( val != 0x434D ) return( 0x80 );      /* device not found          */

        r.h.ah   =  0x02;
        r.h.bl   =  port_no;
        r.x.cx   =  data_size;
        fptr     =  (char far *)data;
        sregs.ds =  FP_SEG (fptr);               /* Segment of Read Buffer   */
        r.x.dx   =  FP_OFF (fptr);               /* Offset  of Read Buffer   */

        int86x (0x50, &r, &r,  &sregs);
        return ( r.h.al );
}

int delayedout(int port, char *data, int size)
```

```
{
        int i, ret,
        for(i=0; i<size; i++)
        {
                delay(COMDELAY);
                ret = writecom(port, data, 1);
                if(ret)
                        break;
                data++;
        }
        return ret;
}

int readcom(int port_no,  char *data, int *data_size)
{
                                                                /*  Read From Comm Buffer
 */
        char      far   *fptr;
        union     REGS      r;
        struct    SREGS  sregs;
        int     seg,val;

        seg       =    u_peek( 0,    0x142 );  /* 0x50 vector address is 0:0x140 */
        val       =    u_peek( seg, 0x103 );

        if( val != 0x434D ) return( 0x80 );     /* device not found          */

        r.h.ah    =   0x03;
        r.h.bl    =   port_no;
        r.x.cx    =   *data_size;
        fptr      =   (char far *)data;
        sregs.ds  =   FP_SEG (fptr);            /* Segment of Read Buffer    */
        r.x.dx    =   FP_OFF (fptr);            /* Offset  of Read Buffer    */

        int86x (0x50, &r, &r,  &sregs);
        *data_size = r.x.cx;
        return ( r.h.al );
}

int readdcom (int port_no, char *w_data, int *data_size, char *s_data, int search_size)
{
                                                                /*  Read From Comm Buffer
 */
        union     REGS      r;
        struct    SREGS  sregs;
        char      far    *f_ptr;
        int       seg,val;

        seg       =    u_peek( 0,    0x142 );  /* 0x50 vector address is 0:0x140 */
        val       =    u_peek( seg, 0x103 );

        if( val != 0x434D ) return( 0x80 );     /* device not found          */

        r.h.ah    =   0x04;
        r.h.bl    =   port_no;
        r.h.bh    =   search_size;              /* Delimiter                 */
        r.x.cx    =   *data_size;

        f_ptr     =   (char far *)w_data;
        sregs.ds  =   FP_SEG (f_ptr);           /* Segment of Read Buffer    */
        r.x.dx    =   FP_OFF (f_ptr);           /* Offset  of Read Buffer    */

        f_ptr     =   (char far *)s_data;
        sregs.es  =   FP_SEG (f_ptr);           /* Segment of Read Buffer    */
        r.x.di    =   FP_OFF (f_ptr);           /* Offset  of Read Buffer    */

        int86x (0x50, &r, &r,  &sregs);
        *data_size = r.x.cx;
        return ( r.h.al );
}

int checkcom(int port_no, int infor_code, int *data) /*  get port status */
{
        union     REGS  r;
        int       seg,val;

        seg       =    u_peek( 0,    0x142 );  /* 0x50 vector address is 0:0x140 */
```

```
        val     =   u_peek( seg, 0x103 );

        if( val != 0x434D ) return( 0x80 );    /* device not found          */

        r.h.ah  =   0x07;                      /*  number of Function       */
        r.h.bl  =   port_no;                   /*  Port Number              */
        r.h.bh  =   infor_code;
        int86(0x50, &r, &r);
        *data   = r.x.cx;
        return (r.h.al);
}

int oncom(int port_no, int mode, char del_addr[], int del_len, int far *u_addr)
{
                union     REGS   r;
                struct    SREGS  sregs;
                int       seg,val;

                seg     =   u_peek( 0,   0x142 );  /* 0x50 vector address is 0:0x140 */
                val     =   u_peek( seg, 0x103 );

                if( val != 0x434D ) return( 0x80 ); /* device not found          */

                r.h.ah  =  0x08;
                r.h.bl  =  port_no;
                r.h.bh  =  mode;                    /* event mode */
                r.x.cx  =  del_len;

                sregs.ds  =  FP_SEG (u_addr);       /* Segment of Read Buffer   */
                r.x.dx    =  FP_OFF (u_addr);       /* Offset  of Read Buffer   */

                sregs.es  =  FP_SEG (del_addr);     /* Segment of delimiter Buffer   */
                r.x.di    =  FP_OFF (del_addr);     /* Offset  of delimiter Buffer   */

                int86x(0x50, &r, &r, &sregs);
                return ( r.h.al);
}
```

```
/*
 * Virtual Memory Manager for XMS
 *
 * HIMEM.SYS 의 존재하에서 extended memory를 사용하기 위한 모듈
 *
*/

#include <dos.h>
#include <stdio.h>
#include <conio.h>
#include <alloc.h>
#include <mem.h>

#include "vm_xms.h"

#ifndef true
        #define true 1
        #define false 0
#endif

/*  Error no
0x80 "Not allowed function"
0x81 "Vdisk detected"
0x82 "A20 error"
0x90 "HMA not exist"
0x91 "HMA was used by another application"
0x92 "Value of DX register less than /HMAMIN"
0x93 "HMA not allocated"
0x94 "A20 line aleady enabled"
0xA0 "All Extended Memory Block allocated"
0xA1 "All Extended Memory Block used out"
0xA2 "Invailed handle"
0xA3 "Invailed source handle"
0xA4 "Invailed source offset"
0xA5 "Invailed destination handle"
0xA6 "Invailed destination offset"
0xA7 "Invailed copy length"
0xA8 "Overlap occurred in moving memory"
0xA9 "Parity error"
0xAA "Block not locked"
0xAB "Handle locked"
0xAC "Overflow locked block"
0xAD "Lock failure"
0xB0
0xB1
0xB2
*/

driver_func_t XMS_DRV = NULL;
int xmserrno = 0;
unsigned int freesize;
XmsMemCopy_t cpy;


int is_xms_install(void)
{
        _AX = 0x4300;
        geninterrupt(0x2f);
        if(_AL == 0x80)
                return true;
        return false;
}

void far *get_xms_drv_adrs(void)
{
        _AX = 0x4310;
        geninterrupt(0x2f);
        return MK_FP(_ES, _BX);
}

int get_xms_ver(void)
{
        if(XMS_DRV == NULL)
                return 0;
        _AH = 0;
        (XMS_DRV)();
        return _AX;
```

```
}

unsigned int get_max_free_xms(void)
{
        _AH = 8;
        (XMS_DRV)();

        xmserrno = _BL;
        return _AX;
}

unsigned int get_total_free_xms(void)
{
        _AH = 8;
        (XMS_DRV)();
        xmserrno = _BL;
        return _DX;
}

unsigned int xms_alloc(int kb)
{
        _AH = 9;
        _DX = kb;
        (XMS_DRV)();
        if(_AX)
                return _DX;

        xmserrno = _BL;
        return false;
}

unsigned int xms_free(int handle)
{
        _AH = 0x0a;
        _DX = handle;
        (XMS_DRV)();
        if(_AX)
                return true;

        xmserrno = _BL;
        return false;
}

int xms_copy(unsigned int sh, unsigned long sadrs,
                                        unsigned int dh, unsigned long dadrs, long len)
{
        asm push DS;

        cpy.length = len;
        cpy.srchandle = sh;
        cpy.srcoffset = sadrs;
        cpy.desthandle = dh;
        cpy.destoffset = dadrs;
        _DS = FP_SEG(&cpy);
        _SI = FP_OFF(&cpy);
        _AH = 0x0b;
        (XMS_DRV)();

        asm pop DS;

        if(_AX)
                return true;
        xmserrno = _BL;
        return false;
}

int xms_lock(unsigned handle)
{
        _AH = 0x0c;
        _DX = handle;

        (XMS_DRV)();
        if(_AX)
                return true;
        xmserrno = _BL;
        return false;
}
```

```
int xms_unlock(unsigned int handle)
{
        _AH = 0x0d;
        _DX = handle;

        (XMS_DRV)();
        if(_AX)
                return true;
        xmserrno = _BL;
        return false;
}

int get_handle_info(unsigned int handle, xms_handle_info_t *i)
{
        _AH = 0x0e;
        _DX = handle;

        (XMS_DRV)();
        if(_AX)
        {
                i->locked_blk = _BH;
                i->free_handle = _BL;
                i->blocksize = _DX;
                return true;
        }
        xmserrno = _BL;
        return false;
}

int resize_block(unsigned int handle, unsigned int newsize)
{
        _AH = 0x0f;
        _BX = newsize;
        _DX = handle;
        (XMS_DRV)();
        if(_AX)
                return true;
        xmserrno = _BL;
        return false;
}

/*
void main(void)
{
 int h, x, y;
 unsigned i;
 unsigned char far *buf;
 x = wherex();
 y = wherey();
 buf = farmalloc(51584l);
 if(buf== NULL)
 {
                return;
 }
 printf("buf ptr %Fp\n", buf);
 for(i = 0; i < 51584u; i++)
        *(buf+i) = i % 256;
 if(is_xms_install())
 {
        XMS_DRV = get_xms_drv_adrs();
        printf("Total free memory %dk bytes\n", get_total_free_xms());
        printf("Maximum free memory %dk bytes\n", get_max_free_xms());

        printf("Now allocate 150kb .. handle = %u\n", h = xms_alloc(152));

        if(h == false)
                printf("Allocation error\n");
        else
        {
                if(xms_copy(0, (long)FP_SEG(buf) * 65536l+(long)FP_OFF(buf),
                                                                h, 0l, 515841) == true)
                {
                        printf("Screen copy success... press any key\n");
                        getch();
                }
                else
```

```
                        printf("Screen copy fail\n");

                puts("press any key");
                getch();
//              setmem(buf, 51584u, 0);
                for(i=0; i<51584u; i++)
                        *(buf+i) = 0;
                xms_copy(h, 0l, 0, (long)FP_SEG(buf) * 65536l+(long)FP_OFF(buf), 51584l)
                for(i = 0; i < 51584u; i++)
                {
                        printf("\rTest....%u",i);
                        if(*(buf+i) != i % 256)
                                printf("error in %d\n", i);
                }
                gotoxy(x, y);
                printf("Now free 150kb..\n");
                if(xms_free(h))
                        printf("Success\n");
                else
                        printf("Error\n");
        }
 } else
        printf("No xms drive\n");
 farfree(buf);
}
*/

int init_xms(void)
{
 if(is_xms_install())
 {
        XMS_DRV = get_xms_drv_adrs();
        freesize =get_total_free_xms();
        if(freesize > 800)
                return true;
 }
 return false;
}
```

APPENDIX 4. LIST OF TOMATO IRRIGATION SYSTEM PROGRAM

```
/*************************************************************************
```

DC-DUTCH.C

```
Analog input channel list
main-ch0 : mux16 expansion (soil moisture)
main-ch1 : temperature - direct
main-ch2 : humidity - through voltage follower
main-ch3 : co2 - direct
main-ch4 : soil moisture (zest) - direct
main-ch5 : soil moisture (Fujiwara) - direct
main-ch6 : mux16 expansion (stem diameter, radiation)
main-ch7 : Internal Temperature

mux16-ch0  soil moisture 0.(NC)
mux16-ch1 . soil moisture 1
mux16-ch2 : soil moisture 2
mux16-ch3 : soil moisture 3
mux16-ch4 : soil moisture 4

mux16-ch5 : NC
mux16-ch6 : NC
mux16-ch7 : NC

mux16-ch8 : stem diameter 0
mux16-ch9 : stem diameter 1
mux16-ch10 : stem diameter 2
mux16-ch11 : stem diameter 3
mux16-ch12 : stem diameter 4

mux16-ch13 : radiation 0
mux16-ch14 : radiation 1
mux16-ch15 : NC

mux-ch0 : positive pulse
mux-ch1 : negative pulse
mux-ch3 : NC
mux-ch4 : NC

ff[0]  : year
ff[1]  : month
ff[2]  day
ff[3]  : hour
ff[4]  : minute
ff[5]  : second
ff[6]  : temperature
ff[7]  : humidity
ff[8]  : soil moisture 0
ff[9]  : soil moisture 1
ff[10] : soil moisture 2
ff[11] : soil moisture 3
ff[12] : soil moisture 4
ff[13] : soil moisture Fujiwara
ff[14] : NU
ff[15] : co2 concentration
ff[16] : stem diameter 0
ff[17] : stem diameter 1
ff[18] : stem diameter 2
```

```
                    ff[19] : stem diameter 3
                    ff[20] : stem diameter 4
                    ff[21] : radiation 0
                    ff[22] : radiation 1 - Reference
                    ff[23] : radiation summation
                    ff[24] : expected gain
                    ff[25] : daily gain of stem diameter 0
                    ff[26] : daily gain of stem diameter 1
                    ff[27] : daily gain of stem diameter 2
                    ff[28] : daily gain of stem diameter 3
                    ff[29] : daily gain of stem diameter 4
                    ff[30] : irrigation mode 0
                    ff[31] : irrigation mode 1
                    ff[32] : irrigation mode 2
                    ff[33] : number of leaf
                    ff[34] : number of truss
                    ff[35] : number of node
                    ff[36] : number of fruit
                    ff[37] : daily transpiration
                    ff[38] : biomass production
                    ff[39] : gross photo synthesis
                    ff[40] : leaf area index
                    ff[41] : irrigation need 0
                    ff[42] : irrigation need 1
                    ff[43] : irrigation need 2
                    ff[44] : instant transpiration
                    ff[45] : net radiation
                    ff[46] : vapor pressure deficit
                    ff[47] : stomatal resistance
                    ff[48] : boundary layer resistance
                    ff[49] : temperature wet bulb
                    Tomato transpiration model outputs...
*****************************************************************************/

#define RUNKERNEL 1
#define NTASKS 8
#define TASK0 0
#define TASK1 1
#define TASK2 2
#define TASK3 3
#define TASK4 4
#define TASK5 5
#define TASK6 6

#use    "default.h"
#use    "kdm.lib"
#use    "serial.lib"
#use    "rtk.lib"
#use    "vdriver.lib"

#define NORECORD 1440
#define CR                  0x0d
#define IBAUD   2400                // modem communication speed
#define TBUFSIZE 1024               // size of transmit buffer for modem
#define RBUFSIZE 1024               // size of receive buffer for modem

#define MANUAL  0
#define AUTO    1
#define BAUD    28800

#define INTERVAL 1
#define KEYPAD_SIZE 24
```

```
#ifdef  LK_COLS
#undef  LK_COLS
#endif

#define LK_COLS      20
#ifdef  LK_LINES
#undef  LK_LINES
#endif
#define LK_LINES     4

/*****************************
 declaration function prototypes
*****************************/

/* control functions */

void mux16_ch_n(int n, int ch);
void high_pulse();
void low_pulse();
int change_time();
void init_control_variable();
void record_finished();
void clear_sram();
void reset_datalogging();
void reset_comport();
void record_data();
void irrigation_control();
void cal_daily_gain();
void cal_real_value();
void read_sensor();
int lk_run_menu(char *call_menu, struct lk_menu *menu, int index)

/* model functions */

void store_xmem();
void restore_xmem();
void zero_variable();

void continue_simulation();

void init_simulation();
void run_daily_simulation();
void run_fast_simulation();
void read_data(void);
float maximum(float, float);
float minimum(float, float);
float vapor_pressure(float);
float vapor_pressure1(float);
float tables(float*, float*, float, int);
void init_variable(void);
void calc_temp_dep_factor(void);
void calc_hourly_photosynthesis(void);
void calc_maint_respiration(void);
void development_rate6(void);
void lossrate5(void);
void calc_drymatter_growth_rate6(void);

void calc_transpiration(void);
void calc_net_radiation(void);
void calc_boundary_layer_resistance(void);
void read_crop_parameter(void);
void read_greenhouse_parameter(void);
void clear_dayvalue(void);
void modify_data(void);

/*******************************************************
```

```
 declaration of variables
**********************************************************/

