ANL/ENG-77-04

CAL-ERDA PROGRAM MANUAL

Gloria A. Bennett, Stephen C. Diamond, Bruce D. Hunn, Mark A. Roschke, Eva F. Tucker

Los Alamos Scientific Laboratory
Los Alamos, New Mexico

Lawrence Berkeley Laboratory

Argonne National Laboratory

Los Alamos Scientific Laboratory

In Cooperation with
Consultants Computation Bureau
and Supported by the
State of California, Energy Resources Conservation and Development Commission
and the United States Energy Research and Development Administration

CAL-ERDA PROGRAM MANUAL

by

B. D. Hunn
S. C. Diamond
G. A. Bennett
E. F. Tucker
M. A. Roschke

Group WX-4, Program Support
Los Alamos Scientific Laboratory
Los Alamos, NM 87545

This document describes the program version as of the following dates:

Subprogram	Cal-ERDA Version	Dated
LOADS	1.2	July 11, 1977 (except as otherwise noted)
SYSTEMS	-	June 15, 1977 (except as otherwise noted)
PLANT	1.1	May 5, 1977
ECONOMICS	1.1	May 5, 1977

As new versions of Cal-ERDA are released, the Program Manual will be updated to reflect changes soon after their release.

October 1977

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

CAL-ERDA PROGRAM MANUAL

TABLE OF CONTENTS

Page

Page

ABSTRACT

A set of computer programs, called Cal-ERDA, is described that is capable of rapid and detailed analysis of energy consumption in buildings. A new user-oriented input language, named the Building Design Language (BDL), has been written to allow simplified manipulation of the many variables used to describe a building and its operation.

This Program Manual provides the user with information necessary to understand in detail the Cal-ERDA set of computer programs. It contains a summary of the equations and algorithms used to perform the calculations, as well as flow charts to describe how each program operates. The relationship of the Cal-ERDA algorithms to ASHRAE algorithms is also given and traced to ASHRAE documentation. The operation, structure, and logical sequence of calculations is described in a step-by-step manner for each of the major computational programs.

The new computer programs described in this manual include:

1. An EXECUTIVE Processor to create computer system control commands.
2. A BDL Processor to analyze the input instructions, execute computer system control commands, perform assignments and data retrieval, and control the operation of the LOADS, SYSTEMS, PLANT, ECONOMICS, and REPORT programs.
3. A LOADS analysis program that calculates peak (design) zone and hourly loads and the effect of the ambient weather conditions, the internal occupancy, lighting, and equipment within the building, as well as variations in the size, location, orientation, construction, walls, roofs, floors, fenestrations, attachments (awnings, balconies), and shape of a building.
4. A Heating, Ventilating, and Air-Conditioning (HVAC) SYSTEMS analysis program capable of modeling the operation of HVAC components including fans, coils, economizers, humidifiers, etc., arranged in 1 of 16 standard configurations and operated according to various temperature and humidity control schedules.
5. A PLANT equipment program that models the operation of boilers, chillers, electrical generation equipment (diesel or turbines), heat storage apparatus (chilled or heated water), and solar heating and/or cooling systems.

6. An ECONOMIC analysis program that calculates life-cycle costs.
7. A REPORT program that produces tables of user-selected variables and arranges them according to user-specified formats.
8. A set of WEATHER ANALYSIS programs capable of manipulating, summarizing, and plotting weather data.

A library of weather data has been prepared that includes temperature, wind, and cloud data for 60 locations in the United States. These data are used to calculate the response of a building for each hour of a year (8760 hr/yr). A library of schedule data has also been prepared that allows hourly, daily, weekly, monthly, seasonal, and yearly specifications of the building operation. These schedules are used to specify desired temperature variations, occupancy patterns, lighting schedules, and equipment operation schedules. Another library contains data on the thermal properties of walls, roofs, floors, windows, doors, and attachments. The user is allowed to create new library entries using his own data and/or to select and assemble data for each specific job.

ACKNOWLEDGMENTS

This Program Manual was prepared by Group WX-4, Program Support, Design Engineering Division, Los Alamos Scientific Laboratory (LASL). It was a joint effort of several people who have prepared the following chapters:

Overall Editing:	Bruce D. Hunn and Eva F. Tucker
LOADS Program:	Stephen C. Diamond
SYSTEMS Program:	Gloria A. Bennett
PLANT Program:	Bruce D. Hunn
ECONOMICS Program:	Eva F. Tucker
Solar Simulator:	Mark A. Roschke and Stephen C. Diamond

Many of the descriptions and flow charts in the subroutine write-ups in PLANT (Chap. V) were obtained from the SIMTEP (Central Plant Simulation) Chapter of the "Program Reference Manual for Building Loads Analysis and System Thermodynamics (BLAST) Program," developed by the U.S. Army Construction Engineering Research Laboratory (CERL). These write-ups were provided by Douglas Hittle, CERL.

J. Marx Ayres, Henry Lau, Edward Sowell, and Daniel Skurkis of Ayres Associates, Los Angeles, California, served as reviewers for this manual and also conducted a series of user tests on the program.

The set of Cal-ERDA computer programs described by this Program Manual is the result of participation and cooperation of many persons. The combined efforts of many different laboratories, private companies, and individuals were essential to the successful completion of this project. The participation of state and federal agencies, university personnel, representatives of professional societies, and many individual architects and engineers is gratefully acknowledged.

The following organizations and personnel have made significant contributions to produce this new set of public domain computer programs for analyzing energy consumption in buildings.

National Laboratories

Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439

Robert M. Graven
Paul R. Hirsch

Lawrence Berkeley Laboratory
University of California
Berkeley, CA 94720

Arthur H. Rosenfeld
Frederick C. Winkelmann

Los Alamos Scientific Laboratory
University of California
P. O. Box 1663
Los Alamos, NM 87545

Bruce D. Hunn

Prime Contractor

Consultants Computation Bureau
594 Howard Street
San Francisco, CA 94105
Telephone: (415) 982-1293

Zulfikar O. Cumali
A. Ender Erdem

Principal Consultants

J. Marx Ayres
Ayres Associates
1180 South Beverly Drive
Los Angeles, CA 90035
Telephone: (213) 553-5285

Metin Lokmanhekim
3026 Fawnwood Drive
Ellicott City, MD 21043
Telephone: (301) 465-6499

STAFF PERSONNEL

The Cal-ERDA staff personnel who created, designed, and directed the development of the BDL were:

Zulfikar O. Cumali
A. Ender Erdem
Robert M. Graven
Arthur H. Rosenfeld

The contributions of each of the following individuals who programmed, implemented, debugged, and documented the language and its library are gratefully acknowledged.

Kenneth C. Arnold
Gloria A. Bennett
Tom R. Borgers
Patricia Bronnenberg
Fred Buhl
Robert L. Clair
Paul K. Davis
Edward T. Dean
Stephen C. Diamond
Ashok J. Gadgil
Dudley V. Goetschel
James J. Hirsch
Paul R. Hirsch
Jerry J. Kaganove
William L. Klein
Metin Lokmanhekim
Alan K. Meier
Howard C. Mitchell
Mark A. Roschke
C. David Sides
Roy L. Smith
Lavette C. Teague, Jr.
Eva F. Tucker
Aydin Ulkucu

The coordination of this project was a very large and complex task, involving many different participants and organizations, and was accomplished by:

Frederick C. Winkelmann

PROJECT SUPPORT

The Cal-ERDA project was funded by the following agencies:

State of California
Energy Resources Conservation
Development Commission
1111 Howe Avenue
Sacramento, CA 94525