/* model variables */

static int reset_model;
static float irri_min;
static float no_iter;
static float dt_iter;
static float x_ageclass;                    //xbox
static float daily_leafdev_rate;            //rdvlv
static float xfpn[10];                      //xfpn
static float leafarea_on_truss[30];         //xla
static float dwleaf_on_truss[30];           //dwl
static float dwstem_on_truss[30];           //dws
static float leafage_on_truss[30];          //agls
static float dwfruit_on_truss[30][7];       //dwf
static float fruitage_on_truss[30][7];      //agf
static float dwfruits_on_truss[30];         //dwtr
static float pot_stemgrow[30];              //pgs
static float pot_leafgrow[30];              //pgl
static float pot_fruitgrow[30][15],         //pgf
static float abortfruit_on_truss[30];       //abnf
static float newfruit_on_truss[30];         //rcnf
static float fruitno_on_truss[30],          //xnft
static float inst_fruitno_on_truss;         //nft
static float x_no_on_truss[30],             //xnsft
static float total_abortfruit[30],          //abor
static float pot_leafexp_on_truss[30];      //ple
static float setfruit_on_truss[30],         //nsf
static float dwstem,                        //tdms
static float dwfruit,                       //tdmf
static float inst_maint_resp,               //rmaintf
static float inst_dwleaf_on_truss;          //dmgl
static float inst_dwfruit_on_truss,         //dmgf
static float init_dwleaf,                   //wlvsi
static float init_dwstem,                   //wstmi
static float photon_density;                //ppfd
static float co2_compensation;              //cd
static float qefficiency;                   //qe quantum use efficiency
static float inst_photosynthesis;           //gpf
static float inst_tempeffect_on_dev;        //temfcf
static float inst_leafdev_rate;             //rdvlvf
static float inst_fruitdev_rate;            //rdvfrf
static float q10;                           //q10
static float fruit_mresp_coeff,             //rmrf
static float leaf_mresp_coeff;              //rmrl
static float stem_mresp_coeff,              //rmrs
static float dwleaf;                        //tdml
static float dwleaf_modified;               //tdml2
static float leafarea_modified;             //plar2
static float init_leafarea_plant;           //plari
static float planting_density;              //plm2
static float extinction_coeff;              //xk
static float node_bet_firsttruss_fruitset;  //frlg
static float pot_fruitno_per_node;          //fpnpt
static float newinit_fruitno;               //trcnf
static float firstfruit_trussno;            //nbrup
static float fruitno;                       //tnf
```

```
static float nodeinit_rate;                 //genr
static float no_node;                       //plstn
static float init_nodeno;                   //plstni
static float tempeffect_on_devrate;         //temfac
static float no_truss;                      //nbru
static float no_leaf;                       //nblv
static float leafarea_plant;                //plar
static float newabort_fruitno;              //tabnf
static float abortfruit_ratio;              //fabor
static float max_abort;                     //abormx
static float sourcesink_ratio;              //sosir
static float abortfruit_no;                 //tabf
static float setfruit_no;                   //tnsf
static float fruitdev_rate;                 //rdvfr
static float petiole_leaf_ratio;            //frpt
static float stem_leaf_ratio;               //frst
static float min_sla;                       //slamn
static float firsttruss_nodeno;             //ftrusn
static float max_sla;                       //slamx
static float dwshoot_growrate;              //ascsp
static float dwshootgrow_truss;             //xasc
static float truss_per_leaf;                //tpl
static float fruitno_mature;                //tnmf
static float dwfruit_mature;                //dmmf
static float lai;
static float rlsi;
static float daily_transpiration;           //dtran
static int currentday;
static int currentrecord;
static float rlc;
static float rls;
static float rlg;

static float grow_efficiency;               //gref
static float grow_resp;                     //gresp
static float total_resp;                    //tresp
static float carbon_pool;                   //cpool
static float maint_resp;                    //rmaint
static float gross_photosynthesis;          //gp
static float biomass_product;               //rcdrw

static float ass_req_leaf;                  //asrl
static float ass_req_stem;                  //asrs
static float ass_req_fruit;                 //asrf
static float pot_grow_shoot;                //pngp
static float pot_grow_leaf;                 //ptnlvs
static float pot_grow_stem;                 //ptnstm
static float pot_grow_fruit;                //ptnfrt
static float vpd;
static float limit_vpd;                     //limit VPD for VPD vs Pmax curve
static float leaf_vpd;
static float ck;
static float vp_air;
static float vp_sat;
static float temp_air;
static float wetbulb_t;
static float t_wetbulb;
static float humidity;
static float co2;
static float global_solar;                  //gsol
static float solarradiation;
static float tau1;
static float tau2;
static float pmax;
static float top;
static float bot;
```

```
static float xm,
static float eps;
static float carbon_pool_max;                    //cpoolmx
static float temp_veg;
static float dl;
static float rahv;
static float rah;
static float transpiration;
static float delta;
static float rnv;
static float ltl;
static float ta;
static float rn;
static float delt;
static float temp_greenhouse,
static float temp_mulching,
static float temp_a;
static float temp_cover;
static float radlong_cover;
static float radlong_screen;
static float radlong_veget;
static float radlong_greenhouse,
static float radshort_veget,
static float epa;
static float epg;
static float reg,
static float trc;
static float rec;
static float trlc;
static float relc;
static float epc;
static float trm;
static float rrem;
static float trlm;
static float relm;
static float epm;
static float trs;
static float _res;
static float epss;
static float trls;
static float rels;
static float scp;
static float epv;
static float kdf,
static float rev,
static float trv;
static float abv;
static float revh;
static float trvd;
static float trlv;
static float relv;
static float plth,
static float bbx;
static float aax;
static float denn;
static float nonn;
static float ffww;
static float resv;
static float resc;
static float recd;
static float rsv;
static float qla;
static float tair;
static float qlc;
static float tc,
static float qls;
static float ts;
static float qlv;
static float tv;
static float qlm;
static float tm;
static float qlg,
```

```
static float tg;
static float relv1,
static float relv2;
static float relg;
static float rlv31;
static float relv3;
static float relv4;
static float rlv32;
static float rlv33;
static float rlv34;
static float rlv35;
static float rlv;
static float rnv1;
static float xxp;
static float ttc;
static float tta,
static float re;
static float nu_forced;
static float grashof;
static float nust;
static float nusm;
static float nusb,
static float nu_mixed;
//shared float NSTART;
//shared float NDAYS;
//shared float NFAST,
//shared float INTOUT,
//shared float TRGH;

/*****************************************************
 declaration of constants
*****************************************************/

static float age_class[12] =
        {0.0, 5.0, 15 0, 25.0, 35.0, 45.0, 55.0, 65.0, 75.0, 85.0, 95.0, 100.0}, //box
static float pot_leafexp_in_age[12] =
        {0 0, 0.00035, 0.00165, 0.00255, 0.0023, 0.00155, 0.0009,
        0 0005, 0.00025, 0 00015, 0 00005, 0 0};                                  //pol
static float pot_fruitgrow_in_age[12] =
        {0.0, 0.09, 0 25, 0.45, 0.57, 0.58, 0 51, 0.41, 0 31, 0.22, 0 11, 0.0};   //pof
static float pot_fruitno_on_truss[30] =
        {5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5};            //nbfpt
static float pmaxchange_on_temp[8] =
        {0.0, 0 67, 1.0, 1.0, 1.0, 1 0, 0.0, 0.0},                                //pgred
static float x_temp_pmax[8] =
        {0.0, 9 0, 12.0, 15 0, 21 0, 28.0, 35 0, 50 0},                           //tmpg
static float tempeffect_to_vegdev[6] = {0.0, 0.0, 0.0, 0 55, 1 0,  0 0};          //gentem
static float x_temp[6] = {0 0, 5.0, 9 0, 12 0, 28 0, 50 0},                       //xtem
static float maxrate_nodelmit [8] =
        {0 0, 0 55, 0.55, 0.55, 0.55, 0.55, 0.55, 0.55};                          //genrat
static float x_node[8] = {0.0, 5 0, 20.0, 43.0, 48.0, 60.0, 90.0, 200.0};         //xgen
static float leafdev_on_temp[9] =
        {0 0, 0 0, 0.0096, 0.0126, 0.0190, 0.0260, 0.0260, 0 0, 0 0},             //rdvlvt
static float x_leafdev[9] =
        {0.0,  9.0, 12 0, 15.0, 21 0, 28.0,  35 0, 50.0, 80.0};                  //xlv
static float fruitdev_on_temp[9] =
        {0 0, 0 0, 0.007, 0.016, 0.021, 0.024, 0.022, 0.02, 0.0};                //rdvfrt
static float x_fruitdev[9] =
        {0.0,  9.0, 15.0, 18.0, 21.0, 24.0,  28.0, 35.0, 50.0};                  //xfr
static float proportion_root[6] = {0.18, 0.15, 0.12, 0.08, 0 02, 0.02},           //proot
static float x_nodeno[6] = {1.0, 10.0, 20.0, 30.0, 50.0, 90 0};                   //xroot
static int totalcount,

static float cp = 1004.0;                          //specific heat of air at constant pressure(J/kg.K)
static float sig = 0.0000000567; //stefan-boltzman constant(w/m2.k)
static float gamma = 66 2;
static float rhoa = 1.204,
static float nhu = 0 0000155;              //kinematic viscosity(m2/s)
static float kapa = 0.0000222;             //thermal diffusivity(m2/s)
static float aex = 1.0/273 16,             //thermal expansion of air
static float lambda = (2500.25-2 365*20.0)*1000.0;    //latent hear of vaporization(J/kg)
```

```
/*****************************************
 xmem variables
*****************************************/
/* model xmem */
xdata xm_reset_model[2];
xdata xm_xfpn[40];                          //xfpn
xdata xm_leafarea_on_truss[120];            //xla
xdata xm_dwleaf_on_truss[120];              //dwl
xdata xm_dwstem_on_truss[120];              //dws
xdata xm_leafage_on_truss[120];             //agls
xdata xm_dwfruit_on_truss[840];             //dwf
xdata xm_fruitage_on_truss[840];            //agf
xdata xm_dwfruits_on_truss[120];            //dwtr
xdata xm_pot_stemgrow[120];                 //pgs
xdata xm_pot_leafgrow[120];                 //pgl
xdata xm_pot_fruitgrow[1800];               //pgf
xdata xm_abortfruit_on_truss[120],          //abnf
xdata xm_newfruit_on_truss[120];            //rcnf
xdata xm_fruitno_on_truss[120];             //xnft
xdata xm_x_no_on_truss[120];                //xnsft
xdata xm_total_abortfruit[120];             //abor
xdata xm_pot_leafexp_on_truss[120];         //ple
xdata xm_setfruit_on_truss[120];            //nsf

xdata xm_no_iter[2];
xdata xm_dt_iter[4],
xdata xm_x_ageclass[4],                     //xbox
xdata xm_daily_leafdev_rate[4];             //rdvlv
xdata xm_inst_fruitno_on_truss[4],          //nft
xdata xm_dwstem[4];                         //tdms
xdata xm_dwfruit[4];                        //tdmf
xdata xm_inst_maint_resp[4];                //rmaintf
xdata xm_inst_dwleaf_on_truss[4];           //dmgl
xdata xm_inst_dwfruit_on_truss[4];          //dmgf
xdata xm_init_dwleaf[4];                    //wlvsi
xdata xm_init_dwstem[4],                    //wstmi
xdata xm_photon_density[4],                 //ppfd
xdata xm_co2_compensation[4];               //cd
xdata xm_qefficiency[4];                    //qe quantum use efficiency
xdata xm_inst_photosynthesis[4];            //gpf
xdata xm_inst_tempeffect_on_dev[4];         //temfcf
xdata xm_inst_leafdev_rate[4];              //rdvlvf
xdata xm_inst_fruitdev_rate[4];             //rdvfrf
xdata xm_q10[4];                            //q10
xdata xm_fruit_mresp_coeff[4];              //rmrf
xdata xm_leaf_mresp_coeff[4],               //rmrl
xdata xm_stem_mresp_coeff[4];               //rmrs
xdata xm_dwleaf[4],                         //tdml
xdata xm_dwleaf_modified[4];                //tdml2
xdata xm_leafarea_modified[4];              //plar2
xdata xm_init_leafarea_plant[4],            //plari
xdata xm_planting_density[4];               //plm2
xdata xm_extinction_coeff[4],               //xk
xdata xm_node_bet_firsttruss_fruitset[4],   //frlg
xdata xm_pot_fruitno_per_node[4],           //fpnpt
xdata xm_newinit_fruitno[4],                //trcnf
xdata xm_firstfruit_trussno[4];             //nbrup
xdata xm_fruitno[4],                        //tnf
xdata xm_nodeinit_rate[4];                  //genr
xdata xm_no_node[4];                        //plstn
xdata xm_init_nodeno[4];                    //plstni
xdata xm_tempeffect_on_devrate[4],          //temfac
xdata xm_no_truss[4];                       //nbru
xdata xm_no_leaf[4];                        //nblv
xdata xm_leafarea_plant[4];                 //plar
xdata xm_newabort_fruitno[4];               //tabnf
xdata xm_abortfruit_ratio[4];               //fabor
xdata xm_max_abort[4];                      //abormx
xdata xm_sourcesink_ratio[4];               //sosir
xdata xm_abortfruit_no[4];                  //tabf
xdata xm_setfruit_no[4];                    //tnsf
xdata xm_fruitdev_rate[4];                  //rdvfr
```

```
xdata xm_petiole_leaf_ratio[4];                    //frpt
xdata xm_stem_leaf_ratio[4];                       //frst
xdata xm_min_sla[4];                               //slamn
xdata xm_firsttruss_nodeno[4];                     //ftrusn
xdata xm_max_sla[4];                               //slamx
xdata xm_dwshoot_growrate[4];                      //ascsp
xdata xm_dwshootgrow_truss[4];                     //xasc
xdata xm_truss_per_leaf[4];                        //tpl
xdata xm_fruitno_mature[4];                        //tnmf
xdata xm_dwfruit_mature[4];                        //dmmf
xdata xm_lai[4];
xdata xm_rlsi[4];
xdata xm_daily_transpiration[4];                   //dtran
xdata xm_currentday[2];
xdata xm_currentrecord[2];
xdata xm_rlc[4];
xdata xm_rls[4];
xdata xm_rlg[4];
xdata xm_grow_efficiency[4];                       //gref
xdata xm_grow_resp[4];                             //gresp
xdata xm_total_resp[4];                            //tresp
xdata xm_carbon_pool[4];                           //cpool
xdata xm_maint_resp[4];                            //rmaint
xdata xm_gross_photosynthesis[4];                  //gp
xdata xm_biomass_product[4];                       //rcdrw
xdata xm_ass_req_leaf[4];                          //asrl
xdata xm_ass_req_stem[4];                          //asrs
xdata xm_ass_req_fruit[4];                         //asrf
xdata xm_pot_grow_shoot[4];                        //pngp
xdata xm_pot_grow_leaf[4];                         //ptnlvs
xdata xm_pot_grow_stem[4];                         //ptnstm
xdata xm_pot_grow_fruit[4];                        //ptnfrt
xdata xm_vpd[4];
xdata xm_limit_vpd[4];                             //limit VPD for VPD vs Pmax curve
xdata xm_leaf_vpd[4];
xdata xm_ck[4];
xdata xm_vp_air[4];
xdata xm_vp_sat[4];
xdata xm_temp_air[4];
xdata xm_wetbulb_t[4];
xdata xm_t_wetbulb[4];
xdata xm_humidity[4];
xdata xm_co2[4];
xdata xm_global_solar[4];                          //gsol
xdata xm_solarradiation[4];
xdata xm_tau1[4];
xdata xm_tau2[4];
xdata xm_pmax[4];
xdata xm_top[4];
xdata xm_bot[4];
xdata xm_xm[4];
xdata xm_eps[4];
xdata xm_carbon_pool_max[4];                       //cpoolmx
xdata xm_temp_veg[4];
xdata xm_dl[4];
xdata xm_rahv[4];
xdata xm_rah[4];
xdata xm_transpiration[4];
xdata xm_delta[4];
xdata xm_rnv[4];
xdata xm_ltl[4];
xdata xm_ta[4];
xdata xm_rn[4];
xdata xm_delt[4];
xdata xm_temp_greenhouse[4];
xdata xm_temp_mulching[4];
xdata xm_temp_a[4];
xdata xm_temp_cover[4];
xdata xm_radlong_cover[4];
xdata xm_radlong_screen[4];
xdata xm_radlong_veget[4];
xdata xm_radlong_greenhouse[4];
```

```
xdata xm_radshort_veget[4];
xdata xm_epa[4];
xdata xm_epg[4];
xdata xm_reg[4];
xdata xm_trc[4];
xdata xm_rec[4];
xdata xm_trlc[4];
xdata xm_relc[4],
xdata xm_epc[4];
xdata xm_trm[4];
xdata xm_rrem[4];
xdata xm_trlm[4];
xdata xm_relm[4];
xdata xm_epm[4];
xdata xm_trs[4];
xdata xm__res[4];
xdata xm_epss[4];
xdata xm_trls[4];
xdata xm_rels[4];
xdata xm_scp[4],
xdata xm_epv[4];
xdata xm_kdf[4];
xdata xm_rev[4];
xdata xm_trv[4],
xdata xm_abv[4];
xdata xm_revh[4];
xdata xm_trvd[4];
xdata xm_trlv[4];
xdata xm_relv[4];
xdata xm_plth[4];
xdata xm_bbx[4],
xdata xm_aax[4];
xdata xm_denn[4],
xdata xm_nonn[4];
xdata xm_ffww[4],
xdata xm_resv[4],
xdata xm_resc[4],
xdata xm_recd[4];
xdata xm_rsv[4];
xdata xm_qla[4],
xdata xm_tair[4],
xdata xm_qlc[4],
xdata xm_tc[4];
xdata xm_qls[4],
xdata xm_ts[4],
xdata xm_qlv[4];
xdata xm_tv[4];
xdata xm_qlm[4];
xdata xm_tm[4];
xdata xm_qlg[4];
xdata xm_tg[4];
xdata xm_relv1[4];
xdata xm_relv2[4];
xdata xm_relg[4],
xdata xm_rlv31[4];
xdata xm_relv3[4];
xdata xm_relv4[4];
xdata xm_rlv32[4],
xdata xm_rlv33[4];
xdata xm_rlv34[4];
xdata xm_rlv35[4];
xdata xm_rlv[4],
xdata xm_rnv1[4];
xdata xm_xxp[4],
xdata xm_ttc[4];
xdata xm_tta[4];
xdata xm_re[4];
xdata xm_nu_forced[4];
xdata xm_grashof[4];
xdata xm_nust[4],
xdata xm_nusm[4],
xdata xm_nusb[4],
```