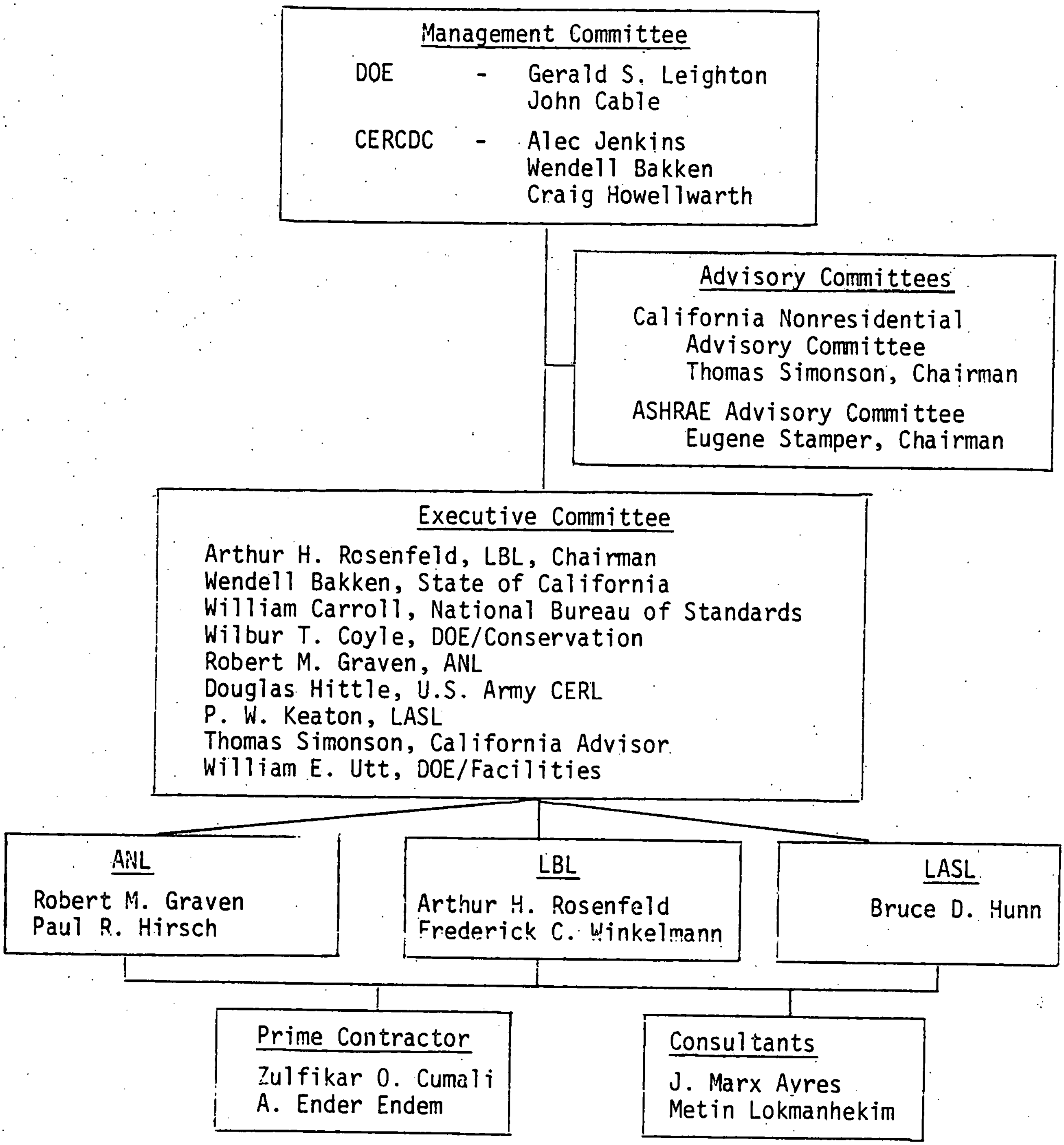
Alec Jenkins
Donald Watson
Wendell Bakken
Craig Howellwarth

U.S. Department of Energy
Office of Conservation
20 Massachusetts Avenue
Washington, DC 20545

Gerald S. Leighton
John Cable
Wilbur T. Coyle
Howard Ross

MANAGEMENT TEAM

A management team having the following organization and members monitored the progress, gave guidance, and managed the Cal-ERDA project.



Management Committee

DOE - Gerald S. Leighton
John Cable

CERCDC - Alec Jenkins
Wendell Bakken
Craig Howellwarth

Advisory Committees

California Nonresidential Advisory Committee
Thomas Simonson, Chairman

ASHRAE Advisory Committee
Eugene Stamper, Chairman

Executive Committee

Arthur H. Rosenfeld, LBL, Chairman
Wendell Bakken, State of California
William Carroll, National Bureau of Standards
Wilbur T. Coyle, DOE/Conservation
Robert M. Graven, ANL
Douglas Hittle, U.S. Army CERL
P. W. Keaton, LASL
Thomas Simonson, California Advisor
William E. Utt, DOE/Facilities

ANL

Robert M. Graven
Paul R. Hirsch

LBL

Arthur H. Rosenfeld
Frederick C. Winkelmann

LASL

Bruce D. Hunn

Prime Contractor

Zulfikar O. Cumali
A. Ender Endem

Consultants

J. Marx Ayres
Metin Lokmanhekim

PARTICIPANTS

The participation of the following people in the Cal-ERDA project is gratefully acknowledged.

Wendell Bakken
State of California, ERCDC
1111 Howe Avenue
Sacramento, CA 94525

William Carroll
National Bureau of Standards
Building 226
Washington, DC 20234

Wilbur T. Coyle
U.S. Department of Energy
Conservation Division
Washington, DC 20545

Donald Doughty
State of California, ERCDC
1111 Howe Avenue
Sacramento, CA 94525

Thomas Fischer
Oak Ridge National Laboratory
P. O. Box X
Oak Ridge, TN 37830

James Heldenbrand
National Bureau of Standards
Building 225
Washington, DC 20234

Douglas Hittle
U.S. Army CERL
Box 4005
Champaign, IL 61820

Craig Hoellwarth
State of California, ERCDC
1111 Howe Avenue
Sacramento, CA 94525

Alec Jenkins
State of California, ERCDC
1111 Howe Avenue
Sacramento, CA 94525

Gerald Jones*
University of Texas
Architectural Engineering
Austin, TX 78712

P. W. Keaton
Los Alamos Scientific Laboratory
P. O. Box 1663
Los Alamos, NM 87545

Henry Lau
Ayres Associates
1180 South Beverly Drive
Los Angeles, CA 90035

Gerald S. Leighton
U.S. Department of Energy
Conservation Division
Washington, DC 20545

Allen Lober
Hellman & Lober
6380 Wilshire Boulevard
Suite 1506
Los Angeles, CA 90048

Robert Pankhurst
U.S. Department of Energy
San Francisco Operations Office
1333 Broadway
Oakland, CA 94612

Thomas Parish*
Entex, Inc.
P. O. Box 2628
Houston, TX 77001

David M. Pellish
U.S. Department of Energy
Conservation Division
Washington, DC 20545

William Rudoy*
University of Pittsburgh
Department of Engineering
Pittsburgh, PA 15261

* ASHRAE Advisory Committee.
Note: This edition does not reflect ASHRAE review comment.

C. David Sides
Omnitechnics, Inc.
1703 Ridge Road
Champaign, IL 61820

Thomas Simonson
Simonson and Simonson
612 Howard Street
San Francisco, CA 94105

Daniel Skurkis
4186 Chase Avenue
Mar Vista, CA 90066

Edward F. Sowell
California State University, Fullerton
Fullerton, CA 92634

Eugene Stamper*
New Jersey Institute of Technology
Newark, NJ 07102

William E. Utt
U.S. Department of Energy
Construction, Planning, and Support
Washington, DC 20545

Donald Watson
State of California, ERCDC
1111 Howe Avenue
Sacramento, CA 94525

* ASHRAE Advisory Committee.
Note: This edition does not reflect ASHRAE review comment.

NOTICE

The explicit function of these computer programs is to aid in the analysis of energy consumption in buildings. These programs are not intended to be the sole source of information relied upon for the design of buildings. The basic authority that should be relied upon is the judgment and experience of the engineer.

The technical information in these computer programs has been compiled from the best available sources and is believed to be correct. Many of the equations and much of the methodology used by the Cal-ERDA computer programs are based on the algorithms published by the American Society of Heating, Refrigerating, and Air Conditioning Engineers, Inc. (ASHRAE), described in Ref. 1. However, this edition of the Program Manual does not reflect ASHRAE review comment.

ASHRAE POLICY STATEMENT ON THE USE OF THE NAME OF ASHRAE IN CONNECTION WITH COMPUTER PROGRAMS

Numerous computer programs have been developed and are being marketed with the statement that they are in accordance with the ASHRAE Handbook of Fundamentals and/or are based on ASHRAE calculation procedures.

The Society does not endorse and is not responsible for any computer programs which may or may not use information published in the ASHRAE Handbook of Fundamentals or other ASHRAE publications.

Persons using the name of ASHRAE in connection with any computer program or persons considering the use of computer programs which infer that ASHRAE has endorsed them are hereby cautioned that ASHRAE does not make any such endorsement.

Any further distribution by any holder of this document or of the data therein to third parties representing foreign interests, foreign governments, foreign companies, and foreign subsidiaries of foreign divisions of U.S. companies should be coordinated with the Director, Conservation Division, Department of Energy.

By acceptance of this report, the recipient acknowledges the U.S. Government's right to retain a nonexclusive, royalty-free license in and to any copyright covering this report. The recipient may not reproduce or change any portion of this report without receiving written approval in advance from the U.S. Government as represented by the U.S. Department of Energy, the University of Chicago, Argonne National Laboratory, and the Lawrence Berkeley Laboratory and the Los Alamos Scientific Laboratory of the University of California.

This report was prepared as an account of work sponsored by the U.S. Government and the State of California. Neither the U.S. nor the State of California, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe on privately owned rights.

No references may be made to the Cal-ERDA computer programs as having been responsible for calculations unless a complete, up-to-date copy of these programs is used to perform the calculations and that copy has been certified by the Cal-ERDA staff as distributed through the DOE/Argonne Code Center at Argonne National Laboratory.

STATUS – OCTOBER 1977

This manual is intended to document the Cal-ERDA computer program, as well as for study and review by the program users. It is incomplete in some respects and also contains some inconsistencies. For example, two types of flow charts are used, conventional and structured; a decision on a common format will await the next draft. Some chapters that have few flow charts are expected to be expanded to include more of them.

Since the Cal-ERDA programs are still under development, this manual can only document "frozen" versions of the program as of specific dates. The subprograms were being developed at different rates, and thus the documentation on each is based on a different version of the program. The listings used are as follows.