```
xdata xm_nu_mixed[4];

/*****************************************************
 control variables
*****************************************************/

int i, j, k, l, m, n, o, p, q, r, u, v, w, z, data;          // looping variables

/* control xmem */
xdata storage[144000];
xdata model_storage[4000];
xdata xm_record_count[2];
xdata xm_rad[4];
xdata xm_irri_need0[4];
xdata xm_irri_need1[4];
xdata xm_irri_need2[4];
xdata xm_temperature_10min[4];
xdata xm_t_wetbulb_10min[4];
xdata xm_humidity_10min[4];
xdata xm_co2_10min[4];
xdata xm_rad_10min[4];

shared float irri_need0;
shared float irri_need1;
shared float irri_need2;
shared int currenttime;
shared int myindex;
shared int indextest;
shared int tempstorage[200][50];

shared char op_rbuf[255];                    // rs232 comm. receiver buffer
shared char op_tbuf[255];                    // rs232 comm transmitter buffer
shared char op_cc, op_rcc;                   // rs232 comm r/t counter

shared long int record_count;                // important!!! - long int type for xdata access
shared long int send_count;
shared int recorded;                         // recorded in this minute
shared int controlled[3];                    // controlled!
struct tm t, lcdt;

shared int cleared;

shared unsigned now_year,now_month,now_day,now_hour,now_min,now_sec, old_min;
shared unsigned start_year,start_month, start_day,start_hour,start_min,start_sec;

shared float radiation_sum;
shared float radiation_10min;
shared float temperature_10min;
shared float t_wetbulb_10min;
shared float humidity_10min;
shared float co2_10min;
shared float expected_daily_gain;
shared float daily_gain[5];
shared float daily_gainA, daily_gainB, daily_gainC;
shared float soilmoistureSet;
shared float stemdiameterSet;
shared float gainSet;
shared float soilmoisture[5];
shared float soil_hujiwara;
shared float soil_zest;
shared float stemdiameter[5];
shared float radiation[2];
shared float temperature;
shared float rhumidity;
shared float co2_concentration;
shared float spare[4];

shared unsigned settime[3], holdtime[3];
shared int stime[3],htime[3];

shared int irrigation_timer[3];
shared int irrigation_mode[3];
```

```
shared int irrigation_status[3];

shared int ff[100];

int tab_keys [] = {   's', 'm', '*', 'c', ' ', ' ', '#', '+',
                      '\n', '4', ' ', 'r', '0', '1', '4', '7',
                      '.', '2', '5', '8', '-', '3', '6', '9' };

int task0(),task1(),task2(),task3(),task4(),task5(),task6(),backgnd();
int (*Ftask[])() = {task0,task1,task2,task3,task4,task5,task6,backgnd};

struct lk_menu treatmentA[20]=
{
        {1, "\x1bp100\x1bc\x1bp000\x1bcsoil moisture 1 \n%10.2f bar", &soilmoisture[1], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcsoil moisture 2-x \n%10.2f bar", &soilmoisture[2], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcstem diameter 0\n%10.1f um", &stemdiameter[0], 0, 0},
        {3, "\x1bp100\x1bc\x1bp000\x1bcsoil set point\n%10.1f bar", &soilmoistureSet, 0, 0},
        {3, "\x1bp100\x1bc\x1bp000\x1bcirrigation need A\n%10.1f g", &irri_need0, 0, 0},
        {22, "\x1bp100\x1bc\x1bp000\x1bcirrigation time", &stime[0],0, change_time},
        {22, "\x1bp100\x1bc\x1bp000\x1bcwaiting time", &htime[0],0, change_time},
        0
};

struct lk_menu treatmentB[20]=
{
        {1, "\x1bp100\x1bc\x1bp000\x1bcsoil moisture 3\n%10.2f bar", &soilmoisture[3], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcstem diameter 1\n%10.1f um", &stemdiameter[1], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcstem diameter 3\n%10.1f um", &stemdiameter[3], 0, 0},
        {3, "\x1bp100\x1bc\x1bp000\x1bcstem set point\n%10.1f %%", &stemdiameterSet, 0, 0},
        {3, "\x1bp100\x1bc\x1bp000\x1bcirrigation need B\n%10.1f g", &irri_need1, 0, 0},
        {22, "\x1bp100\x1bc\x1bp000\x1bcirrigation time", &stime[1],0, change_time},
        {22, "\x1bp100\x1bc\x1bp000\x1bcwaiting time", &htime[1],0, change_time},
        0
},

struct lk_menu treatmentC[20]=
{
        {1, "\x1bp100\x1bc\x1bp000\x1bcsoil moisture 4\n%10.2f bar", &soilmoisture[4], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcstem diameter 2\n%10.1f um", &stemdiameter[2], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcstem diameter 4\n%10.1f um", &stemdiameter[4], 0, 0},
        {3, "\x1bp100\x1bc\x1bp000\x1bcgain set point\n%10.1f um", &gainSet, 0, 0},
        {3, "\x1bp100\x1bc\x1bp000\x1bcirrigation need C\n%10.1f g", &irri_need2, 0, 0},
        {22, "\x1bp100\x1bc\x1bp000\x1bcirrigation time", &stime[2],0, change_time},
        {22, "\x1bp100\x1bc\x1bp000\x1bcwaiting time", &htime[2],0, change_time},
        0
},

struct lk_menu statusmonitor[50]=
{
        {1, "\x1bp100\x1bc\x1bp000\x1bctemperature\n%10.2f 'C", &temperature, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bctemp. wet\n%10.2f 'C", &t_wetbulb, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcrelative humidity\n%10.2f %%", &rhumidity, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcstem diameter 0\n%10.2f um", &stemdiameter[0], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcstem diameter 1\n%10.2f um", &stemdiameter[1], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcstem diameter 2\n%10.2f um", &stemdiameter[2], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcstem diameter 3\n%10.2f um", &stemdiameter[3], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcstem diameter 4\n%10.2f um", &stemdiameter[4], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcsoil moisture 0\n%10.2f bar", &soilmoisture[0], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcsoil moisture 1\n%10.2f bar", &soilmoisture[1], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcsoil moisture 2\n%10.2f bar", &soilmoisture[2], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcsoil moisture 3\n%10.2f bar", &soilmoisture[3], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcsoil moisture 4\n%10.2f bar", &soilmoisture[4], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcsoil water Hujiwara\n%10.2f pF", &soil_hujiwara, 0, 0},
//      {1, "\x1bp100\x1bc\x1bp000\x1bcsoil water Zest\n%10.2f pF", &soil_zest, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcco2 concentration\n%10.1f ppm", &co2_concentration, 0, 0}
        {1, "\x1bp100\x1bc\x1bp000\x1bcradiation 0\n%10.1f W/m2", &radiation[0], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcradiation 1\n%10.1f W/m2", &radiation[1], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcTotal radiation\n%11.1f J/day", &radiation_sum, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcirrigation mode 0\n% 5d", &irrigation_mode[0], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcirrigation mode 1\n% 5d", &irrigation_mode[1], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcirrigation mode 2\n% 5d", &irrigation_mode[2], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcexpected gain\n%10.1f", &expected_daily_gain, 0, 0},
```

```
        {1, "\x1bp100\x1bc\x1bp000\x1bcreal gain no 0\n%10.1f", &daily_gain[0], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcreal gain no 1\n%10.1f", &daily_gain[1], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcreal gain no 2\n%10.1f", &daily_gain[2], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcreal gain no 3\n%10.1f", &daily_gain[3], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcreal gain no 4\n%10.1f", &daily_gain[4], 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcreal gain tr A\n%10.1f", &daily_gainA, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcreal gain tr B\n%10.1f", &daily_gainB, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcreal gain tr C\n%10.1f", &daily_gainC, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcrecording index\n% 5d", &myindex, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcfirst rec. value\n% 5d", &indextest, 0, 0},
        {3, "\x1bp100\x1bc\x1bp000\x1bcwatering(ml/min)\n%10.1f ml/min", &irri_min, 0, 0},
        0
};

struct lk_menu modelmonitor[30]=
{
        {1, "\x1bp100\x1bc\x1bp000\x1bcrnv\n%10.2f", &rnv, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bctranspiration\n%10.2f g/day", &transpiration, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcvpd\n%10.2f", &vpd, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcinternal r\n%10.2f s/m", &rlsi, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcboundary r\n%10.2f s/m", &rah, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bctemp_air \n%10.2f 'C", &temp_air, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcwetbulb_t \n%10.2f 'C", &wetbulb_t, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcno of leaf\n%10.2f", &no_leaf, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcno of truss\n%10.2f", &no_truss, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcno of node\n%10.2f", &no_node, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcno of fruit\n%10.2f", &fruitno, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcno of set fruit\n%10.2f", &setfruit_no, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcno of abort fruit\n%10.2f", &abortfruit_no, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcsource/sink ratio\n%10.2f", &sourcesink_ratio, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcdaily transpiration\n%8.2f g/plant/day", &daily_transpiration, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcbiomass production\n%8.2f g/plant/day", &biomass_product, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcmaint. respiration\n%8.2f g/plant/day", &maint_resp, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcgross photosynthesis\n%8.2f g/plant/day", &gross_photosynthesis, 0, 0}
        {1, "\x1bp100\x1bc\x1bp000\x1bcdry weight of leaf\n%10.2f g", &dwleaf, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcdry weight of stem\n%10.2f g", &dwstem, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcdry weight of fruit\n%10.2f g", &dwfruit, 0, 0},
        {1, "\x1bp100\x1bc\x1bp000\x1bcleaf area index\n%10.2f m2/m2", &lai, 0, 0},
};

struct lk_menu setup[7]=
{
        {32,"\x1bp000\x1bc\n\x1bc\x1bp200\x1bcTreatment A", &treatmentA, 0, 0},
        {32,"\x1bp000\x1bc\n\x1bc\x1bp200\x1bcTreatment B", &treatmentB, 0, 0},
        {32,"\x1bp000\x1bc\n\x1bc\x1bp200\x1bcTreatment C", &treatmentC, 0, 0},
        {32,"\x1bp000\x1bc\n\x1bc\x1bp200\x1bcView Sensors", &statusmonitor, 0, 0},
        {32,"\x1bp000\x1bc\n\x1bc\x1bp200\x1bcView Growth Model", &modelmonitor, 0, 0},
        0
};

/*****************************************************
 main function
*****************************************************/


main()
{
        zero_variable();
        continue_simulation();
//      init_simulation();
        init_control_variable();
        ser_kill_s0();
        setdaisy(1);
        ser_init_s0(4, BAUD /1200);
        op_rcc=1;
        op_rbuf[0] = 0;
        ser_rec_s0(op_rbuf, &op_rcc);

        lk_init_keypad();
        lk_loadtab (tab_keys, sizeof (tab_keys));
        lk_settab (1);
```

```
        outport (DCNTL, 0x30);
        Reset_PBus();
        exp_init_n(0,0,0,0,-1, 1);
        exp_init_n(1,0,0,0,-1, 0);
        exp_init_n(2,0,0,0,-1, 0);
        exp_init_n(3,0,0,0,-1, 0);

//      recled(0);
        DI();
        init_kernel();                  // initialize runtime kernel
        run_every(TASK0, 20);           // every .2 seconds - Watch dog timer
        run_every(TASK1, 3000);         // every 30 seconds - Main
//      run_every(TASK2, 1000);         // every 10 seconds -
        run_every(TASK3, 3000);         // every 30 seconds - recording
        run_every(TASK4, 20);           // every .2 seconds  - communication
//      run_every(TASK6, 100);
        init_timer0 (9216);             // 400 Hz, no keypad driver
        EI();                           // start running
        backgnd();                      // this runs after everything else

}

/*****************************************************
 multi task functions
*****************************************************/

indirect task0()
{
    runwatch ();
    hitwd();
}

indirect task1()                // ADC, Recording, Control, Simulation
{
        tm_rd(&t);
        now_year = t.tm_year;           tempstorage[0][0] = now_year;
        now_month = t.tm_mon;           tempstorage[0][1] = now_month;
        now_day = t.tm_mday;            tempstorage[0][2] = now_day;
        now_hour = t.tm_hour;           tempstorage[0][3] = now_hour;
        now_min = t.tm_min;             tempstorage[0][4] = now_min;
        now_sec = t.tm_sec;             tempstorage[0][5] = now_sec;
        currenttime = ((t.tm_hour*60)+t.tm_min);

        read_sensor();                  // read from sensor channels and deposit to temporary array
        cal_real_value();               // calculate values for display
        if (now_min != old_min)         {
                old_min = now_min;
                recorded = 0;
                if (((now_min%10)==0) & (recorded==0)) // every 10 minutes and only one time
                {
                        record_data();                                                          //
record data to SRAM
                        run_fast_simulation();
                        if ((now_hour==0) & (now_min==0)) {
                                run_daily_simulation();
//                              printf("==============\n");
//                              printf("%10.3f %10.3f\n", daily_transpiration, no_node);
                        }
                        store_xmem();
                }
                cal_daily_gain();                       // calculate daily gain - preparing for control
                irrigation_control();           // Irrigation control routine
        }
        return;
}

indirect task2() {
//      printf("time passed!\n");
}

indirect task3() {}
```

```
indirect task4()                    // communication
{
        auto int kk, yy;
    EI();
//      printf("%d\n", op_rcc);
        if (op_rcc) return;
        if (op_rbuf[0]=='b')  {
                op_rbuf[0]=0;
                op_rcc=1;
                ser_rec_s0(op_rbuf, &op_rcc);
                for(yy=0,yy<144;yy++)           {
                        op_cc = 100;
                        for(kk=0;kk<100;kk++)
                                op_tbuf[kk] = xgetchar(storage+(send_count*100)+kk)
//                      if (op_tbuf[10]) printf("%d' %d\n", send_count, op_tbuf[10]);
                        ser_send_s0(op_tbuf, &op_cc);
                        while(op_cc) suspend(1);
                        send_count=(send_count+1)%1440;
                }
        }
        else if (op_rbuf[0]=='h')       {
                record_count = xgetint(xm_record_count);
                op_rbuf[0]=0;
                op_rcc=1;
                ser_rec_s0(op_rbuf, &op_rcc);
                op_cc=10;
                op_tbuf[0] = 'h';
                op_tbuf[1] = 'e';
                op_tbuf[2] = 'a';
                op_tbuf[3] = 'd';
                op_tbuf[4] = 'e';
                op_tbuf[5] = 'r';
                op_tbuf[6] = ' ';
                op_tbuf[7] = '.';
                op_tbuf[8] = 1440%256;
                op_tbuf[9] = (char) (1440/256);
                ser_send_s0(op_tbuf, &op_cc);
                while(op_cc) suspend(10);
                send_count = record_count;
        }
        else if ((op_rbuf[0]!='h') & (op_rbuf[0]!='b')) {
                reset_comport();
        }
}

indirect task5()                    // spare task
{
}

indirect task6()                    // spare task
{
}

indirect backgnd()
{
        int q, k, e, dummy, led_on;
        int pp;
        char hour, sec, min,

        while(1)
        {
                for(q=10000L;--q;);
                lk_setbeep (1);
                lk_led (led_on),
                led_on = (led_on == 1) ? 2 : 1;

                tm_rd(&lcdt);
                lk_printf("^x1bp000Irrigation controls");for(q=200L;--q;);
                lk_printf("^x1bp300    %02d/%02d/%02d %02d:%02d:%02d",
                        lcdt.tm_year,lcdt.tm_mon,lcdt.tm_mday,
                        lcdt.tm_hour,lcdt.tm_min,lcdt.tm_sec);
                for(q=200L;--q;);
```

```
                k = lk_kxget(0);
                switch(k) {
                        case 's' :
                                dummy=lk_kxget(0);
                                lk_setbeep(100);
                                lk_printf("\x1be");for(q=200L;--q;);
                                lk_run_menu("\x1bp300\x1bcMonitor and Set", setup, 0);
                                lk_setbeep(100);
                                dummy=lk_kxget(0);
                                lk_printf("\x1be");for(q=200L;--q;);
                                break;
                        case 'c' :
                                dummy=lk_kxget(0);
                                lk_setbeep(100);
                                lk_printf("\x1bp100reset comport"); for(q=200L;--q;);
                                reset_comport();
                                break;
                        case 'r' :
                                dummy=lk_kxget(0);
                                lk_setbeep(100);
                                lk_printf("\x1bp100Reset data logging"); for(q=200L;--q;)
                                reset_datalogging();
                                clear_sram();
                                break;
                        case 'm' :
                                dummy=lk_kxget(0);
                                lk_setbeep(100);
                                lk_printf("\x1bp100doing nothing"); for(q=200L;--q;);
                                break;
                }
        }
}