Subprogram	Cal-ERDA Version	Dated
LOADS	1.2	July 11, 1977 (except as otherwise noted)
SYSTEMS	-	June 15, 1977 (except as otherwise noted)
PLANT	1.1	May 5, 1977
ECONOMICS	1.1	May 5, 1977

Significant changes and additions have been made to the SYSTEMS Program that are not described in this document. These changes, which are included in the October 26, 1977 version 1.3 of the code, will be documented in the next edition of this manual.

The PLANT Program was undergoing very few changes, and therefore the May 5, 1977, listing was nearly up to date. An updated listing of ECONOMICS was not received from the programmers in time for inclusion in this edition. Subsequent editions will cover updated versions of the program.

I. INTRODUCTION

A. Documentation

1. Users Manual. This manual provides the information and instructions required to use the Cal-ERDA set of computer programs. It contains a brief description of each program and an introduction to a new programming language called the Building Design Language (BDL). It also contains a description of the library data used to perform the desired calculations. Finally, it contains examples that illustrate the use of the Cal-ERDA computer programs.

2. Program Manual. The Program Manual provides the user with necessary information to understand in detail the Cal-ERDA set of computer programs. It contains a summary of the equations and algorithms used to perform the calculations, as well as flow charts to describe how each program operates. (See Sec. D.) The relationship of the Cal-ERDA algorithms to ASHRAE algorithms is also given and traced to ASHRAE documentation (Refs. 1 and 2). The operation, structure, and logical sequence of calculations is described in a step-by-step manner for each of the major computational programs.

3. Listings. A listing of the FORTRAN IV statements for each program gives the specific instructions to perform the desired calculations. Either update source or compiler input format listings are available. Listings of the weather analysis programs and the response factor program are also available. The most desirable form is to request listings on microfiche, since that form is easily sent through the mail. A paper copy (print-out) listing all the Cal-ERDA computer programs and the library data is quite thick and should only be requested by persons wishing to implement these codes on their machines.

4. Magnetic Tape. Card image copies of the source code for the Cal-ERDA programs on magnetic tape are available. This form provides all the necessary programs and library data to allow a convenient implementation on other machines. It is the recommended form for transferring these programs to other machines. The standard form is on nine-track tapes, at 800 bits per inch. For CDC machines, a binary code is used; for IBM machines, the EBCDIC code is used.

B. Program Manual Purpose and Organization

This manual provides a detailed description of each of the Cal-ERDA programs. It includes a description of each subroutine, as well as the data handled by each, and function and/or flow charts for most subroutines.

The purpose of this manual is to be the complete engineering documentation of the actual hardware modeled by the code, and, as such, includes system schematics and discussion of how the subroutines relate to the appropriate ASHRAE literature. Since most of the methods are based on ASHRAE algorithms, those that are have been associated with the appropriate documents: Refs. 1 and 2. Where algorithms are obtained elsewhere, or where the ASHRAE algorithms have been modified, this is so stated.

Also, this manual is to serve as a programmer's manual and contains detailed flow charts, as well as data structure, manipulation, and variable name descriptions.

Chapter I contains an outline of the Cal-ERDA programs and a list of general program manual references.

Chapter II discusses the operation of the BDL input processor and how the data arrays are structured.

Chapter III describes the LOADS Program, Chap. IV the SYSTEMS Program, Chap. V the PLANT Program, and Chap. VI the ECONOMICS Program.

Chapter VII outlines the REPORT Program.

Chapter VIII describes the weather programs, while Chap. IX describes the utility programs, specifically the EXECUTIVE Program.

Chapter X describes the library data file formats, structure, and access.

C. Summary of Cal-ERDA Computer Programs

Figures I.1 and I.2 illustrate the Cal-ERDA computer programs used to analyze energy consumption in buildings. A user controls the programs by using BDL instructions. The instructions are decoded using the BDL Processor Program, and commands are forwarded to any of the other appropriate programs. For example, commands to select weather data are sent to the weather data library and used to select weather data for the LOADS Program. Similarly, BDL commands are used to instruct the computational programs (LOADS, SYSTEMS, PLANT, and ECONOMICS) to save the values of various calculations for later use, such as for inclusion in a print-out of the results.

1. EXECUTIVE Program. The EXECUTIVE Program performs the necessary bookkeeping functions to assure that the operation of the programs is properly executed. It connects the independent programs, provides the necessary computer memory space (storage), and it creates a set of control cards in the proper sequence to control the operation of the programs.

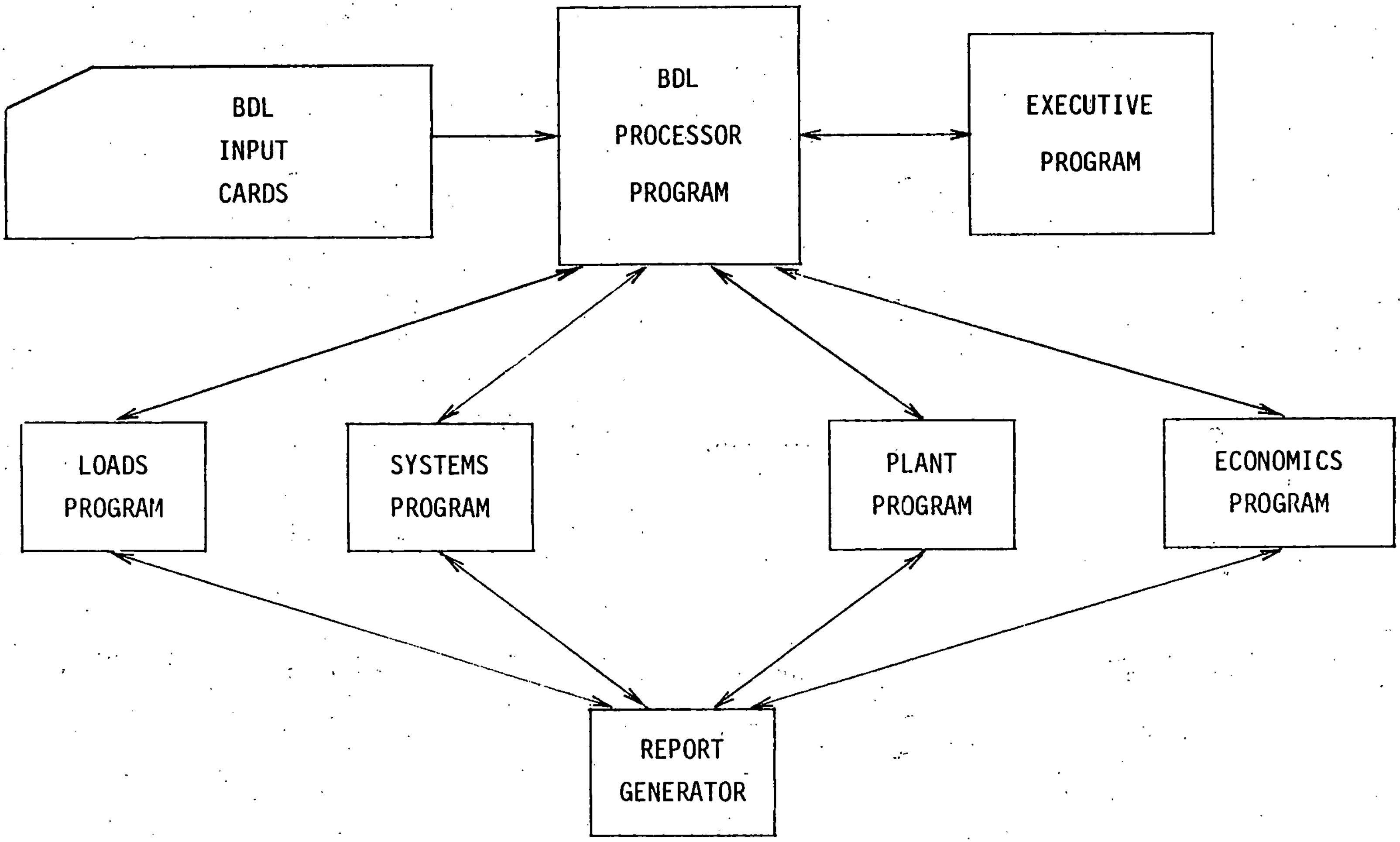


Fig. I.1. Cal-ERDA program network.