/*******************************************************
 function definition : control functions
*******************************************************/

void init_control_variable()    {
        auto int pp;
        tm_rd(&t);
        old_min = t.tm_min;
        now_min = old_min;
        myindex = 0;
        indextest = 0;
        stime[0] = mk_st(6,00,0);
        stime[1] = mk_st(6,00,0);
        stime[2] = mk_st(6,00,0);
        htime[0] = mk_st(1,30,0);
        htime[1] = mk_st(1,30,0);
        htime[2] = mk_st(1,30,0);
        irri_min = 35.0;
        for(p=0;p<5;p++)    {
                stemdiameter[p] = 0;
                soilmoisture[p] = 0;
        }
        for(p=0;p<150;p++)
                for (pp=0;pp<50;pp++)
                        tempstorage[p][pp] = 0;
        radiation[0] = 0;
        radiation[1] = 0;
        temperature = 0;
        rhumidity = 0;
        co2_concentration = 0;
        for (p=0;p<3;p++)   {
                irrigation_mode[p] = 0;
                irrigation_timer[p] = 0;
                irrigation_status[p] = 0;
        }
        cleared = 1;
        recorded = 0;
        soilmoistureSet = 0.2;
        controlled[0] = 1;
```

```
        controlled[1] = 1;
        controlled[2] = 1;

        record_count = xgetint(xm_record_count);
        send_count = 0;
//      radiation_sum = xgetfloat(xm_rad);
        expected_daily_gain = 0;
        daily_gain[0] = 0;
        daily_gain[1] = 0;
        daily_gain[2] = 0;
        daily_gain[3] = 0;
        daily_gain[4] = 0;
        daily_gainA = 0;
        daily_gainB = 0;
        daily_gainC = 0;
        change_time();
//      printf("%d %d %d %d\n", settime[1],holdtime[1], settime[2],holdtime[2])
}

void reset_datalogging()        {
        init_control_variable();
        record_count = 0;
        send_count = 0;
        cleared = 1;
        myindex = 0;
        xputint(xm_record_count, 0);
        xputfloat(xm_rad, 0.0);
        radiation_sum = 0.0;
}

void clear_sram()    {
        auto long int q;
        init_control_variable();
        for(q=0;q<144000;q++)           {
                xputchar(storage+q, 0);
        }
        myindex = 0;
        record_count = 0;
        send_count = 0;
        cleared = 1;
        xputint(xm_record_count, 0);
        xputfloat(xm_rad, 0.0);
        radiation_sum = 0.0;

}

change_time()
{
        settime[0] = st_hour(stime[0])*60 + st_min(stime[0]);
        settime[1] = st_hour(stime[1])*60 + st_min(stime[1]);
        settime[2] = st_hour(stime[2])*60 + st_min(stime[2]);
        holdtime[0] = st_hour(htime[0])*60 + st_min(htime[0]);
        holdtime[1] = st_hour(htime[1])*60 + st_min(htime[1]);
        holdtime[2] = st_hour(htime[2])*60 + st_min(htime[2]);
}

void reset_comport()
{
        ser_kill_s0();
        setdaisy(1);
        ser_init_s0(4, BAUD /1200);
        op_rcc=1;
        op_rbuf[0] = 0;
        ser_rec_s0(op_rbuf, &op_rcc);
}

void read_sensor()
{
        auto int ii, jj, kk, cc;
        float sum;
        float positive, negative;
        sum = 0.0;
```

```
        kk = 0;

        sum = 0.0;                                              // Read Temperature
        ad_rd12(9);         //dummy reading... Greg Young recommended
        for (ii=0;ii<256;ii++) sum += (float) abs(ad_rd12(9));
        ff[6] = (int) (sum / 256.0);

        sum = 0.0;                                              // Read Temperature wet bulb
        ad_rd12(10);        //dummy reading... Greg Young recommended
        for (ii=0;ii<256;ii++) sum += (float) abs(ad_rd12(10));
        ff[49] = (int) (sum / 256.0);

        temperature = ff[6]*50.0/3146.0-12.0;
        t_wetbulb = ff[49]*50.0/3146.0-12.0;
//      temperature = 22.0;
//      t_wetbulb = 17.5;
        rhumidity = (float) (vapor_pressure(t_wetbulb)/vapor_pressure(temperature)*100.0);
//      rhumidity = ff[7]*60.0/751.0-0.439;
        ff[7] = (int) rhumidity;

        for(ii=0;ii<5;ii++)     {                               // Read soil moisture block
                positive = 0.0;
                negative = 0.0;
                mux16_ch_n(1, ii);
                for(jj=0;jj<10;jj++)    {
                        mux_ch_n(3,kk,0);
                        high_pulse();
                        ad_rd12(0);
                        if (kk) for(cc=0;cc<10;cc++) positive += (float) abs(ad_rd12(0));
                        else  for(cc=0;cc<10;cc++) negative += (float) abs(ad_rd12(0));
                        low_pulse();
                        for(cc=0;cc<10;cc++) ad_rd12(0);
                        if (!kk) kk=1;
                        else kk=0;
                }
                ff[8+ii] = (int) ((positive+negative)/100.0);
        }

        ff[8] = ff[8] - 0;
        ff[9] = ff[9] - 390;
        ff[10] = ff[10] - 250;
        ff[11] = ff[11] - 135;
        ff[12] = ff[12] - 300;

        sum = 0.0;                                              // Read soil moisture Hujiwara
        ad_rd12(13);        //dummy reading.  Greg Young recommended
        for (ii=0;ii<256;ii++) sum += (float) abs(ad_rd12(13));
        ff[13] = (int) (sum / 256.0);

        sum = 0.0;                                              // Read radiation from HANS
7mv/1000Wm-2
        ad_rd12(12);        //dummy reading... Greg Young recommended
        for (ii=0;ii<256;ii++) sum += (float) abs(ad_rd12(12));
        ff[21] = (int) (sum / 256.0);
        ff[21] = ff[21] - 31;
        sum = 0.0;                                              // Read co2 concentration
        ad_rd12(11);        //dummy reading... Greg Young recommended
        for (ii=0;ii<256;ii++) sum += (float) abs(ad_rd12(11));
        ff[15] = (int) (sum / 256.0);

        for(ii=0;ii<5;ii++)     {                               // Read stem diameter sensor
                sum = 0.0;
                mux16_ch_n(1, ii+8);
                ad_rd12(14);                            //dummy reading... Young recommended
                for (jj=0;jj<256;jj++) {
                        sum += (float) ad_rd12(14);
                }
                ff[ii+16] = (int) (sum / 256.0);
        }

/*      mux16_ch_n(1, 13);                      // Read radiation 0
        sum = 0.0;
```

```
        ad_rd12(6);                                  //dummy reading... Greg Young recommended
        for (jj=0;jj<256;jj++) sum += (float) ad_rd12(6);
        ff[21] = (int) (sum / 256);
*/
        mux16_ch_n(1, 14);                           // Read radiation 1
        sum = 0.0;
        ad_rd12(6);                                  //dummy reading... Greg Young recommended
        for (jj=0;jj<256;jj++) sum += (float) ad_rd12(6);

        ff[22] = (int) (sum / 256);
//      ff[21] = ff[21];
//      ff[22] = ff[22];
        if (ff[21]<0) ff[21] = 0;
        if (ff[22]<0) ff[22] = 0;
        ff[23] = (int) (radiation_sum/1000.0);       // Calculate radiation total
        ff[24] = (int) (expected_daily_gain*10.0);   // Calculate expected gain
        ff[25] = (int) (daily_gain[0]*10.0);         // deposit daily gain 0
        ff[26] = (int) (daily_gain[1]*10.0);         // deposit daily gain 1
        ff[27] = (int) (daily_gain[2]*10.0);         // deposit daily gain 2
        ff[28] = (int) (daily_gain[3]*10.0);         // deposit daily gain 3
        ff[29] = (int) (daily_gain[4]*10.0);         // deposit daily gain 4
        ff[30] = (int) irrigation_mode[0];           // Irrigation mode A
        ff[31] = (int) irrigation_mode[1];           // Irrigation mode B
        ff[32] = (int) irrigation_mode[2];           // Irrigation mode C
        ff[33] = (int) no_leaf;
        ff[34] = (int) no_truss;
        ff[35] = (int) no_node;
        ff[36] = (int) fruitno;
        ff[37] = (int) (daily_transpiration);
        ff[38] = (int) (biomass_product*100.0);
        ff[39] = (int) (gross_photosynthesis*100.0);
        ff[40] = (int) (lai*1000.0);
        ff[41] = (int) irri_need0;
        ff[42] = (int) irri_need1;
        ff[43] = (int) irri_need2;
        ff[44] = (int) transpiration;
        ff[45] = (int) rnv;
        ff[46] = (int) vpd;
        ff[47] = (int) rlsi;
        ff[48] = (int) rah;

        for(jj=6;jj<23;jj++)                         // temporary storage
                tempstorage[myindex][jj] = ff[jj];
        for (ii=23;ii<49;ii++)
                tempstorage[0][ii] = ff[ii];
        tempstorage[myindex][49] = ff[49];

        indextest = tempstorage[0][6];
        myindex = (myindex+1)%200;

//      printf("%d %d\n", myindex, tempstorage[myindex-1][6]);
}

// calibration routine here;
void cal_real_value()
{
        auto int ii;
        auto float vs_vx, rs;
        for (ii=0;ii<5;ii++)    {
                if (ff[8+ii] > 2122) ff[8+ii] = 2122;
                if (ff[8+ii] < 10) ff[8+ii] = 10;
                vs_vx = ((500.0/2122.5)*ff[8+ii]) / 500.0;
                if (vs_vx > 0.8333) vs_vx = 0.8333;
                if (vs_vx < 0.0566) vs_vx = 0.0566;
                rs = vs_vx/(1-vs_vx);
                soilmoisture[ii] = (0.06516 + 0.95117*rs - 0.25159*pow(rs,2.0) -
                        0.03736*pow(rs,3.0) + 0.03273*pow(rs,4.0) - 0.00394*pow(rs,5.0));
//              soilmoisture[ii] = (0.15836 + 6.1445*rs - 8.4189*pow(rs,2.0) +
//                      9.2493*pow(rs,3.0) - 3.1685*pow(rs,4.0) + 0.33392*pow(rs,5.0));
        }
        soil_hujiwara = (0.000445*(float) ff[13]) + 1.15023;        // unit: pF
//      soil_hujiwara = 2.1;                                        //temporary
```

```
        soil_zest = (0.0018303* (float) ff[14]) - 0.748593;          // unit: pF
        co2 concentration = (float) ff[15]*3000/3146-749.5;          // unit: ppm
//      co2_concentration = 420.0;                                   //temporary
        stemdiameter[0] = (6.710546 * 0.1 * ff[16]);                 // + 74.928823);
        stemdiameter[1] = (6.21414 * 0.1 * ff[17]);                  // - 220.977780);
        stemdiameter[2] = (6.407376 * 0.1 * ff[18]);                 // - 150.361135);
        stemdiameter[3] = (6.769002* 0.1 * ff[19]);                  // - 623.923672);
        stemdiameter[4] = (6.679632 * 0.1 * ff[20]);                 // - 746.868477);

        //radiation[0] = (float) (0.9496676 * ff[21]);
        radiation[0] = (float) (0.77110769945 * ff[21]);
//      radiation[0] = (float) ( 0.385553849725 * ff[21]);

        if (radiation[0] >1000) radiation[0] = 1000.0;

//      radiation[0] = (float) 1.514238705388* ff[21];       // real radiation value 0 * 0.7571193526943
//      radiation[1] = (float) 1.5422153989 * ff[22];        // real radiation value 1
        if (radiation[0] < 0) radiation[0] = 0;
        if (radiation[1] < 0) radiation[1] = 0;
        radiation_sum = xgetfloat(xm_rad);
        radiation_sum = radiation_sum + (radiation[0]*30.0); // radiation summation
        xputfloat(xm_rad, radiation_sum);                    // record it for reset
}

void cal_daily_gain()
{
        auto long int yesterday_record;
        auto int yy, tt;

        yesterday_record = ((record_count-1) - 144);
        if (yesterday_record < 0) yesterday_record += 1440;

        yy = xgetint(storage+((yesterday_record)*100+16*2));
        tt = xgetint(storage+((record_count-1)*100+16*2));
        daily_gain[0] = (0.6710546) * (float)(tt-yy);

        yy = xgetint(storage+((yesterday_record)*100+17*2));
        tt = xgetint(storage+((record_count-1)*100+17*2));
        daily_gain[1] = (0.621414) * (float)(tt-yy);

        yy = xgetint(storage+((yesterday_record)*100+18*2));
        tt = xgetint(storage+((record_count-1)*100+18*2));
        daily_gain[2] = (0.6407376) * (float)(tt-yy);

        yy = xgetint(storage+((yesterday_record)*100+19*2));
        tt = xgetint(storage+((record_count-1)*100+19*2));
        daily_gain[3] = (0.6769002) * (float)(tt-yy);

        yy = xgetint(storage+((yesterday_record)*100+20*2));
        tt = xgetint(storage+((record_count-1)*100+20*2));
        daily_gain[4] = (0.6679632) * (float)(tt-yy);

//      printf("%d %d %d %d %10.1f\n", (int)yesterday_record, (int) record_count,tt, yy, daily_gain[1])

        daily_gainA = daily_gain[0];
        daily_gainB = (daily_gain[1]+daily_gain[3])/2.0;
        daily_gainC = (daily_gain[2]+daily_gain[4])/2.0;
        expected_daily_gain = 30.1 + (radiation_sum/1000000.0) * 3.6;
        return;
}

void irrigation_control()
{
        if (currenttime < settime[0])        controlled[0] = 0;
        if (currenttime < settime[1])        controlled[1] = 0;
        if (currenttime < settime[2])        controlled[2] = 0;

        holdtime[0] = (int) irri_need0/irri_min;
        holdtime[1] = (int) irri_need1/irri_min;
        holdtime[2] = (int) irri_need2/irri_min;
        if (holdtime[0]>60) holdtime[0] = 60;
        if (holdtime[1]>60) holdtime[0] = 60;
```

```
f (holdtime[2]>60) holdtime[0] = 60;

f ((currenttime >= settime[0]) && !controlled[0])    {
        if(!irrigation_status[0])          {
                if(((soilmoisture[1]+soilmoisture[2])/2.0) > soilmoistureSet)       {
                        Set_PBus_Relay(3,1,1);
                        irrigation_mode[0] = holdtime[0];
                        irrigation_status[0] = 1;
                        irrigation_timer[0] = 1;
                }
                else {
                        controlled[0] = 1;
                        radiation_sum = 0.0;
                        xputfloat(xm_rad,0.0);
                }
        }
        else if (irrigation_status[0])     {
                if (irrigation_timer[0] < holdtime[0])       {
                        irrigation_mode[0]--;
                }
                if (irrigation_timer[0] >= holdtime[0])      {
                        Set_PBus_Relay(3,1,0);
                        irrigation_mode[0] = 0;
                        irrigation_status[0] = 0;
                        controlled[0] = 1;
                        radiation_sum = 0.0;
                        xputfloat(xm_rad,0.0);
                        irri_need0 = 0.0;
                }
                irrigation_timer[0] ++;
        }
}
if ((currenttime >= settime[1]) && !controlled[1]) {
        if(!irrigation_status[1])          {
                if (daily_gainB < (0.9*expected_daily_gain))          {
                        Set_PBus_Relay(3,2,1);
                        irrigation_mode[1] = holdtime[1];
                        irrigation_status[1] = 1;
                        irrigation_timer[1] = 1;
                }
                else {
                        controlled[1] = 1;
                        radiation_sum = 0.0;
                        xputfloat(xm_rad,0.0);
                }
        }
        else if (irrigation_status[1])     {
                if (irrigation_timer[1] < holdtime[1])       {
                        irrigation_mode[1]--;
                }
                if (irrigation_timer[1] >= holdtime[1])      {
                        Set_PBus_Relay(3,2,0);
                        irrigation_mode[1] = 0;
                        irrigation_status[1] = 0;
                        controlled[1] = 1;
                        radiation_sum = 0.0;
                        xputfloat(xm_rad,0.0);
                        irri_need1 = 0.0;
                }
                irrigation_timer[1] ++;
        }
}
if ((currenttime >= settime[2]) && !controlled[2]) {
        if(!irrigation_status[2])          {
                if (daily_gainC < 0.0)          {
                        Set_PBus_Relay(3,3,1);
                        irrigation_mode[2] = holdtime[2];
                        irrigation_status[2] = 1;
                        irrigation_timer[2] = 1;
                }
                else {
                        controlled[2] = 1;
```