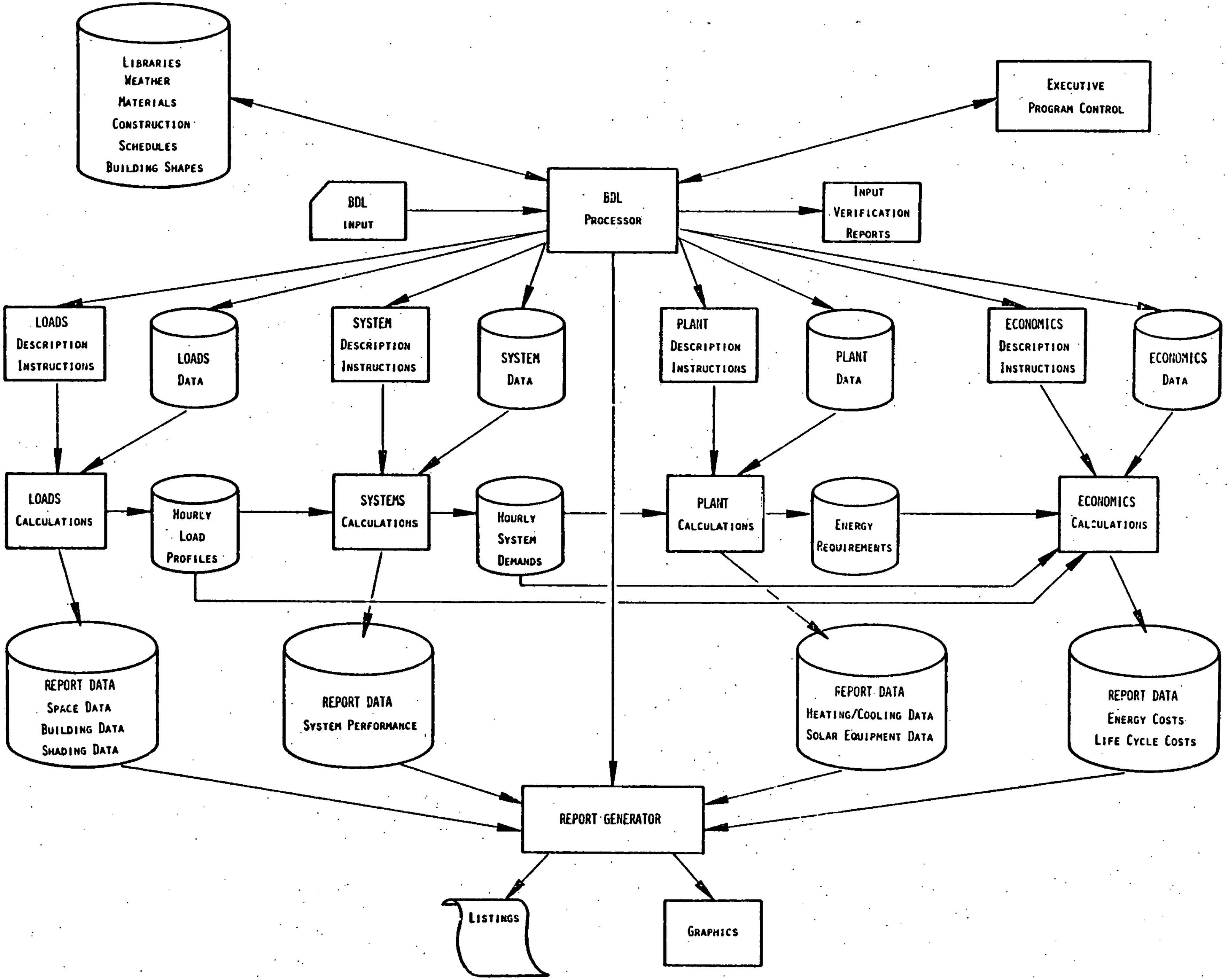


Fig. I.2. Cal-ERDA functional flow chart.

2. BDL Processor. The BDL processor checks each BDL instruction for proper form and content. The input text (BDL) is read sequentially and each instruction is examined to determine if any reserved descriptors or keywords have been used and if any values (numbers) have been assigned to specific variables. The BDL Processor contains all the decoding routines and it checks for values that are beyond the expected range for input variables. If no values are specified, the Processor inserts a default value. If a default value is used, it will appear in the listing of input data that is printed before each major program is run. The BDL Processor also prepares the data input files for use by the LOADS, SYSTEMS, PLANT, or ECONOMICS (LSPE) Programs.

The BDL Processor collects whatever data the user desires from the various permanent libraries (e.g., data from the weather files) and from various user-defined libraries (e.g., a user-entered schedule for simulating the hour-by-hour use of lighting). It then creates a new "job input file" that contains the sequence of instructions and data a user wants the computer to use to perform the energy consumption calculations.

The final task the BDL Processor performs is to assemble the results of calculations performed by the LSPE Programs. The results of each program are sent to an output printer (or microfiche, or plotter) according to instructions given by the user. It is important to recognize that each of the LSPE Programs depends on the results of some or all of the previous programs and that many variations and combinations are allowed. Each of the LSPE Programs can be run repeatedly to study the effect of variations in the building's shape, its HVAC system, its plant equipment, or the cost parameters.

3. LOADS Program. The LOADS Program calculates the cooling and heating loads largely using the algorithms described in Ref. 1. The subroutine descriptions in Chap. III state which subroutines are not based on ASHRAE literature. It is assumed that the reader is familiar with the contents of Ref. 1.

The LOADS Program calculates the amount of heat entering and leaving a building for each hour of a year (8760 hrs). The heat gains and losses through walls, roofs, floors, windows, and doors are each calculated separately. Heat transfer by radiation and conduction through the building skin is computed considering the effects of the thermal mass, placement of insulation, sun angle, cloud cover, and building location, orientation, and architectural features.

Infiltration loads are calculated based on the difference between the inside and outside conditions and an assumed leak rate (crack method).

Internal use of energy for lighting and equipment is also computed, according to schedules assigned by the user for each piece of equipment that affects the energy balance of each space. The latent and sensible heat given off by the building occupants is calculated as an hour-by-hour function of the occupancy of the building.

4. SYSTEMS Program. The SYSTEMS Program contains the algorithms and equations for simulating performance of the HVAC distribution systems (secondary equipment) used to control the air temperature and humidity within the building. Many of the equations used in the SYSTEMS simulation procedure are given in Refs. 2 and 3. The subroutine descriptions of Chap. IV specify the origin of each of the algorithms used in the SYSTEMS simulation procedure. The ASHRAE algorithms have been organized and coded to allow a user to select one of the preprogrammed space conditioning systems described in Chap. IV. A user may choose one of these systems and provide the necessary input data for the simulation calculations.

The SYSTEMS Program uses the output information from the LOADS Program and a list of user-defined system characteristics (e.g., air flow rates, thermostat settings, schedules of HVAC operation, temperature setback schedules, etc.) to calculate the hour-by-hour energy requirements that the HVAC distribution system must supply to maintain the desired conditions.

5. PLANT Program. The PLANT Program contains the necessary equations to calculate the performance of the primary energy conversion equipment. The operation of each plant component (e.g., boiler, absorption chiller, compression chiller, cooling tower, etc.) is modeled based on operating conditions and part-load performance characteristics. A user selects the type of plant equipment he decides to model (e.g., two-stage absorption chiller), the size of each unit (e.g., 100 tons), the number of units, and the number of units simultaneously available. Values for equipment lifetime and maintenance may also be entered by a user if he decides not to use the default values given for these variables. A user may also specify the sequence of equipment operation as a function (e.g., from 0 to 500,000 BtuH, use unit 1; from 500,001 to 10^6 BtuH, use units 1 and 2). The PLANT Program uses this user input information and results from the LOADS and SYSTEMS Programs to calculate the energy consumption of the primary heating, cooling, and ventilation equipment.

Monthly consumption of electricity, natural gas, oil, etc., is placed on an output file for later use; this file is subsequently read by the ECONOMICS Program.

6. ECONOMICS Program. The ECONOMICS Program computes the life-cycle cost of operating the building according to the methodology of Ref. 4. The usual input required is the interest rate, labor inflation rate, materials inflation rate, energy inflation rate, project life, cost of labor ($/hr), and site cost factors. Equipment costs can be assigned by generic class or by specific size. The present cost of electricity, natural gas, oil, and coal may also be entered by a user, as well as various cost escalation factors. These user-specified input data are combined with the energy consumption results of the LOADS, SYSTEMS, and PLANT Programs to run the economic analysis program. A present value for each item (e.g., electricity or equipment) is calculated and summed to compute the life-cycle cost. Results of the ECONOMICS Program calculations are sent to its output file.

7. REPORT Program. A REPORT Program is used to collect information from the standard output files of the LSPE Programs. The output data are then arranged in lists or tables according to one of the "standard output reports" a user may select. If a user (e.g., a research user) wishes to examine a particular variable that is not available in the standard output reports, he may select the variable and print it using the REPORT Program.

8. WEATHER Programs. The WEATHER Programs are used to examine and prepare weather data for use by the Cal-ERDA Programs. A table of weather stations for which data are presently available in the Cal-ERDA library is given in Chap. VIII. The weather data used for the hourly simulations are the Test Reference Year (TRY) data compiled and distributed by the National Oceanic and Atmospheric Administration (NOAA). Each meteorological variable may be changed, printed, or plotted as desired by the user.

Weather data may be manipulated independent of the LSPE Programs. The ordinary use of the LSPE Programs to calculate the energy consumption of a building does not require use of the weather analysis programs. The typical user will merely select a particular weather station by specifying an access code to inform the LSPE programs which library weather data to use.

9. LIBRARY Program. The LIBRARY Program is used to add, delete, or modify data in the library. The library contains the following types of

data: weather, construction, materials, schedules, building shapes, and cost. The weather section of the library contains TRY data for the locations listed in Chap. VIII. It also contains weather data for the national laboratory sites and the California climate regions data compiled by L. W. Crow, Consultant to the California Energy Resources Conservation and Development Commission. As additional weather data becomes available, it will be added.

The construction section of the library contains data used to simulate the thermal performance of walls, roofs, and floors. Each of the ASHRAE constructions listed in Ref. 5 and each of the typical California constructions listed in Ref. 6 is given a unique alphanumeric name (e.g., WALL.14, ROOF.21, or FLOOR.5).