```
                                        radiation_sum = 0.0;
                                        xputfloat(xm_rad,0.0);
                                }
                        }
                        else if (irrigation_status[2])      {
                                if (irrigation_timer[2] < holdtime[2])      {
                                        irrigation_mode[2]--;
                                }
                                if (irrigation_timer[2] >= holdtime[2])      {
                                        Set_PBus_Relay(3,3,0);
                                        irrigation_mode[2] = 0;
                                        irrigation_status[2] = 0;
                                        controlled[2] = 1;
                                        radiation_sum = 0.0;
                                        xputfloat(xm_rad,0.0);
                                        irri_need2 = 0.0;
                                }
                                irrigation_timer[2] ++;
                        }
                }
                if (irrigation_status[0] | irrigation_status[1] | irrigation_status[2])
                        Set_PBus_Relay(3,0,1);
                else Set_PBus_Relay(3,0,0);
                return;
}

void record_data()
{
                auto float summation;
                auto int max;
                max = myindex;
                recorded = 1;
                for (u=6;u<23;u++)   {                                          // everage value recording
                        summation = 0.0;
                        if (max!= 0)        {
                                for(v=0;v<max;v++) {          summation += (float) tempstorage[v][u];   }
                                tempstorage[0][u] = (int) (summation/(max*1.0));
                        }
                        else {      tempstorage[0][u] = 0;          }
                }
                summation = 0.0;
                if (max!= 0)        {
                        for(v=0;v<max;v++) {          summation += (float) tempstorage[v][49];  }
                        tempstorage[0][49] = (int) (summation/(max*1.0));
                }
                else {      tempstorage[0][49] = 0;          }

                temperature_10min = (float) tempstorage[0][6]*50.0/3146.0-12.0;
                t_wetbulb_10min = (float) tempstorage[0][49]*50.0/3146.0-12.0;
//              temperature_10min = 22.0;
//              t_wetbulb_10min = 17.5;
                humidity_10min = (float) (vapor_pressure(t_wetbulb_10min) /
                                                                                vapor_pressure(temperature_10min)
* 100.0);
//              humidity_10min = (float) tempstorage[0][7]*60.0/751.0-0.439;
                radiation_10min = (float) (0.77110769945 * tempstorage[0][21]);
                co2_10min = (float) tempstorage[0][15]*3000.0/3146.0-749.5;
//              co2_10min = 420.0;
                for(v=0;v<50;v++)                                               // recording to SRAM
                        xputint(storage+(record_count*100+v*2), tempstorage[0][v]);

                record_count=(record_count+1)%1440;
                xputint(xm_record_count, (int) record_count);
//              xputfloat(xm_rad, radiation_sum);
                record_count = xgetint(xm_record_count);
                myindex = 0;
//                      printf("record count %d\n", record_count);
//                      for(j=0;j<50;j++)       {
//                              k = xgetint(storage+((record_count-1)*100+j*2));
//                              printf("%5d",k);
//                      }
//                      printf("\n");
```

```
        return;
}


/*****************************************************
 simulation functions
*****************************************************/

float maximum(float x0, float x1)          {
        if (x0>x1) return x0,
        else return x1,
}

float minimum(float x0, float x1)          {
        if (x0<x1) return x0;
        else return x1;
}

float vapor_pressure(float x)    {
        float r;
        r = 610 78 * exp (17.269 * x / (x+237 3));
        return r;
}

float vapor_pressure1(float x)   {
        float r;
        r = 610.78*exp(17 269*x/(x+237.3))*(4097.9337/(pow((x+237.30),2)))
        return r;
}

float tables(float* y, float* x, float dummy, int n)    {
        int i;
        float r;
        for(i=1;i<n;i++)        {
                if (dummy > x[i]) continue;
                else break,
        }
        if (dummy < x[1]) i = 1;

        r = (y[i-1]+(y[i]-y[i-1])/(x[i]-x[i-1])*(dummy-x[i-1]));
//      printf("%6.3f \n", r);
        return (r);
},

void init_variable(void)          {
        int i,j;
        abortfruit_no = 0.0;
        setfruit_no = 0.0;
        fruitno = 0.0;

        dwleaf_on_truss[0] = init_dwleaf;
        dwstem_on_truss[0] = init_dwstem;
        dwfruits_on_truss[0] = 0.0,

        leafarea_plant = init_leafarea_plant,
        leafarea_modified = leafarea_plant;
        leafarea_on_truss[0] = leafarea_plant;
        total_abortfruit[0] = 0.0;

        // the organ age at starting date is very important for
        // the final weight of first organ !!!
        leafage_on_truss[0] = 0 3;
        // here, up to 30 trusses may be simulated
        for (i=1;i<30,i++)     {
                dwleaf_on_truss[i] = 0.0;
                dwstem_on_truss[i] = 0.0;
                leafage_on_truss[i] = 0.0;
                leafarea_on_truss[i] = 0 0;
                total_abortfruit[i] = 0.0;
                fruitno_on_truss[i] = 0.0;
                dwfruits_on_truss[i] = 0.0;
        }
```

```
        for (i=0;i<30;i++)    {
                for (j=0;j<7;j++)    {
                        dwfruit_on_truss[i][j] = 0.0;
                        fruitage_on_truss[i][j] = 0.0;
                }
        }
        // set initial conditions on a per plant basis
        no_node = init_nodeno;
        // cpool is supposed to start at 10% leaf d.wt
        carbon_pool = 0.1 * init_dwleaf;
        dwstem = init_dwstem;
        dwleaf = init_dwleaf;
        dwfruit = 0.0;
        inst_dwfruit_on_truss = 0.0;
        inst_dwleaf_on_truss = init_dwleaf;
        return;
}

void calc_temp_dep_factor(void)
{
        // temperature effect on development rate
        inst_tempeffect_on_dev = (float) tables(tempeffect_to_vegdev, x_temp, temp_air, 6),
        //leaf aging
        inst_leafdev_rate = (float) tables(leafdev_on_temp, x_leafdev, temp_air, 9);
        //fruit aging
        inst_fruitdev_rate = (float) tables(fruitdev_on_temp, x_fruitdev, temp_air, 9),
//      printf("%d %5.3f %5.3f %5.3f \n", currentrecord, inst_tempeffect_on_dev, inst_leafdev_rate
//              inst_fruitdev_rate);
        return;
}

void calc_hourly_photosynthesis(void)
{
        int i;
        photon_density = 2.1479 * global_solar;

        //parameters of the photosynthesis equation(Acock, 1991)
        co2_compensation = 50.0 * exp(0.0295*(temp_air - 23.0)),
        qefficiency = 0.084 * ((co2 - co2_compensation)/co2),
        inst_photosynthesis = 0.0,

        //effect of co2 on Pmax (from Gainesville)
        tau1 = 0.10;
        tau2 = 0.03;
        pmax = tau1 * co2;
        if (co2>1500.0)       pmax = tau1*1500.0 + tau2*(co2-1500.0);

        //reduction of Pmax at extreme temperature
        pmax = pmax * tables(pmaxchange_on_temp, x_temp_pmax, temp_air, 8);
        if (photon_density<0.001)       return;
        //reduction of Pmax by VPD
        if (vpd >= limit_vpd)           {
                pmax = pmax * exp(ck*(vpd/1000.0-limit_vpd));
        }
        //Acock's model
        top = (1.0-xm)*pmax + qefficiency * extinction_coeff * photon_density;

        bot = (1.0-xm)*pmax + qefficiency * extinction_coeff * photon_density *
                exp(-1.0 * extinction_coeff * leafarea_modified * planting_density);

        inst_photosynthesis = (pmax / extinction_coeff)* log(top/bot);
        //conversion from co2 to CH2O (30/44 = 0.682)
        inst_photosynthesis = inst_photosynthesis * 0.682;
        //conversion of inst_photosynthesis from m/m2.s to g/m2.day
        // m/m2.s * 0.000044g/m * 3600s/h * 24h/d

        inst_photosynthesis = inst_photosynthesis * 3.8016 / planting_density;
        return;
}

void calc_maint_respiration(void)
{
```

```
        q10    1.4,
        leaf_mresp_coeff= 0.03;                    //maintenance respiration rate at 25'C
        stem_mresp_coeff = 0.015;
        fruit_mresp_coeff = 0.01;
        inst_maint_resp = (leaf_mresp_coeff * dwleaf_modified +
                stem_mresp_coeff * dwstem +
                fruit_mresp_coeff * inst_dwfruit_on_truss) *
                pow(q10, (0.1*(temp_air-25.0))) ;
        return;
}

void development_rate6(void)
{
        int tt,i,j;
        float xx;
        //node number
        nodeinit_rate = tempeffect_on_devrate * tables(maxrate_nodeinit, x_node, no_node, 8);
        no_node = no_node + nodeinit_rate;

        //truss number
        no_truss = (int) ((no_node - firsttruss_nodeno +
                ((1.0 + truss_per_leaf)/truss_per_leaf)) *
                (truss_per_leaf / (1.0 + truss_per_leaf)));
        no_truss = maximum(0.0, no_truss),

        //number of trusses bearing fruit
        firstfruit_trussno = (int) ((no_node - firsttruss_nodeno - node_bet_firsttruss_fruitset +
                (1.0 + truss_per_leaf)/truss_per_leaf) * truss_per_leaf/(1.0+truss_per_leaf));
        firstfruit_trussno = maximum(0.0, firstfruit_trussno),

        //leaf number
        no_leaf = (int) no_node - no_truss;

        //number of fruits (diameter>20mm) on each truss
        //note:FPN is now the ratio of fruit initiation on a truss per node initiation
        newinit_fruitno = 0.0,
        for (i=0;i<firstfruit_trussno;i++) {
                xx = fruitno_on_truss[i],
                //FPN is now replaced by FPNPT=0.8
                fruitno_on_truss[i] = minimum(pot_fruitno_on_truss[i], fruitno_on_truss[i] + nodeinit_rate
pot_fruitno_per_node);
                newfruit_on_truss[i] = fruitno_on_truss[i] - xx,
                newinit_fruitno = newinit_fruitno + newfruit_on_truss[i];
                //rncf[i] and trcnf is used in losrate
        }
        fruitno = fruitno + newinit_fruitno,

        /*********************************************************
         aging and sink strength(determined for leaves and stems
         and for each fruit on each reproductive unit
        *********************************************************/
        pot_grow_leaf = 0.0,
        pot_grow_stem = 0.0;
        pot_grow_fruit = 0.0,
        //leaves and stems
        //new leaves start aging when a new truss appears
        //(2 leaves below and 1 leaf above)
        for (i=0;i<(int)no_truss;i++)     {
                if (leafage_on_truss[i]==(-1.0 * eps))      {
                        pot_leafexp_on_truss[i] = 0.0;
                        pot_leafgrow[i] = 0.0;
                        pot_stemgrow[i] = 0.0;
                }
                else {
                        leafage_on_truss[i] = minimum(1.0, (leafage_on_truss[i] + daily_leafdev_rate));
                        x_ageclass = (float) (100.0 * leafage_on_truss[i]);

                        if (i==0)  {
                                pot_leafexp_on_truss[i] = tempeffect_on_devrate *
                                maximum(0.0, tables(pot_leafexp_in_age, age_class, x_ageclass, 12))
                                * firsttruss_nodeno;
                        }
```

```
                        else        {
                                    pot_leafexp_on_truss[i] = tempeffect_on_devrate *
                                    maximum(0.0, tables(pot_leafexp_in_age, age_class, x_ageclass, 12))
                                    / truss_per_leaf;
                        }
                        pot_leafgrow[i] = pot_leafexp_on_truss[i] * (1.0 + petiole_leaf_ratio) / min_sla;
                        pot_stemgrow[i] = pot_leafgrow[i] * stem_leaf_ratio;
            }
            pot_grow_leaf = pot_grow_leaf + pot_leafgrow[i];
            pot_grow_stem = pot_grow_stem + pot_stemgrow[i];
//          printf("---%10.5f %10.5f %10.5f %10.5f %10.5f %10.5f\n", pot_grow_shoot, pot_grow_leaf, pot_grow_stem,
pot_grow_fruit,
//                  leafage_on_truss[i],daily_leafdev_rate );

        }

        // fruit
        for(i=0;i<no_truss;i++)         {
                for(j=0;j<(int)fruitno_on_truss[i];j++)         {
                        fruitage_on_truss[i][j] = minimum(1.0, fruitage_on_truss[i][j] + fruitdev_rate);
                        x_ageclass = 100.0 * fruitage_on_truss[i][j];
                        if (dwfruit_on_truss[i][j]<0.0)    pot_fruitgrow[i][j] = 0.0;
                        else {
                                pot_fruitgrow[i][j] = tempeffect_on_devrate *
                                maximum(0.0, tables(pot_fruitgrow_in_age, age_class, x_ageclass, 12));
                                pot_grow_fruit = pot_grow_fruit + pot_fruitgrow[i][j];
                        }
                }
        }
        //total sink demand
        pot_grow_shoot = pot_grow_leaf + pot_grow_stem + pot_grow_fruit;
        return;
}

void lossrate5(void)
{
        float bbb;
        int i,j;
        //number of "aborted" fruits
        newabort_fruitno = 0.0;
        if (dwfruit >= eps)  {
                abortfruit_ratio = minimum(1.0, (0.67-max_abort*sourcesink_ratio));
                abortfruit_ratio = maximum(0.0, abortfruit_ratio);
                newabort_fruitno = abortfruit_ratio * newinit_fruitno;
        }
        abortfruit_no = abortfruit_no + newabort_fruitno;
        setfruit_no = fruitno - abortfruit_no;

        //location of "aborted" fruits: distal position
        bbb = 0.0;
        for(i=0;i<firstfruit_trussno;i++)  {
                //if no new frt or less than 2 frt on truss 1 or no mor abortion...
                if ((newfruit_on_truss[i] == 0.0) | (fruitno_on_truss[i] <=2.0) | (bbb >= newabort_fruitno))
                        continue;
                abortfruit_on_truss[i] = minimum(4.0, newfruit_on_truss[i]);
                abortfruit_on_truss[i] = minimum(abortfruit_on_truss[i], newabort_fruitno-bbb);
                abortfruit_on_truss[i] = minimum(abortfruit_on_truss[i], fruitno_on_truss[i] - 2);
                abortfruit_on_truss[i] = minimum(abortfruit_on_truss[i], (-1.0 * total_abortfruit[i]));
                abortfruit_on_truss[i] = maximum(0.0, newfruit_on_truss[i]);
                bbb = bbb + newfruit_on_truss[i];
                total_abortfruit[i] = total_abortfruit[i] + newfruit_on_truss[i];
                setfruit_on_truss[i] = (fruitno_on_truss[i] - total_abortfruit[i]);
                if (total_abortfruit[i]<1.0)  continue;
                for (j=setfruit_on_truss[i];j<(int)fruitno_on_truss[i];i++)          {
                        dwfruit_on_truss[i][j] = -1 * eps;
                }
        }
        return;
}

void calc_drymatter_growth_rate6(void)
{
```

```
int i,j;
int tmp;
dwshoot_growrate = 0.0;
dwleaf = 0.0;
dwleaf_modified = 0.0;
inst_dwleaf_on_truss = 0.0;
inst_dwfruit_on_truss = 0.0;
dwstem = 0.0;
dwfruit = 0.0;
fruitno_mature = 0.0;
dwfruit_mature = 0.0;
leafarea_plant = 0.0;
leafarea_modified = 0.0;

//leaf dry weight
for(i=0;i<no_truss;i++)          {
        dwshootgrow_truss = minimum(pot_leafgrow[i], pot_leafgrow[i] * sourcesink_ratio);
        dwshoot_growrate = dwshoot_growrate + dwshootgrow_truss;
        dwleaf_on_truss[i] = dwleaf_on_truss[i] + dwshootgrow_truss;
        //dry weight of growing leaves
        if (leafage_on_truss[i]<1) inst_dwleaf_on_truss = inst_dwleaf_on_truss + dwleaf_on_truss[i]
        //now leaf area expansion is either potential or limited by a max SLA
        //it is available for each unit
        leafarea_on_truss[i] = leafarea_on_truss[i] + minimum(pot_leafexp_on_truss[i],
                dwshootgrow_truss*max_sla/(1.0+petiole_leaf_ratio));
        dwleaf = dwleaf + dwleaf_on_truss[i];
        // total leaf area
        leafarea_plant = leafarea_plant + leafarea_on_truss[i];
        //total minus pruned leaves(leaves pruned when truss harvested)
        if (leafage_on_truss[i]>0.0) {
                leafarea_modified = leafarea_modified + leafarea_on_truss[i];
                dwleaf_modified = dwleaf_modified + dwleaf_on_truss[i];
        }
}
//stem dry weight
for(i=0;i<no_truss+1;i++)        {
        dwshootgrow_truss = minimum(pot_stemgrow[i], pot_stemgrow[i]*sourcesink_ratio);
        dwshoot_growrate = dwshoot_growrate + dwshootgrow_truss;
        dwstem_on_truss[i] = dwstem_on_truss[i] + dwshootgrow_truss;
        dwstem = dwstem + dwstem_on_truss[i];
}

//fruit dry weight
dwfruit = 0.0;
dwshootgrow_truss = 0.0;
for(i=0;i<no_truss;i++)          {
        dwfruits_on_truss[i] = 0.0;
        inst_fruitno_on_truss = (int) fruitno_on_truss[i];
        for(j=0;j<inst_fruitno_on_truss;j++)          {
                //for "aborted" fruit, DWF = -eps
                if ((fruitage_on_truss[i][j]<1.0) & (dwfruit_on_truss[i][j]>=0.0)) {
                        dwshootgrow_truss = minimum(pot_fruitgrow[i][j],
                                pot_fruitgrow[i][j] * sourcesink_ratio);
                        dwshoot_growrate = dwshoot_growrate + dwshootgrow_truss;
                        dwfruit_on_truss[i][j] += dwshootgrow_truss;
                }
                if (dwfruit_on_truss[i][j] > 0.0)
                        dwfruits_on_truss[i] += dwfruit_on_truss[i][j];
        //mature fruit
                if ((fruitage_on_truss[i][j] == 1.0) & (dwfruit_on_truss[i][j] >0.0))          {
                        dwfruit_mature = dwfruit_mature + dwfruit_on_truss[i][j];
                        fruitno_mature ++;
                }
        }

        //leaf pruning (when the truss is harvested)
        //the age of pruned leaves is set to -EPS to be identified
        tmp = (int) (pot_fruitno_on_truss[i]-1.0);
        if (fruitage_on_truss[i][tmp]>=1.0)          leafage_on_truss[i] = -1 * eps;
        dwfruit = dwfruit + dwfruits_on_truss[i];
}
```

```
        //carbohydrate pool
        carbon_pool = maximum(0.0, (biomass_product-dwshoot_growrate)/grow_efficiency);
        //when cpool is higher than a threshold value the gp is limited
        carbon_pool_max = 0.06*dwleaf/((1.0+petiole_leaf_ratio)*grow_efficiency);
        if (carbon_pool > carbon_pool_max)      {
                gross_photosynthesis = gross_photosynthesis - (carbon_pool - carbon_pool_max);
                carbon_pool = carbon_pool_max;
        }

        return;
}

void calc_transpiration(void)
{
        int maxit;
        int k;
        lai = leafarea_plant * planting_density;
        maxit = 30;

        for(k=0;k<maxit;k++){
                temp_veg = temp_air;
                tv = temp_air;
                dl = 0.06; //characteristic length of tomato vegetation in m
                vp_air = vapor_pressure(wetbulb_t);
                vp_sat = vapor_pressure(temp_air);
                vpd = (vp_sat - vp_air);
                leaf_vpd = vapor_pressure(temp_veg) - vp_air;
                if (leaf_vpd <= 0.0) leaf_vpd = 1.0;
                if ((temp_air>35.0)||(temp_air<-10.0)||(temp_veg>35.0)||(temp_veg<-10.0))
                        rlsi = 142.7 + 953.9 * exp(-0.0081 * global_solar);
                else
                        rlsi = 79.4 * (1.0 + pow((exp(0.0234*(global_solar - 76.6))), -1.0)) *
                                (1.0 + exp(0.079 * (leaf_vpd / 100.0 - 2.832)));
                ttc = temp_veg;
                tta = temp_air;

                calc_boundary_layer_resistance();
                calc_net_radiation();

                rahv = rah;
                transpiration =
                        ((delta*rnv/gamma)+2.0*lai*rhoa*cp*vpd/gamma/rahv)/
                        (0.93+delta/gamma+rlsi/rahv)/lambda;
                ltl = ta + ((0.93*rah+rlsi)*rn/2/lai/rhoa/cp - vpd/gamma) / (0.93+delta/gamma+rlsi/rah)
                delt = abs(ltl - temp_veg);
                if (delt<=0.00001)      {
                        transpiration = ((transpiration * 3600.0 * 24.0 * 1000.0) / planting_density);
                        return;
                }
                temp_veg = ltl;
        }
        transpiration = ((transpiration * 3600.0 * 24.0 * 1000.0) / planting_density);
//      if (transpiration < 0.0) transpiration = 0.0;
        return;
}

void calc_net_radiation(void)
{
        temp_greenhouse = temp_air;
        temp_mulching = temp_air;
        temp_a = temp_air;
        tair = temp_air;
        temp_cover = temp_air;
        radlong_cover = 0.0;
        radlong_screen = 0.0;
        radlong_veget = 0.0;
        radshort_veget = 0.0;
        radlong_greenhouse = 0.0;

        rlc = 0.0;
        rlv = 0.0;
        rls = 0.0;
```

```
rsv = 0.0;
rlg = 0.0;
tc = temp_air;
tair = temp_air;
tm = temp_air;
tg = temp_air;

epa = 0.9;
epg = 0.9;
reg = 0.1;
trc = 1.0;
rec = 0.0;
trlc = 0.0;
relc = 0.1;
epc = 0.9;
trm = 0.88;
rrem = 0.084;
trlm = 0.85;
relm = 0.12;
epm = 0.03;
trs = 1.0;
_res = 0.0;
epss = 0.0;
trls = 1.0;
rels = 0.0;
scp = 0.2;
epv = 0.95;

//short wave radiation
kdf = 0.58;
rev = (1.0 - sqrt(1.0-scp))/(1.0 + sqrt(1.0-scp));
trv = exp(-1.0 * kdf * lai);
abv = (1.0-revh)*(1.0-trvd);
trlv = exp(-0.8 * sqrt(1.0-0.1)*lai);
relv = (1.0-sqrt(1.0-0.1))/(1.0+sqrt(1.0-0.1));
plth = 0.495 + 0.439*lai;
bbx = 0.9/1.5;
aax = plth/1.5;
bbx = 1.35/2.05;
aax = plth/1.5;
denn = 1.0 - 0.8*pow(trlv,(1.18 * bbx)) - 0.2 * pow(trlv,(2.75 * bbx));
nonn = 1.0 - 0.2*pow(trlv,1.18) - 0.2 * pow(trlv,2.75);
ffww = (1.0 - 0.07 * (1.0 - bbx)*(4.7 - aax))*denn/nonn;
abv = abv * ffww;
trv = 1.0 - ffww + ffww * trv;
trlv = 1.0 - ffww + ffww * trlv;
rev = rev * ffww;
relv = relv * ffww;

resv = rev + rrem * pow(trv, 2.0) + reg * pow((trv*trm), 2.0);
resc = _res + recd * pow(trs, 2.0);
rsv = abv * global_solar / (1.0 - resv * resc);

qla = epa * sig * pow((tair + 273.16), 4.0);
qlc = epc * sig * pow((tc + 273.16), 4.0);
qls = epss * sig * pow((ts + 273.16), 4.0);          //ts = temp_air
qlv = epv * sig * pow((tv + 273.16), 4.0);
qlm = epm * sig * pow((tm + 273.16), 4.0);
qlg = epg * sig * pow((tg + 273.16), 4.0);

//long wave radiation
relv1 = rels + relc * pow(trls, 2.0);
relv2 = relv + pow(trlv, 2.0) * relm + relg * pow((trlv*trlm), 2.0);
rlv31 = (qla * trlc * trls + qlc * trls + qls) * (1.0-trlv)/(1.0-relv1*relv2);
relv3 = relv + pow(trlv, 2.0) * rels + relc * pow((trlv*trls), 2.0);
relv4 = relm + pow(trlm, 2.0) * relg;
rlv32 = (qlm + qlg * trm)*(1.0-trlv)/(1.0-relv3*relv4);
rlv33 = qlv * pow((1.0-trlv), 2.0) * (1.0-relv) * relv1 / (1.0 - relv1*relv2)
rlv34 = qlv * pow((1.0-trlv), 2.0) * (1.0-relv) * relv4 / (1.0 - relv3*relv4)
rlv35 = 2.0 * qlv * (1.0-trlv);
rlv = rlv31 + rlv32 + rlv33 + rlv34 - rlv35;
rnv1 = rsv + rlv;
```

```
        rnv = - 2.3896 + 0.74676 * rnvl;
//      rn = rnv;
//      printf("%f\n", rnv);
        return;
}

void calc_boundary_layer_resistance(void)
{
        float u;
        u = 0.2;
        xxp = (ttc + tta)/2.0,
        delta = vapor_pressure1(xxp);
        re = u * dl/nhu;
        nu_forced = 0 66 * (pow(re, 0.5))*(pow(0 7, 0 33)),
        grashof = aex * 9.8 * pow(dl, 3.0) * abs((ttc-tta))/nhu/nhu,    //Grashof number
        nust = 0.26 * pow(grashof*0.7, 0.25);
        nusb = 0.50 * pow(grashof, 0.25);
        nu_mixed = ((nust + nusb) / 2.0 + nu_forced);
        if ((ttc-tta)<=0)
                nu_mixed = (pow((pow(nu_forced, 3 55) + pow(nust, 3.55)), 0.28) +
                        pow((pow(nu_forced, 3.55) + pow(nusb, 3.55)), 0.28))/2.0;
        rah = dl / (kapa * nu_mixed);
        return;
}

void read_crop_parameter(void)
{
        extinction_coeff = 0.58;
        xm = 0.10;
        ck = -0.8;
        limit_vpd = 1 0;
        truss_per_leaf = 0.333,
        eps = 0.00000000001,
        grow_efficiency = 0.75;
        pot_fruitno_per_node = 0.8;
        max_abort = 0.73;
        q10 = 1.4;
        leaf_mresp_coeff = 0 015;
        fruit_mresp_coeff = 0.010,
        firsttruss_nodeno = 10.0;
        max_sla = 0 060;
        min_sla = 0.022;
        node_bet_firsttruss_fruitset = 9.0;
        petiole_leaf_ratio = 0.43,
        stem_leaf_ratio = 0 33,
}

void read_greenhouse_parameter(void)
{
//      NSTART = 3,
//      NDAYS = 200,
        delt = 1 0;
//      NFAST = 24.0;
//      INTOUT = 1.0;
//      TRGH = 0.65;
        planting_density = 8 0;
        init_nodeno = 23.0,             //18.0
        init_dwleaf = 37 049, //10 557;
        init_leafarea_plant = 0.518683;  //0 1478;
        init_dwstem = 17.0054;          //4.8457;
}

void clear_dayvalue(void)
{
        gross_photosynthesis = 0.0;
        tempeffect_on_devrate = 0.0;
        maint_resp= 0 0;
        daily_leafdev_rate = 0.0;
        fruitdev_rate = 0.0;
        daily_transpiration = 0.0;
        return;
}
```

```
void modify_data(void)
{
        int i;
        for(i=0;i<30;i++)       pot_fruitno_on_truss[i] = 4.0;
        for(i=1;i<8;i++)        maxrate_nodeinit[i] = 0.73;
        for (i=0;i<12;i++)      {
                pot_fruitgrow_in_age[i] = pot_fruitgrow_in_age[i] * 0.55
                pot_leafexp_in_age[i] = pot_leafexp_in_age[i] * 1.25;
        }
        for (i=0;i<9;i++)       {
                if (i==5 | i==6)        fruitdev_on_temp[i] = 0.032,
                fruitdev_on_temp[i] = fruitdev_on_temp[i] * 1.35,
                leafdev_on_temp[i] = leafdev_on_temp[i] * 1.0;
        }
        max_sla = max_sla * 0.65;
        min_sla = min_sla * 0.6;
        firsttruss_nodeno = 10.0;
        node_bet_firsttruss_fruitset = 4.0,
        max_abort = 0.55;
        planting_density = 8.0;
        init_nodeno =  25.670669556;
        init_dwleaf = 44.740379383;
        init_leafarea_plant = 0.614755273;
        init_dwstem = 20.535743713;
        petiole_leaf_ratio = 0.436;
        stem_leaf_ratio = 0.459;
        truss_per_leaf = 0.333;
        return,
}

void continue_simulation()
{
        no_iter = 24.0*6.0;
        dt_iter = 1.0/no_iter;
        read_crop_parameter();
        read_greenhouse_parameter(),
        modify_data();
        clear_dayvalue();
        restore_xmem();
        return,
}

void init_simulation()
{
        no_iter = 24.0*6.0;
        dt_iter = 1.0/no_iter;
        read_crop_parameter();
        read_greenhouse_parameter();
        modify_data();
        init_variable();
/*      leafage_on_truss[0] = 0.8454,
        leafage_on_truss[1] = 0.4334;
        leafage_on_truss[2] = 0.2334;
        leafage_on_truss[3] = 0.075;
*/
leafage_on_truss[3] = 0.156251639;
leafage_on_truss[2] = 0.314651698;
leafage_on_truss[1] = 0.514651716;
leafage_on_truss[0] = 0.926651478;


//      currentrecord = 0;
        clear_dayvalue();
//      irri_need0 = 0.0;
//      irri_need1 = 0.0,
//      irri_need2 = 0.0;
        return;
}

void run_daily_simulation()
{
```

```
        int date;
        date = currentday,
        development_rate6();
        ass_req_leaf = 1.39;
        ass_req_stem = 1.45;
        ass_req_fruit = 1.39;
        grow_efficiency = pot_grow_shoot / (ass_req_leaf * pot_grow_leaf +
                ass_req_stem * pot_grow_stem + ass_req_fruit * pot_grow_fruit);

        // daily production of biomass
        biomass_product = grow_efficiency * (gross_photosynthesis +
                carbon_pool - maint_resp);
        if (biomass_product<0)          {
                biomass_product = 0.0;
                maint_resp = gross_photosynthesis + carbon_pool;
        }

        //printf("biomass product : %10.6f %10.6f %10.6f %10.6f %10.6f \n",
        //biomass_product, grow_efficiency, gross_photosynthesis, carbon_pool, maint_resp)
        //calculate root growth

        biomass_product = biomass_product *
                (1.0-tables(proportion_root, x_nodeno, no_node, 6));

        //calculate source/sink ratio
        sourcesink_ratio = minimum(1.0, biomass_product/(pot_grow_shoot+eps));

        //daily growth respiration
        grow_resp = (gross_photosynthesis + carbon_pool - maint_resp) *
                (1.0 - grow_efficiency * 1.125);
        total_resp = maint_resp + grow_resp;      //daily total respiration
        lossrate5();                        //fruit setting
        calc_drymatter_growth_rate6(), //dry matter partitioning
        clear_dayvalue();

        return;
}

void run_fast_simulation()
{
//      temp_air = 35.0;
//      humidity = 30.0;
//      solarradiation = 300.0;
//      co2 = 1440.0;

        temp_air = (float) temperature_10min;
        wetbulb_t = (float) t_wetbulb_10min;
        humidity = (float) humidity_10min;
        solarradiation = (float) radiation_10min;
        co2 = (float) co2_10min;

        if (temp_air<5.0) temp_air = 5.0;
        if (temp_air>40.0) temp_air = 40.0;
        if (humidity >100.0) humidity = 100.0;
        if (humidity <20.0) humidity = 20.0;
        if (solarradiation < -10.0) solarradiation = -10.0;
        if (solarradiation > 1000.0) solarradiation = 1000.0;
        if (co2>1500.0) co2 = 1500.0;
        if (co2<50.0) co2 = 50.0;

        vp_air = vapor_pressure(wetbulb_t);
        vp_sat = vapor_pressure(temp_air);
        global_solar = solarradiation;

//      printf("start  ");
//      printf("%10.5f %10.5f %10.5f %10.5f %10.5f %10.5f\n",
//      inst_tempeffect_on_dev, tempeffect_on_devrate,
//      temp_air, humidity, solarradiation, co2);

        calc_temp_dep_factor(),
        calc_transpiration();
        calc_hourly_photosynthesis();
```

```
        calc_maint_respiration();

        tempeffect_on_devrate = tempeffect_on_devrate +
                inst_tempeffect_on_dev * dt_iter;
        maint_resp = maint_resp +
                inst_maint_resp * dt_iter;
        daily_leafdev_rate = daily_leafdev_rate +
                inst_leafdev_rate * dt_iter;
        fruitdev_rate = fruitdev_rate +
                inst_fruitdev_rate * dt_iter;
        gross_photosynthesis = gross_photosynthesis +
                inst_photosynthesis * dt_iter;
        daily_transpiration = daily_transpiration +
                transpiration * dt_iter;
        irri_need0 = irri_need0 + transpiration * dt_iter;
        irri_need1 = irri_need1 + transpiration * dt_iter;
        irri_need2 = irri_need2 + transpiration * dt_iter;
        currentrecord++;

        return;
}

void zero_variable()
{
//      co2_10min = 0;
//      humidity_10min = 0;
//      temperature_10min = 0;
//      t_wetbulb_10min = 0;
//      radiation_10min = 0;
        x_ageclass = 0.0;
        daily_leafdev_rate= 0.0;
        inst_fruitno_on_truss = 0.0;
        dwstem = 0.0;
        dwfruit = 0.0;
        inst_maint_resp = 0.0 ;
        inst_dwleaf_on_truss = 0.0;
        inst_dwfruit_on_truss = 0.0;
        init_dwleaf = 0.0;
        init_dwstem = 0.0;
        photon_density = 0.0;
        co2_compensation = 0.0;
        qefficiency = 0.0;
        inst_photosynthesis = 0.0;
        inst_tempeffect_on_dev = 0.0;
        inst_leafdev_rate = 0.0;
        inst_fruitdev_rate = 0.0;
        fruit_mresp_coeff= 0.0;
        leaf_mresp_coeff = 0.0;
        stem_mresp_coeff = 0.0;
        dwleaf = 0.0;
        extinction_coeff = 0.0;
        node_bet_firsttruss_fruitset = 0.0;
        pot_fruitno_per_node = 0.0;
        newinit_fruitno = 0.0;
        firstfruit_trussno = 0.0;
        fruitno = 0.0;
        nodeinit_rate = 0.0;
        no_node = 0.0;
        init_nodeno = 0.0;
        tempeffect_on_devrate = 0.0;
        no_truss = 0.0;
        no_leaf = 0.0;
        leafarea_plant = 0.0;
        newabort_fruitno = 0.0;
        abortfruit_ratio = 0.0;
        max_abort = 0.0;
        sourcesink_ratio = 0.0;
        abortfruit_no = 0.0;
        setfruit_no = 0.0;
        fruitdev_rate = 0.0;
        petiole_leaf_ratio = 0.0;
        stem_leaf_ratio = 0.0;
```