The user can select the data for a surface simply by using its given library name. For example, the library item named WALL.14 contains the response factor data used in the simulation of the heat transfer through the wall having the layer-by-layer characteristics described in the library under the library name WALL.14. The data transferred are the response factors for each of the typical constructions.

The materials section of the library contains data on the thermal properties of materials. These data are used in the calculation of heat transfer through space boundaries. Different materials may be laminated (mathematically) to model a wall layer by layer.

The schedule section includes graphs of various schedules that may be applied to calculate the hour-by-hour thermal energy input to a space because of lighting, equipment, or occupants. If none of the preprogrammed schedules apply to the problem being studied, a user may define schedules as necessary. Changes to the schedule data may also be made by the user.

In summary, the library consists of permanent data (preprogrammed) and private, user-defined data. Permanent library data are available to all Cal-ERDA users, while private data files are for individual use only.

D. Description of Flow Chart Formats

Two types of flow charts have been included in this draft of the Cal-ERDA Program Manual: the more familiar "standard" or "programmer's" flow charts in which arrows lead the reader from box to box and the less familiar "structured" flow charts that appear like a typed page with lines drawn between statements. Because of some readers' unfamilarity with both types, Fig. I.3 shows two equivalent flow charts, one in each format.

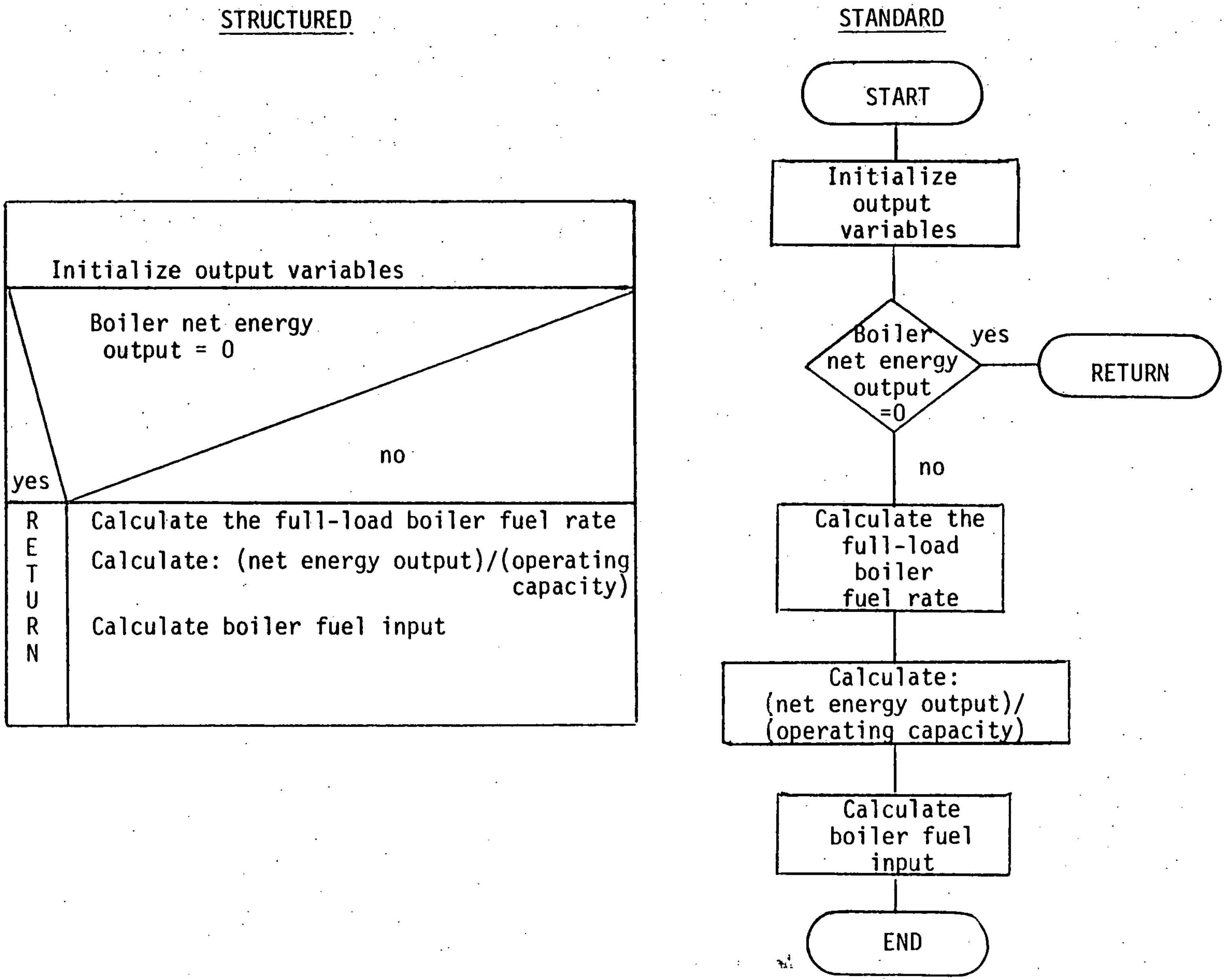

STRUCTURED
Initialize output variables
Boiler net energy output = 0
yes
no
RETURN
Calculate the full-load boiler fuel rate
Calculate: (net energy output)/(operating capacity)
Calculate boiler fuel input
STANDARD
START
Initialize output variables
Boiler net energy output =0
yes
RETURN
no
Calculate the full-load boiler fuel rate
Calculate: (net energy output)/ (operating capacity)
Calculate boiler fuel input
END

Fig. I.3. Structured and standard flow chart equivalence examples.

Later editions of the Cal-ERDA Program Manual will contain only one flow-chart format. As a decision has not yet been made between formats, the editors request that users make their preference known and the reasons for preferring one format over the other.

II. BDL PROCESSOR

A. Building Design Language (BDL)

BDL is a command-word-structured, free-format language designed to greatly simplify the process of defining a model for a given building and manipulating the components of that model to produce an energy-conservative design. It consists of a set of commands, each command having associated with it a set of variables called keywords. These variables are assigned values by the user, or, if ignored, take on predetermined default values. Each of the four principal computational programs of Cal-ERDA, LOADS, SYSTEMS, PLANT, and ECON, has its own BDL vocabulary (set of keywords). These are called LOADS Design Language (LDL), SYSTEM Design Language (SDL), PLANT Design Language (PDL), and ECON Design Language (EDL). Only the LOADS Design Language (LDL) will be discussed as an example.

B. LDL Hierarchy

LDL has a hierarchal structure in which each level defines a class of data shared by all levels below it. This permits defining a class of data once, and then recalling that data at whatever level is desired by use of an appropriate name. This data level structure is also true for each of the other subset languages, SDL, PDL, and EDL.

A building must be systematically subdivided into its elemental components to simulate its thermal performance. The first order of subdivision is a definable volume called a space. This is the beginning of the hierarchy, and the space is the basic building block for simulation. The LDL command SPACE is used to identify this unique volume. Spaces are normally selected to encompass conditioned volumes that have unique loads and are controlled by a single thermostat. Spaces can also be unconditioned, such as a plenum or an attic.

The next level in the LDL hierarchy is the boundary surface of SPACE. This includes exterior walls, roofs, interior walls, ceilings, floors, and underground walls and floors. Each of these categories of boundary surfaces is identified by a command word in LDL.

Finally, each of the boundary surfaces may have apertures (doors and windows), as well as attachments (overhangs and fins).

C. The Building

Although not actually a part of the LDL hierarchy, building location and orientation data must be input before SPACE data can have meaning. The building must be located as to latitude, longitude, and international time zone. Also, the altitude of the site and the direction of the building with respect to north must be established. The latter item is identified by assigning an appropriate value to the BDL keyword BUILDING-AZIMUTH, which is defined as the angle, measured clockwise from true north, to the Y-axis of the building (YB) as shown in Fig. II.1, where

XB = X-axis of building,
YB = Y-axis of building,
ZB = Z-axis of building, and
BAZ = building azimuth angle.

The building-coordinate system may be assigned to the building in whatever manner the user may desire, and, once fixed, will be the reference point for locating the various spaces that comprise the building.

D. Response Factors

Calculation of transient heat transfer in the LOADS Program requires thermal response factors for each wall construction in the building. These are calculated in the BDL program subroutine PONSFAC using ASHRAE methodology (Ref. 1). Each wall (roof) is input as layer-by-layer data by the user, and a set of response factors is generated. There is no library of values for predefined walls in the program. A detailed write-up of subroutine PONSFAC will be provided in subsequent editions of this manual.