```
min_sla = 0.0;
firsttruss_nodeno = 0.0;
max_sla = 0.0,
dwshoot_growrate = 0.0;
dwshootgrow_truss = 0.0,
truss_per_leaf = 0.0;
fruitno_mature = 0.0;
dwfruit_mature = 0.0;
lai = 0.0;
rlsi = 0.0;
daily_transpiration = 0.0,
currentday = 0;
currentrecord = 0;
rlc = 0 0;
rls = 0 0;
rlg = 0.0;

grow_efficiency = 0 0;
grow_resp = 0 0,
total_resp = 0 0;
carbon_pool = 0.0,
maint_resp = 0.0,
gross_photosynthesis = 0.0,
biomass_product = 0.0;

ass_req_leaf = 0 0;
ass_req_stem = 0.0;
ass_req_fruit = 0.0;
pot_grow_shoot = 0.0;
pot_grow_leaf = 0.0;
pot_grow_stem = 0.0;
pot_grow_fruit = 0.0,
vpd = 0.0;
limit_vpd = 0.0;
leaf_vpd = 0.0;
ck = 0.0;
vp_air = 0.0;
vp_sat = 0.0;
temp_air = 0.0;
wetbulb_t = 0.0;
t_wetbulb = 0.0,
humidity = 0.0;
co2 = 0.0;
global_solar = 0.0;
solarradiation = 0 0;
tau1 = 0.0,
tau2 = 0 0;
pmax = 0 0;
top = 0.0;
bot = 0.0;
xm = 0 0;
eps = 0 0;
carbon_pool_max = 0.0;
temp_veg = 0.0;
dl = 0.0;
rahv = 0.0;
rah = 0.0;
transpiration = 0.0;
delta = 0.0;
rnv = 0.0;
ltl = 0.0;
ta = 0.0;
rn = 0.0;
delt = 0.0;
temp_greenhouse = 0.0;
temp_mulching = 0.0;
temp_a = 0.0;
temp_cover = 0 0;
radlong_cover = 0 0;
radlong_screen = 0.0;
radlong_veget = 0.0;
radlong_greenhouse = 0.0;
```

```
radshort_veget = 0.0;
epa = 0.0;
epg = 0.0;
reg = 0.0,
trc = 0.0,
rec = 0.0;
trlc = 0.0,
relc = 0.0;
epc = 0.0;
trm = 0.0;
rrem = 0.0;
trlm = 0 0;
relm = 0 0;
epm = 0.0;
trs = 0.0;
_res = 0.0;
epss = 0.0;
trls = 0.0;
.rels = 0.0;
scp = 0.0;
epv = 0.0;
kdf = 0.0;
rev = 0.0;
trv = 0.0;
abv = 0.0,
revh = 0.0;
trvd = 0.0;
trlv = 0 0;
relv = 0 0;
plth = 0 0;
bbx = 0.0;
aax = 0.0;
denn = 0.0;
nonn = 0 0;
ffww = 0.0;
resv = 0.0,
resc = 0.0;
recd = 0 0;
rsv = 0.0,
qla = 0 0;
tair = 0.0;
qlc = 0.0;
tc = 0 0;
qls = 0.0;
ts = 0 0;
qlv = 0.0;
tv = 0.0;
qlm = 0 0,
tm = 0 0;
qlg = 0 0,
tg = 0 0;
relv1 = 0.0;
relv2 = 0.0;
relg = 0 0;
rlv31 = 0.0,
relv3 = 0 0,
relv4 = 0.0;
rlv32 = 0 0;
rlv33 = 0 0;
rlv34 = 0.0;
rlv35 = 0.0;
rlv = 0 0;
rnv1 = 0.0;
xxp = 0.0,
ttc = 0 0;
tta = 0.0;
re = 0 0;
nu_forced = 0.0,
grashof = 0.0,
nust = 0 0;
nusm = 0 0;
nusb = 0 0.
```

```
        nu_mixed = 0.0;
        irri_need0 = 0.0;
        irri_need1 = 0.0;
        irri_need2 = 0.0;
        return;
}

/*************************************
 soil moisture sensor library
*************************************/

void mux16_ch_n(int n, int ch)
{
        int s;
        s = inport(PPIA_E);
        RES(&s, 4),             // chan 0
        RES(&s, 5);
        RES(&s, 6);
        RES(&s, 7),
        if (BIT(&ch, 0))        SET(&s, 4),
        if (BIT(&ch, 1))        SET(&s, 5);
        if (BIT(&ch, 2))        SET(&s, 6);
        if (BIT(&ch, 3))        SET(&s, 7);
        outport(PPIA_E, s),
        return;
}

void high_pulse()
{
        int s,
        s = inport(PPIB);
        SET(&s, 0);
        SET(&s, 1);
        outport(PPIB, s);
        return;
}

void low_pulse()
{
        int s;
        s = inport(PPIB),
        RES(&s, 0);
        RES(&s, 1);
        outport(PPIB, s);
        return;
}


void store_xmem()
{
        /* model variables */
        xputfloat(xm_rad, radiation_sum);
        xputint(xm_reset_model, reset_model);
        xputfloat(xm_temperature_10min, temperature_10min),
        xputfloat(xm_t_wetbulb_10min, t_wetbulb_10min);
        xputfloat(xm_humidity_10min , humidity_10min);
        xputfloat(xm_rad_10min, radiation_10min);
        xputfloat(xm_co2_10min, co2_10mun),
        xputfloat(xm_irri_need0, irri_need0),
        xputfloat(xm_irri_need1, irri_need1),
        xputfloat(xm_irri_need2, irri_need2);

        root2xmem(xfpn, xm_xfpn, 40);
        root2xmem(leafarea_on_truss, xm_leafarea_on_truss, 120);        //xla
        root2xmem(dwleaf_on_truss, xm_dwleaf_on_truss, 120);            //dwl
        root2xmem(dwstem_on_truss, xm_dwstem_on_truss, 120);            //dws
        root2xmem(leafage_on_truss, xm_leafage_on_truss, 120);          //agls
        root2xmem(dwfruit_on_truss, xm_dwfruit_on_truss, 840);          //dwf
        root2xmem(fruitage_on_truss, xm_fruitage_on_truss, 840);        //agf
        root2xmem(dwfruits_on_truss, xm_dwfruits_on_truss, 120);        //dwtr
        root2xmem(pot_stemgrow, xm_pot_stemgrow, 120);                  //pgs
        root2xmem(pot_leafgrow, xm_pot_leafgrow, 120);                  //pgl
```

```
root2xmem(pot_fruitgrow, xm_pot_fruitgrow, 1800);                    //pgf
root2xmem(abortfruit_on_truss, xm_abortfruit_on_truss, 120);         //abnf
root2xmem(newfruit_on_truss, xm_newfruit_on_truss, 120);             //rcnf
root2xmem(fruitno_on_truss, xm_fruitno_on_truss, 120);               //xnft
root2xmem(x_no_on_truss, xm_x_no_on_truss, 120);                     //xnsft
root2xmem(total_abortfruit, xm_total_abortfruit, 120);               //abor
root2xmem(pot_leafexp_on_truss, xm_pot_leafexp_on_truss, 120),       //ple
root2xmem(setfruit_on_truss, xm_setfruit_on_truss, 120);             //nsf

xputfloat(xm_inst_fruitno_on_truss, inst_fruitno_on_truss);          //nft
xputint(xm_no_iter, no_iter);
xputfloat(xm_dt_iter, dt_iter),
xputfloat(xm_x_ageclass, x_ageclass);                                //xbox
xputfloat(xm_daily_leafdev_rate, daily_leafdev_rate);                //rdvlv
xputfloat(xm_dwstem, dwstem);                                        //tdms
xputfloat(xm_dwfruit, dwfruit);                                      //tdmf
xputfloat(xm_inst_maint_resp, inst_maint_resp);                      //rmaintf
xputfloat(xm_inst_dwleaf_on_truss, inst_dwleaf_on_truss);            //dmgl
xputfloat(xm_inst_dwfruit_on_truss, inst_dwfruit_on_truss);          //dmgf
xputfloat(xm_init_dwleaf, init_dwleaf);                              //wlvsi
xputfloat(xm_init_dwstem, init_dwstem);                              //wstmi
xputfloat(xm_photon_density, photon_density);                        //ppfd
xputfloat(xm_co2_compensation, co2_compensation),                    //cd
xputfloat(xm_qefficiency, qefficiency),                              //qe quantum use efficiency
xputfloat(xm_inst_photosynthesis, inst_photosynthesis);              //gpf
xputfloat(xm_inst_tempeffect_on_dev, inst_tempeffect_on_dev),        //temfcf
xputfloat(xm_inst_leafdev_rate, inst_leafdev_rate);                  //rdvlvf
xputfloat(xm_inst_fruitdev_rate, inst_fruitdev_rate);                //rdvfrf
xputfloat(xm_q10, q10);                                              //q10
xputfloat(xm_fruit_mresp_coeff, fruit_mresp_coeff);                  //rmrf
xputfloat(xm_leaf_mresp_coeff, leaf_mresp_coeff),                    //rmrl
xputfloat(xm_stem_mresp_coeff, stem_mresp_coeff),                    //rmrs
xputfloat(xm_dwleaf, dwleaf);                                        //tdml
xputfloat(xm_dwleaf_modified, dwleaf_modified);                      //tdml2
xputfloat(xm_leafarea_modified, leafarea_modified);                  //plar2
xputfloat(xm_init_leafarea_plant, init_leafarea_plant);              //plari
xputfloat(xm_planting_density, planting_density);                    //plm2
xputfloat(xm_extinction_coeff, extinction_coeff);                    //xk
xputfloat(xm_node_bet_firsttruss_fruitset, node_bet_firsttruss_fruitset);  //frlg
xputfloat(xm_pot_fruitno_per_node, pot_fruitno_per_node);            //fpnpt
xputfloat(xm_newinit_fruitno, newinit_fruitno);                      //trcnf
xputfloat(xm_firstfruit_trussno, firstfruit_trussno);                //nbrup
xputfloat(xm_fruitno, fruitno);                                      //tnf
xputfloat(xm_nodeinit_rate, nodeinit_rate);                          //genr
xputfloat(xm_no_node, no_node);                                      //plstn
xputfloat(xm_init_nodeno, init_nodeno);                              //plstni
xputfloat(xm_tempeffect_on_devrate, tempeffect_on_devrate);          //temfac
xputfloat(xm_no_truss, no_truss);                                    //nbru
xputfloat(xm_no_leaf, no_leaf),                                      //nblv
xputfloat(xm_leafarea_plant, leafarea_plant);                        //plar
xputfloat(xm_newabort_fruitno, newabort_fruitno);                    //tabnf
xputfloat(xm_abortfruit_ratio, abortfruit_ratio);                    //fabor
xputfloat(xm_max_abort, max_abort);                                  //abormx
xputfloat(xm_sourcesink_ratio, sourcesink_ratio),                    //sosir
xputfloat(xm_abortfruit_no, abortfruit_no),                          //tabf
xputfloat(xm_setfruit_no, setfruit_no);                              //tnsf
xputfloat(xm_fruitdev_rate, fruitdev_rate);                          //rdvfr
xputfloat(xm_petiole_leaf_ratio, petiole_leaf_ratio);                //frpt
xputfloat(xm_stem_leaf_ratio, stem_leaf_ratio);                      //frst
xputfloat(xm_min_sla, min_sla),                                      //slamn
xputfloat(xm_firsttruss_nodeno, firsttruss_nodeno);                  //ftrusn
xputfloat(xm_max_sla, max_sla);                                      //slamx
xputfloat(xm_dwshoot_growrate, dwshoot_growrate);                    //ascsp
xputfloat(xm_dwshootgrow_truss, dwshootgrow_truss),                  //xasc
xputfloat(xm_truss_per_leaf, truss_per_leaf),                        //tpl
xputfloat(xm_fruitno_mature, fruitno_mature);                        //tnmf
xputfloat(xm_dwfruit_mature, dwfruit_mature);                        //dmmf
xputfloat(xm_lai, lai);
xputfloat(xm_rlsi, rlsi);
xputfloat(xm_daily_transpiration, daily_transpiration),              //dtran
xputint(xm_currentday, currentday);
xputint(xm_currentrecord, currentrecord);
```

```
xputfloat(xm_rlc, rlc);
xputfloat(xm_rls, rls);
xputfloat(xm_rlg, rlg);

xputfloat(xm_grow_efficiency, grow_efficiency);                    //gref
xputfloat(xm_grow_resp, grow_resp);                                //gresp
xputfloat(xm_total_resp, total_resp);                              //tresp
xputfloat(xm_carbon_pool, carbon_pool);                            //cpool
xputfloat(xm_maint_resp, maint_resp);                              //rmaint
xputfloat(xm_gross_photosynthesis, gross_photosynthesis);          //gp
xputfloat(xm_biomass_product, biomass_product);                    //rcdrw

xputfloat(xm_ass_req_leaf, ass_req_leaf);                          //asrl
xputfloat(xm_ass_req_stem, ass_req_stem);                          //asrs
xputfloat(xm_ass_req_fruit, ass_req_fruit);                        //asrf
xputfloat(xm_pot_grow_shoot, pot_grow_shoot);                      //pngp
xputfloat(xm_pot_grow_leaf, pot_grow_leaf);                        //ptnlvs
xputfloat(xm_pot_grow_stem, pot_grow_stem);                        //ptnstm
xputfloat(xm_pot_grow_fruit, pot_grow_fruit);                      //ptnfrt
xputfloat(xm_vpd, vpd);
xputfloat(xm_limit_vpd, limit_vpd);                //limit VPD for VPD vs Pmax curve
xputfloat(xm_leaf_vpd, leaf_vpd);
xputfloat(xm_ck, ck);
xputfloat(xm_vp_air, vp_air);
xputfloat(xm_vp_sat, vp_sat);
xputfloat(xm_temp_air, temp_air);
xputfloat(xm_humidity, humidity);
xputfloat(xm_co2, co2);
xputfloat(xm_global_solar, global_solar);                          //gsol
xputfloat(xm_solarradiation, solarradiation);
xputfloat(xm_tau1, tau1);
xputfloat(xm_tau2, tau2);
xputfloat(xm_pmax, pmax);
xputfloat(xm_top, top);
xputfloat(xm_bot, bot);
xputfloat(xm_xm, xm);
xputfloat(xm_eps, eps);
xputfloat(xm_carbon_pool_max, carbon_pool_max);                    //cpoolmx
xputfloat(xm_temp_veg, temp_veg);
xputfloat(xm_dl, dl);
xputfloat(xm_rahv, rahv);
xputfloat(xm_rah, rah);
xputfloat(xm_transpiration, transpiration);
xputfloat(xm_delta, delta);
xputfloat(xm_rnv, rnv);
xputfloat(xm_ltl, ltl);
xputfloat(xm_ta, ta);
xputfloat(xm_rn, rn);
xputfloat(xm_delt, delt);
xputfloat(xm_temp_greenhouse, temp_greenhouse);
xputfloat(xm_temp_mulching, temp_mulching);
xputfloat(xm_temp_a, temp_a);
xputfloat(xm_temp_cover, temp_cover);
xputfloat(xm_radlong_cover, radlong_cover);
xputfloat(xm_radlong_screen, radlong_screen);
xputfloat(xm_radlong_veget, radlong_veget);
xputfloat(xm_radlong_greenhouse, radlong_greenhouse);
xputfloat(xm_radshort_veget, radshort_veget);
xputfloat(xm_epa, epa);
xputfloat(xm_epg, epg);
xputfloat(xm_reg, reg);
xputfloat(xm_trc, trc);
xputfloat(xm_rec, rec);
xputfloat(xm_trlc, trlc);
xputfloat(xm_relc, relc);
xputfloat(xm_epc, epc);
xputfloat(xm_trm, trm);
xputfloat(xm_rrem, rrem);
xputfloat(xm_trlm, trlm);
xputfloat(xm_relm, relm);
xputfloat(xm_epm, epm);
xputfloat(xm_trs, trs);
```