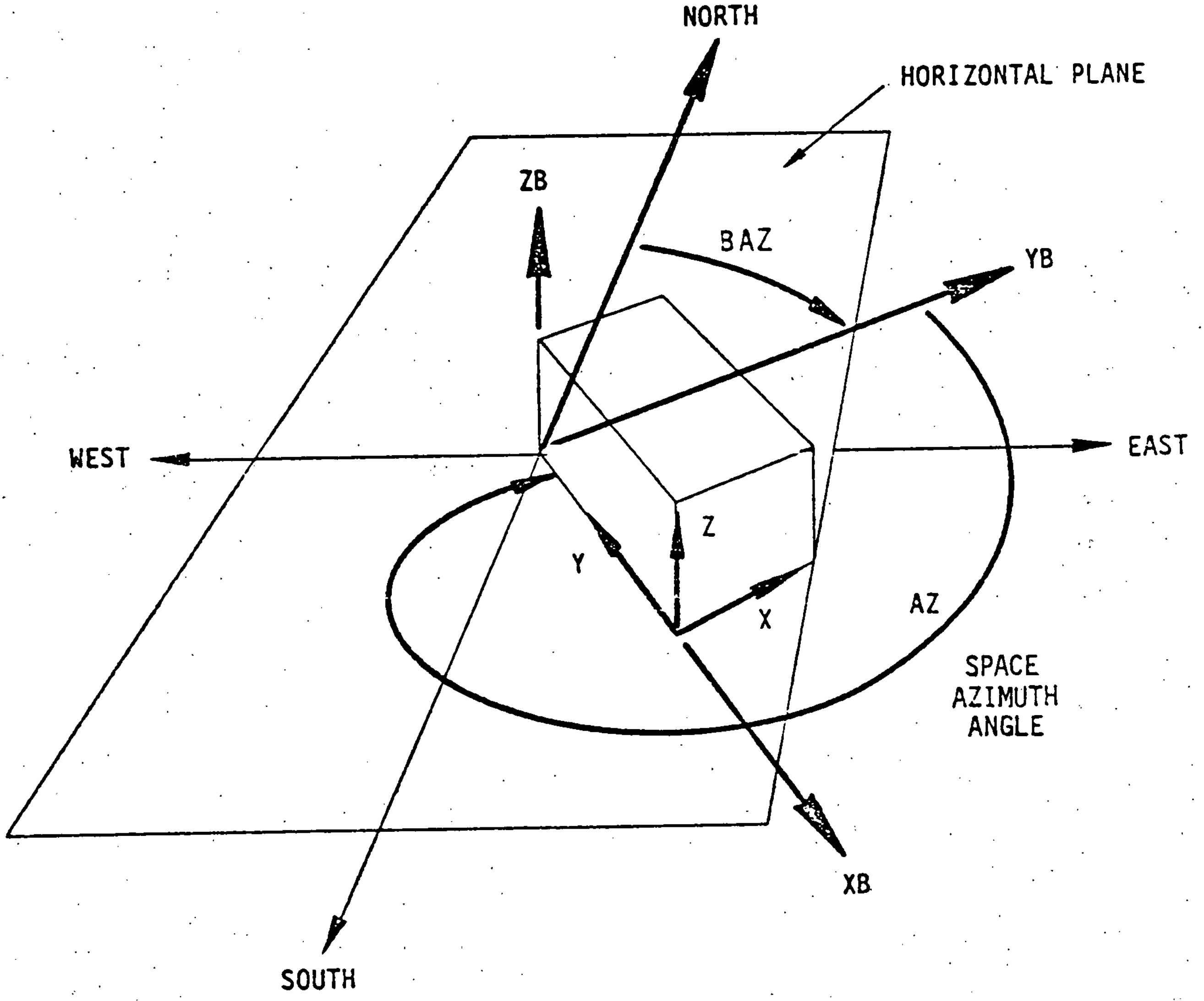


Fig. II.1. Building location and orientation.

III. LOADS PROGRAM

A. Objective and Description

The LOADS Program calculates the heating and cooling loads seen by each space (zone) in the building for each hour of a year (8760 hrs). Loads resulting from the following are calculated:

1. Transmission gains and losses through walls, roofs, floors, and windows.
2. Solar gains through windows.
3. Internal gains from people, lights, and equipment.
4. Infiltration gains and losses caused by pressure differences across openings.

Note that ventilation air gains and losses resulting from fresh air requirements are calculated in the SYSTEMS Program as described below.

The load calculations are largely based on the algorithms set forth by ASHRAE in Ref. 1. Differences between Cal-ERDA and ASHRAE algorithms are discussed in the individual subroutine descriptions.

Using these algorithms, the LOADS Program performs an hourly energy analysis using actual hourly weather data read from a TRY tape. Hourly heating and cooling loads for each space are calculated for the year, or portion thereof, and the results passed to the SYSTEMS Program (Chap. IV). The peak heating and cooling loads observed for the calculational period are printed out.

Note that since ventilation loads are calculated in SYSTEMS rather than in LOADS, they must be accounted for separately to determine required equipment sizes.

Input to LOADS represents building architecture, orientation, surroundings, and construction, as well as local weather and solar characteristics. The output consists of hourly weather and psychrometric data, concurrent hourly sensible loads, and latent loads (cooling only), broken down by walls, ceilings, glass conductance, internal surfaces, underground surfaces, occupants, light, equipment, and process heat to the space. The time of occurrence of the space and total building cooling and heating peak is identified, along with the concurrent dry- and wet-bulb temperatures.

Although a considerable increase in speed over previous programs has been achieved by careful programming and a reorganization of the algorithms and input/output, the algorithms used are basically the same as the ASHRAE algorithms except where noted in the subroutine descriptions below. The results, however, are virtually identical to those produced by NECAP (Ref. 3). The following major differences between Cal-ERDA and NECAP should be noted.

- Whereas the wall and roof response factors are computed in a separate program in NECAP, they are computed in the Cal-ERDA BDL Program, with weighting factors computed in the LOADS Program.
- Ventilation, return air, and fan heat loads, as well as design supply air flows, are calculated in SYSTEMS rather than in LOADS.
- The shading algorithm used in NECAP has been replaced by a more sophisticated, faster version.
- Recommended heating and cooling extraction rates for the SYSTEMS Program are not calculated.

The other differences are in the data and subroutine arrangements. Many NECAP subroutines and/or ASHRAE algorithms and variables have been renamed, and some subroutines have been combined or separated. Also, the data array structure has been modified to facilitate more efficient input/output and retrieval and to allow a more efficient use of core storage. These differences are detailed in the subroutine listings and calculation sequence outlined below.

1. Design Loads vs Building Energy Requirements. There is a difference between thermal load calculation procedures for use in the design of the heating and cooling facilities and the procedure for estimates of energy requirements. The load calculation procedure, as described in Ref. 5, is for the design calculation. It is valid for simplified design conditions that assume steady-state conditions (such as is largely the case for heating load calculations) or a steady periodic heat flow (as is the case of the cooling load calculation). The load calculated under these design conditions may be adequate for sizing or selecting heating and cooling equipment and systems, but it is unsatisfactory for predicting the actual hourly thermal loads and building energy consumption.

A load calculation procedure for the determination of energy requirements should be able to predict the performance of the building's heating and cooling system when combined with a total system simulation program under actual (randomly fluctuating) climatic and operating conditions. An important distinction between the design load calculation and energy calculation is that the former uses a single value, while the latter generates a series of values or time series of thermal loads evaluated at every hour of the year.

Since the load determination of energy requirements involves many more calculations when compared to an ordinary design load determination, the use of a computer is mandatory.

The basic scheme of the load calculation procedure is first to evaluate the instantaneous heat gains because of solar radiation and heat conduction through the building envelope as accurately as possible. These heat gains are then balanced with those from lighting and other internal sources with a specific consideration that the sum of all the instantaneous heat gains is not the instantaneous cooling load.

The solar radiation is first absorbed by solid objects in the space and is not manifested as a cooling load until some time later. Exact evaluation of the space cooling load requires solution of a set of the energy balance equations for all the space surfaces, space air, and space heat gains.

To simplify this calculation procedure, the weighting factor concept is used so that each heat gain contributes to the space cooling load through its own weighting factors.

2. Response Factors/Weighting Factors Concept. The use of response factors and weighting factors for the calculation of the building thermal load is based on the work of G. P. Mitalas and D. G. Stephenson of the National Research Council of Canada (NRC) (Refs. 7, 8, and 9, and Appendix of Ref. 1).

The basic scheme of this concept is first to calculate the set of heat transfer functions, called response factors, required to accurately determine the transient flow of heat through building exterior walls and roofs as they react to randomly fluctuating climatic conditions. These response factors are a function of the type of materials used and their order of placement in the wall or roof. They are calculated once for each Cal-ERDA run period.

These heat fluxes are then balanced with other internal sources (e.g., lights) with the consideration that the sum of all of the instantaneous heat fluxes is not the instantaneous load on the system. Solar radiation, for example, is not manifested as a load until some time after entering a space. It is first absorbed by some opaque object that then transfers the energy to the air and finally to the HVAC system. Exact evaluation of this load requires solution of a set of the heat balance equations for all the space surfaces, space air, and space heat fluxes.

To simplify this calculation, the weighting factor concept is introduced so that each heat flux contributes to the load through its own weighting factors. The Cal-ERDA Program uses a set of weighting factors for a typical room, previously calculated and used by ASHRAE, to "weight" the effect of solar gains, lights, and exterior heat conduction on the building HVAC system. These factors are not calculated in Cal-ERDA, but the response factors are.

The NRC method of calculating response factors is based on the assumption that the characteristics of any physical system can be described by defining it with the system being linear and invariable (principle of superposition). Linearity implies that the magnitude of the response is linearly related to the magnitude of the excitation; invariability means that equal excitations applied at different times always produce equal responses. There are, therefore, three steps in calculating the response (response factors) of a linear, invariable system (wall, roof, etc.).