```
        xputfloat(xm__res, _res);
        xputfloat(xm_epss, epss);
        xputfloat(xm_trls, trls);
        xputfloat(xm_rels, rels);
        xputfloat(xm_scp, scp);
        xputfloat(xm_epv, epv);
        xputfloat(xm_kdf, kdf);
        xputfloat(xm_rev, rev);
        xputfloat(xm_trv, trv);
        xputfloat(xm_abv, abv);
        xputfloat(xm_revh, revh);
        xputfloat(xm_trvd, trvd);
        xputfloat(xm_trlv, trlv);
        xputfloat(xm_relv, relv);
        xputfloat(xm_plth, plth);
        xputfloat(xm_bbx, bbx);
        xputfloat(xm_aax, aax);
        xputfloat(xm_denn, denn);
        xputfloat(xm_nonn, nonn);
        xputfloat(xm_ffww, ffww);
        xputfloat(xm_resv, resv);
        xputfloat(xm_resc, resc);
        xputfloat(xm_recd, recd);
        xputfloat(xm_rsv, rsv);
        xputfloat(xm_qla, qla);
        xputfloat(xm_tair, tair);
        xputfloat(xm_qlc, qlc);
        xputfloat(xm_tc, tc);
        xputfloat(xm_qls, qls);
        xputfloat(xm_ts, ts);
        xputfloat(xm_qlv, qlv);
        xputfloat(xm_tv, tv);
        xputfloat(xm_qlm, qlm);
        xputfloat(xm_tm, tm);
        xputfloat(xm_qlg, qlg);
        xputfloat(xm_tg, tg);
        xputfloat(xm_relv1, relv1);
        xputfloat(xm_relv2, relv2);
        xputfloat(xm_relg, relg);
        xputfloat(xm_rlv31, rlv31);
        xputfloat(xm_relv3, relv3);
        xputfloat(xm_relv4, relv4);
        xputfloat(xm_rlv32, rlv32);
        xputfloat(xm_rlv33, rlv33);
        xputfloat(xm_rlv34, rlv34);
        xputfloat(xm_rlv35, rlv35);
        xputfloat(xm_rlv, rlv);
        xputfloat(xm_rnv1, rnv1);
        xputfloat(xm_xxp, xxp);
        xputfloat(xm_ttc, ttc);
        xputfloat(xm_tta, tta);
        xputfloat(xm_re, re);
        xputfloat(xm_nu_forced, nu_forced);
        xputfloat(xm_grashof, grashof);
        xputfloat(xm_nust, nust);
        xputfloat(xm_nusm, nusm);
        xputfloat(xm_nusb, nusb);
        xputfloat(xm_nu_mixed, nu_mixed);
        return;
}

void restore_xmem()
{
        /* model variables */
        radiation_sum = xgetfloat(xm_rad);
        reset_model = xgetint(xm_reset_model);
        temperature_10min = xgetfloat(xm_temperature_10min)
        t_wetbulb_10min = xgetfloat(xm_t_wetbulb_10min);
        humidity_10min = xgetfloat(xm_humidity_10min);
        radiation_10min = xgetfloat(xm_rad_10min);
        co2_10min = xgetfloat(xm_co2_10min);
        irri_need0 = xgetfloat(xm_irri_need0);
```

```
irri_need1 = xgetfloat(xm_irri_need1);
irri_need2 = xgetfloat(xm_irri_need2);

xmem2root(xm_xfpn, xfpn, 40);
xmem2root(xm_leafarea_on_truss, leafarea_on_truss, 120);          //xla
xmem2root(xm_dwleaf_on_truss, dwleaf_on_truss, 120);              //dwl
xmem2root(xm_dwstem_on_truss, dwstem_on_truss, 120);              //dws
xmem2root(xm_leafage_on_truss, leafage_on_truss, 120);            //agls
xmem2root(xm_dwfruit_on_truss, dwfruit_on_truss, 840);            //dwf
xmem2root(xm_fruitage_on_truss, fruitage_on_truss, 840);          //agf
xmem2root(xm_dwfruits_on_truss, dwfruits_on_truss, 120);          //dwtr
xmem2root(xm_pot_stemgrow, pot_stemgrow, 120);                    //pgs
xmem2root(xm_pot_leafgrow, pot_leafgrow, 120);                    //pgl
xmem2root(xm_pot_fruitgrow, pot_fruitgrow, 1800);                 //pgf
xmem2root(xm_abortfruit_on_truss, abortfruit_on_truss, 120);      //abnf
xmem2root(xm_newfruit_on_truss, newfruit_on_truss, 120);          //rcnf
xmem2root(xm_fruitno_on_truss, fruitno_on_truss, 120);            //xnft
xmem2root(xm_x_no_on_truss, x_no_on_truss, 120);                  //xnsft
xmem2root(xm_total_abortfruit, total_abortfruit, 120);            //abor
xmem2root(xm_pot_leafexp_on_truss, pot_leafexp_on_truss, 120);    //ple
xmem2root(xm_setfruit_on_truss, setfruit_on_truss, 120);          //nsf

inst_fruitno_on_truss = xgetfloat(xm_inst_fruitno_on_truss);      //nft
no_iter = xgetint(xm_no_iter);
dt_iter = xgetfloat(xm_dt_iter);
x_ageclass = xgetfloat(xm_x_ageclass);                            //xbox
daily_leafdev_rate = xgetfloat(xm_daily_leafdev_rate);            //rdvlv
dwstem = xgetfloat(xm_dwstem);                                    //tdms
dwfruit = xgetfloat(xm_dwfruit);                                  //tdmf
inst_maint_resp = xgetfloat(xm_inst_maint_resp);                  //rmaintf
inst_dwleaf_on_truss = xgetfloat(xm_inst_dwleaf_on_truss);        //dmgl
inst_dwfruit_on_truss = xgetfloat(xm_inst_dwfruit_on_truss);      //dmgf
init_dwleaf = xgetfloat(xm_init_dwleaf);                          //wlvsi
init_dwstem = xgetfloat(xm_init_dwstem);                          //wstmi
photon_density = xgetfloat(xm_photon_density);                    //ppfd
co2_compensation = xgetfloat(xm_co2_compensation);                //cd
qefficiency = xgetfloat(xm_qefficiency);                          //qe quantum use efficiency
inst_photosynthesis = xgetfloat(xm_inst_photosynthesis);          //gpf
inst_tempeffect_on_dev = xgetfloat(xm_inst_tempeffect_on_dev);    //temfcf
inst_leafdev_rate = xgetfloat(xm_inst_leafdev_rate);              //rdvlvf
inst_fruitdev_rate = xgetfloat(xm_inst_fruitdev_rate);            //rdvfrf
q10 = xgetfloat(xm_q10);                                          //q10
fruit_mresp_coeff = xgetfloat(xm_fruit_mresp_coeff);              //rmrf
leaf_mresp_coeff = xgetfloat(xm_leaf_mresp_coeff);                //rmrl
stem_mresp_coeff = xgetfloat(xm_stem_mresp_coeff);                //rmrs
dwleaf = xgetfloat(xm_dwleaf);                                    //tdml
dwleaf_modified = xgetfloat(xm_dwleaf_modified);                  //tdml2
leafarea_modified = xgetfloat(xm_leafarea_modified);              //plar2
init_leafarea_plant = xgetfloat(xm_init_leafarea_plant);          //plari
planting_density = xgetfloat(xm_planting_density);                //plm2
extinction_coeff = xgetfloat(xm_extinction_coeff);                //xk
node_bet_firsttruss_fruitset = xgetfloat(xm_node_bet_firsttruss_fruitset);  //frlg
pot_fruitno_per_node = xgetfloat(xm_pot_fruitno_per_node);        //fpnpt
newinit_fruitno = xgetfloat(xm_newinit_fruitno);                  //trcnf
firstfruit_trussno = xgetfloat(xm_firstfruit_trussno);            //nbrup
fruitno = xgetfloat(xm_fruitno);                                  //tnf
nodeinit_rate = xgetfloat(xm_nodeinit_rate);                      //genr
no_node = xgetfloat(xm_no_node);                                  //plstn
init_nodeno = xgetfloat(xm_init_nodeno);                          //plstni
tempeffect_on_devrate = xgetfloat(xm_tempeffect_on_devrate);      //temfac
no_truss = xgetfloat(xm_no_truss);                                //nbru
no_leaf = xgetfloat(xm_no_leaf);                                  //nblv
leafarea_plant = xgetfloat(xm_leafarea_plant);                    //plar
newabort_fruitno = xgetfloat(xm_newabort_fruitno);                //tabnf
abortfruit_ratio = xgetfloat(xm_abortfruit_ratio);                //fabor
max_abort = xgetfloat(xm_max_abort);                              //abormx
sourcesink_ratio = xgetfloat(xm_sourcesink_ratio);                //sosir
abortfruit_no = xgetfloat(xm_abortfruit_no);                      //tabf
setfruit_no = xgetfloat(xm_setfruit_no);                          //tnsf
fruitdev_rate = xgetfloat(xm_fruitdev_rate);                      //rdvfr
petiole_leaf_ratio = xgetfloat(xm_petiole_leaf_ratio);            //frpt
stem_leaf_ratio = xgetfloat(xm_stem_leaf_ratio);                  //frst
```

```
min_sla = xgetfloat(xm_min_sla);                          //slamn
firsttruss_nodeno = xgetfloat(xm_firsttruss_nodeno);      //ftrusn
max_sla = xgetfloat(xm_max_sla);                          //slamx
dwshoot_growrate = xgetfloat(xm_dwshoot_growrate);        //ascsp
dwshootgrow_truss = xgetfloat(xm_dwshootgrow_truss),      //xasc
truss_per_leaf = xgetfloat(xm_truss_per_leaf);            //tpl
fruitno_mature = xgetfloat(xm_fruitno_mature);            //tnmf
dwfruit_mature = xgetfloat(xm_dwfruit_mature);            //dmmf
lai = xgetfloat(xm_lai);
rlsi = xgetfloat(xm_rlsi);
daily_transpiration = xgetfloat(xm_daily_transpiration);  //dtran
currentday = xgetint(xm_currentday);
currentrecord = xgetint(xm_currentrecord);
rlc = xgetfloat(xm_rlc),
rls = xgetfloat(xm_rls),
rlg = xgetfloat(xm_rlg);

grow_efficiency = xgetfloat(xm_grow_efficiency);          //gref
grow_resp = xgetfloat(xm_grow_resp);                      //gresp
total_resp = xgetfloat(xm_total_resp);                    //tresp
carbon_pool = xgetfloat(xm_carbon_pool);                  //cpool
maint_resp = xgetfloat(xm_maint_resp),                    //rmaint
gross_photosynthesis = xgetfloat(xm_gross_photosynthesis); //gp
biomass_product = xgetfloat(xm_biomass_product);          //rcdrw

ass_req_leaf = xgetfloat(xm_ass_req_leaf),                //asrl
ass_req_stem = xgetfloat(xm_ass_req_stem);                //asrs
ass_req_fruit = xgetfloat(xm_ass_req_fruit);              //asrf
pot_grow_shoot = xgetfloat(xm_pot_grow_shoot);            //pngp
pot_grow_leaf = xgetfloat(xm_pot_grow_leaf),              //ptnlvs
pot_grow_stem = xgetfloat(xm_pot_grow_stem);              //ptnstm
pot_grow_fruit = xgetfloat(xm_pot_grow_fruit);            //ptnfrt
vpd = xgetfloat(xm_vpd);
limit_vpd = xgetfloat(xm_limit_vpd);      //limit VPD for VPD vs Pmax curve
leaf_vpd = xgetfloat(xm_leaf_vpd);
ck = xgetfloat(xm_ck);
vp_air = xgetfloat(xm_vp_air);
vp_sat = xgetfloat(xm_vp_sat);
temp_air = xgetfloat(xm_temp_air),
wetbulb_t = xgetfloat(xm_wetbulb_t);
humidity = xgetfloat(xm_humidity);
co2 = xgetfloat(xm_co2);
global_solar = xgetfloat(xm_global_solar),                //gsol
solarradiation = xgetfloat(xm_solarradiation),
tau1 = xgetfloat(xm_tau1),
tau2 = xgetfloat(xm_tau2),
pmax = xgetfloat(xm_pmax);
top = xgetfloat(xm_top);
bot = xgetfloat(xm_bot);
xm = xgetfloat(xm_xm);
eps = xgetfloat(xm_eps);
carbon_pool_max = xgetfloat(xm_carbon_pool_max);          //cpoolmx
temp_veg = xgetfloat(xm_temp_veg);
dl = xgetfloat(xm_dl);
rahv = xgetfloat(xm_rahv);
rah = xgetfloat(xm_rah);
transpiration = xgetfloat(xm_transpiration);
delta = xgetfloat(xm_delta);
rnv = xgetfloat(xm_rnv);
ltl = xgetfloat(xm_ltl);
ta = xgetfloat(xm_ta),
```

```
rn = xgetfloat(xm_rn);
delt = xgetfloat(xm_delt);
temp_greenhouse = xgetfloat(xm_temp_greenhouse);
temp_mulching = xgetfloat(xm_temp_mulching);
temp_a = xgetfloat(xm_temp_a);
temp_cover = xgetfloat(xm_temp_cover);
radlong_cover = xgetfloat(xm_radlong_cover);
radlong_screen = xgetfloat(xm_radlong_screen);
radlong_veget = xgetfloat(xm_radlong_veget);
radlong_greenhouse = xgetfloat(xm_radlong_greenhouse)
radshort_veget = xgetfloat(xm_radshort_veget);
epa = xgetfloat(xm_epa);
epg = xgetfloat(xm_epg);
reg = xgetfloat(xm_reg);
trc = xgetfloat(xm_trc);
rec = xgetfloat(xm_rec),
trlc = xgetfloat(xm_trlc),
relc = xgetfloat(xm_relc);
epc = xgetfloat(xm_epc);
trm = xgetfloat(xm_trm);
rrem = xgetfloat(xm_rrem);
trlm = xgetfloat(xm_trlm);
relm = xgetfloat(xm_relm);
epm = xgetfloat(xm_epm);
trs = xgetfloat(xm_trs),
_res = xgetfloat(xm__res);
epss = xgetfloat(xm_epss),
trls = xgetfloat(xm_trls);
rels = xgetfloat(xm_rels);
scp = xgetfloat(xm_scp);
epv = xgetfloat(xm_epv);
kdf = xgetfloat(xm_kdf),
rev = xgetfloat(xm_rev);
trv = xgetfloat(xm_trv);
abv = xgetfloat(xm_abv);
revh = xgetfloat(xm_revh);
trvd = xgetfloat(xm_trvd);
trlv = xgetfloat(xm_trlv),
relv = xgetfloat(xm_relv);
plth = xgetfloat(xm_plth);
bbx = xgetfloat(xm_bbx);
aax = xgetfloat(xm_aax);
denn = xgetfloat(xm_denn);
nonn = xgetfloat(xm_nonn);
ffww = xgetfloat(xm_ffww);
resv = xgetfloat(xm_resv),
resc = xgetfloat(xm_resc);
recd = xgetfloat(xm_recd);
rsv = xgetfloat(xm_rsv);
qla = xgetfloat(xm_qla);
tair = xgetfloat(xm_tair);
qlc = xgetfloat(xm_qlc),
tc = xgetfloat(xm_tc);
qls = xgetfloat(xm_qls);
ts = xgetfloat(xm_ts);
qlv = xgetfloat(xm_qlv);
tv = xgetfloat(xm_tv);
qlm = xgetfloat(xm_qlm);
```

```
        tm    xgetfloat(xm_tm);
        qlg = xgetfloat(xm_qlg);
        tg = xgetfloat(xm_tg);
        relv1 = xgetfloat(xm_relv1);
        relv2 = xgetfloat(xm_relv2);
        relg = xgetfloat(xm_relg);
        rlv31 = xgetfloat(xm_rlv31),
        relv3 = xgetfloat(xm_relv3);
        relv4 = xgetfloat(xm_relv4);
        rlv32 = xgetfloat(xm_rlv32);
        rlv33 = xgetfloat(xm_rlv33),
        rlv34 = xgetfloat(xm_rlv34),
        rlv35 = xgetfloat(xm_rlv35);
        rlv = xgetfloat(xm_rlv);
        rnv1 = xgetfloat(xm_rnv1);
        xxp = xgetfloat(xm_xxp);
        ttc = xgetfloat(xm_ttc);
        tta = xgetfloat(xm_tta),
        re = xgetfloat(xm_re);
        nu_forced = xgetfloat(xm_nu_forced),
        grashof = xgetfloat(xm_grashof),
        nust = xgetfloat(xm_nust);
        nusm = xgetfloat(xm_nusm),
        nusb = xgetfloat(xm_nusb);
        nu_mixed = xgetfloat(xm_nu_mixed),
        return,
}
```