(1) Resolve the excitation into a series of simple components.

(2) Calculate the response for each component.

(3) Add the responses for the separate components.

Calculation of these response factors for Cal-ERDA takes place in the BDL Program and not in LOADS.

B. Program LOADS

PROGRAM LOADS (STNDFL = 1000, SYSDES = 400, SYSHRL, WEATHR, OUTPUT, DEBUG)

Description

This is the main controlling routine for operation of the Cal-ERDA LOADS Program. All other subroutines are called either directly or indirectly by this program.

Subprograms Calling This Program

None

Subprograms Called by This Program

SUBROUTINE BLOCKIO (ISYSHR)
SUBROUTINE READSF (IERRF)
FUNCTION MONLEN
SUBROUTINE DAYCLC
SUBROUTINE RPT1
SUBROUTINE RPT2
SUBROUTINE DESFOU

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/DATE/	IDOY, IDOW	IFSTHR, IPRDFL, IYR, ILEAPF, IMON, IDAY, IDOY, IDOW
/FILES/	None obtained	ISTNDF, ISYSDS, ISYSHR, IWEATH, IPRT, IDEBUG
/GPNTRS/	Not currently used here	
/PNTRS/	Not currently used here	
/RUNPRD/	NRUNPR, IRUNPR(6,12)	None placed
/RUNTIM/	None obtained	IDAYBG, IMONBG, IYRBG, IDAYND, IMONND, IYRND
/SHADWF/	None obtained	IGOLGE

Declarations

```
DATA ISTNDF/GLSTNDFL/, ISYSDS/GLSYSDES/, ISYSHR/GLSYSHRL/
DATA IWEATH/GLWEATHR/
DATA IPRT/GLOUTPUT/, IDEBUG/GLDEBUGF/
```

Input

Name	Description
IERRF	Error flag, no error if equal to zero
NRUNPR	The number of LOADS run periods specified
IRUNPR(6,12)	The beginning and ending dates for each LOADS run period

Output

Name	Description
IDAYBG	The beginning day of the LOADS run period
IMONBG	The beginning month of the LOADS run period
IYRBGG	The beginning year of the LOADS run period
IDAYND	The ending day of the LOADS run period
IMONND	The ending month of the LOADS run period
IYRNDD	The ending year of the LOADS run period
ILEAPF	Leap year flag
IGOLGE	Shadow calculation flag, 1 = calculate, 2 = do not calculate; shadow ratios are calculated once per month
IDOY	The day of the year
IDOW	The day of the week

Calculation Procedure

1. Define LOADS files.

```
DATA ISTNDF/GLSTNDFL/, ISYSDS/GLSYSDES/, ISYSHR/GLSYSHRL/
DATA IWEATH/GLWEATHR/
DATA IPRT/GLOUTPUT/, IDEBUG/GLDEBUGF/
```

2. Set up hourly output file for SYSTEMS.

```
CALL BLOCKIO (ISYSHR)
```

This subroutine enables the writing of a blocked binary file with no record marks. It is an LBL BKY operating system subroutine and must be replaced by an equivalent subroutine on other systems.

3. Initialize for first hour.

```
IFSTHR = 1
```

4. Read the standard files, initialize for each LOADS run, and return error flag.

```
Call READSF (IERRF)
```

5. Check error flag.

```
IF (IERRF. NE. 0) GO TO 8000
```

Statement No. 8000 will stop the program and write an error message.

6. Loop through each LOADS run period and initialize the beginning and ending dates.

```
DO 900 IRN = 1, NRUNPR
IDAYBG = IRUNPR (1,IRN)
IMONBG = IRUNPR (2,IRN)
IYRBGG = IRUNPR (3,IRN)
IDAYND = IRUNPR (4,IRN)
IMONND = IRUNPR (5,IRN)
IYRNDD = IRUNPR (6,IRN)
```

7. Set initial preparation flag.

```
IPRDFL = 3
```

This will cause the LOADS run to loop through the first day calculations three times before proceeding. This allows the weighting factors to stabilize.

8. Loop through year, check for leap year, and set beginning and ending months of run period.

```
DO 800 IYR = IYRBG, IYRND
ILEAPF = 0
IF ((IYR/4)*4 .EQ. IYR) ILEAPF = 1
IMON2 = 12
IF (IYR .EQ. IYRND) IMON2 = IMONND
IMON1 = 1
IF (IYR .EQ. IYRBG) IMON1 = IMONBG
```

9. Loop through months and set beginning and ending days.

```
DO 600 IMON = IMON1, IMON2
IDAY1 = 1
IF ((IYR .EQ. IYRBG) .AND. (IMON .EQ. IMONBG)) IDAY1 = IDAYBG
IDAY2 = MONLEN (IMON,ILEAPF)
IF ((IMON .EQ. IMONND) .AND. (IYR .EQ. IYRND)) IDAY2 = IDAYND
```

10. Set shadow calculation flag to initialize shadow ratios.

```
IGOLGE = 1
```

The shadow ratios are calculated for the first run-through and thereafter once per month.

11. Loop through the daily calculations.

```
DO 400 IDAY = IDAY1, IDAY2
IDOY = IDOY + 1
IDOW = IDOW + 1
IF (IDOW .GT. 7) IDOW = 1
CALL DAYCLC
IGOLGE = 0
```

12. Check initial preparation flag. If IPRDFL is not equal to zero, rerun the first-day calculations. This will be done three times to allow stabilization of the weighting factors.

```
IF (IPRDFL .EQ. 0) GO TO 400
IPRDFL = IPRDFL - 1
GO TO 300
```

13. Continue each loop.

```
400 CONTINUE
600 CONTINUE
800 CONTINUE
900 CONTINUE
```

14. Produce user reports.

```
CALL RPT1
CALL RPT2
```

15. Create an output design file for SYSTEMS.

```
CALL DESFOU
```

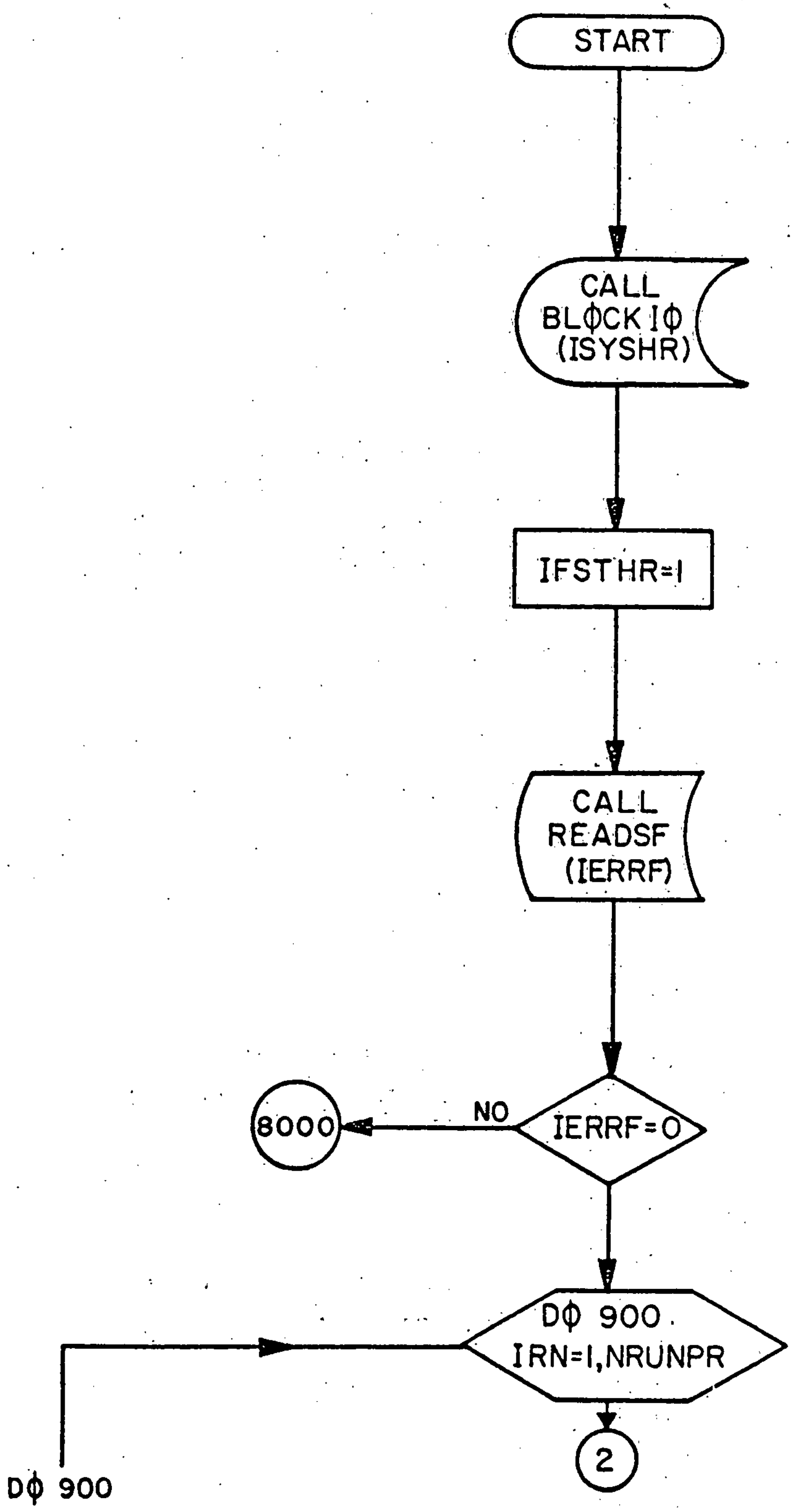
16. End program.

```
GO TO 9000
8000 WRITE (IPRT, 8001)
8001 FORMAT (*0---ERROR---LOADS---*)
9000 CONTINUE
END
```

ASHRAE Verification

The Program LOADS routine in Cal-ERDA has no equivalent ASHRAE routine. It simply controls the calculation sequence and disposition of data during a run period.

1. Program LOADS Flow Chart.



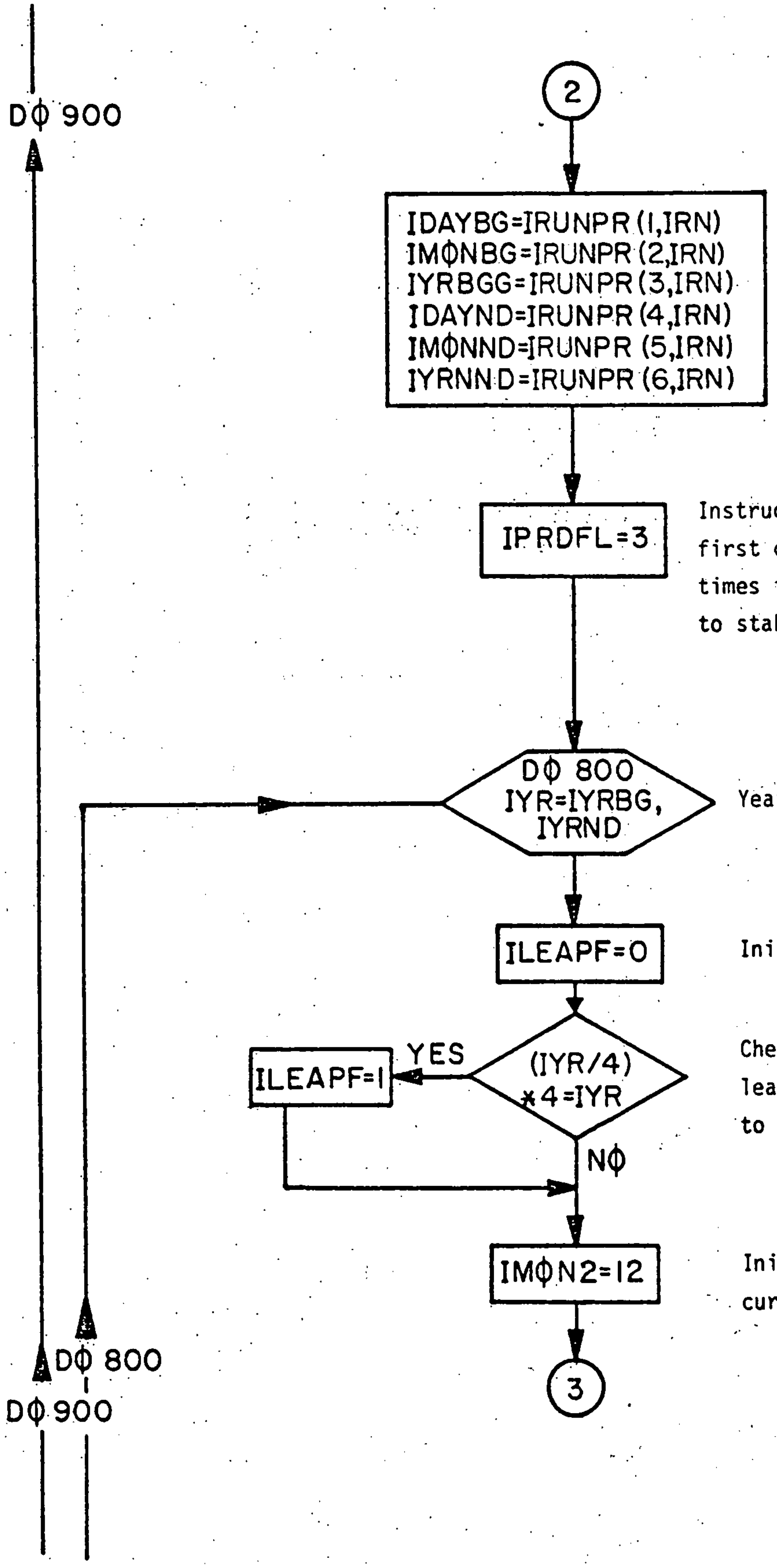
Enables LØADS to write a blocked binary file with no record marks for SYSTEMS.

Initialize first hour for LØADS run

Read LØADS standard file file and prepare data.

Check error flag for input error and terminate if not equal to zero.

Initialize for run period, set beginning and ending dates. Run LØADS for stated period.



Set beginning and ending dates.

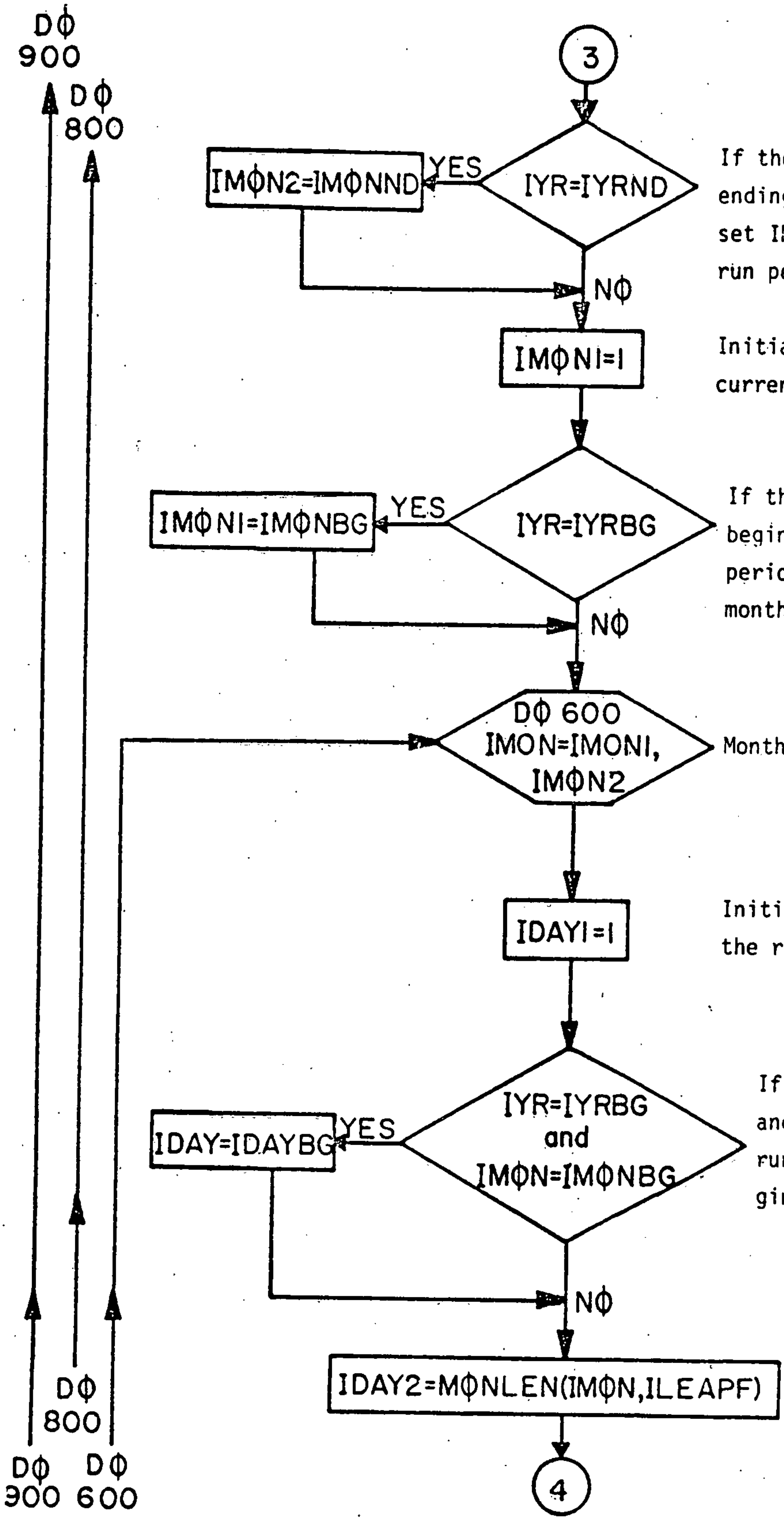
Instructs LØADS to loop through first day of run period three times to allow weighting factors to stabilize.

Yearly LØADS run loop.

Initialize leap year flag.

Check for occurrence of leap year, if so, set flag to one.

Initialize last month for current year.



If the current year is the ending year of the run period set IMON2 to ending month of run period.

Initialize first month for current year.

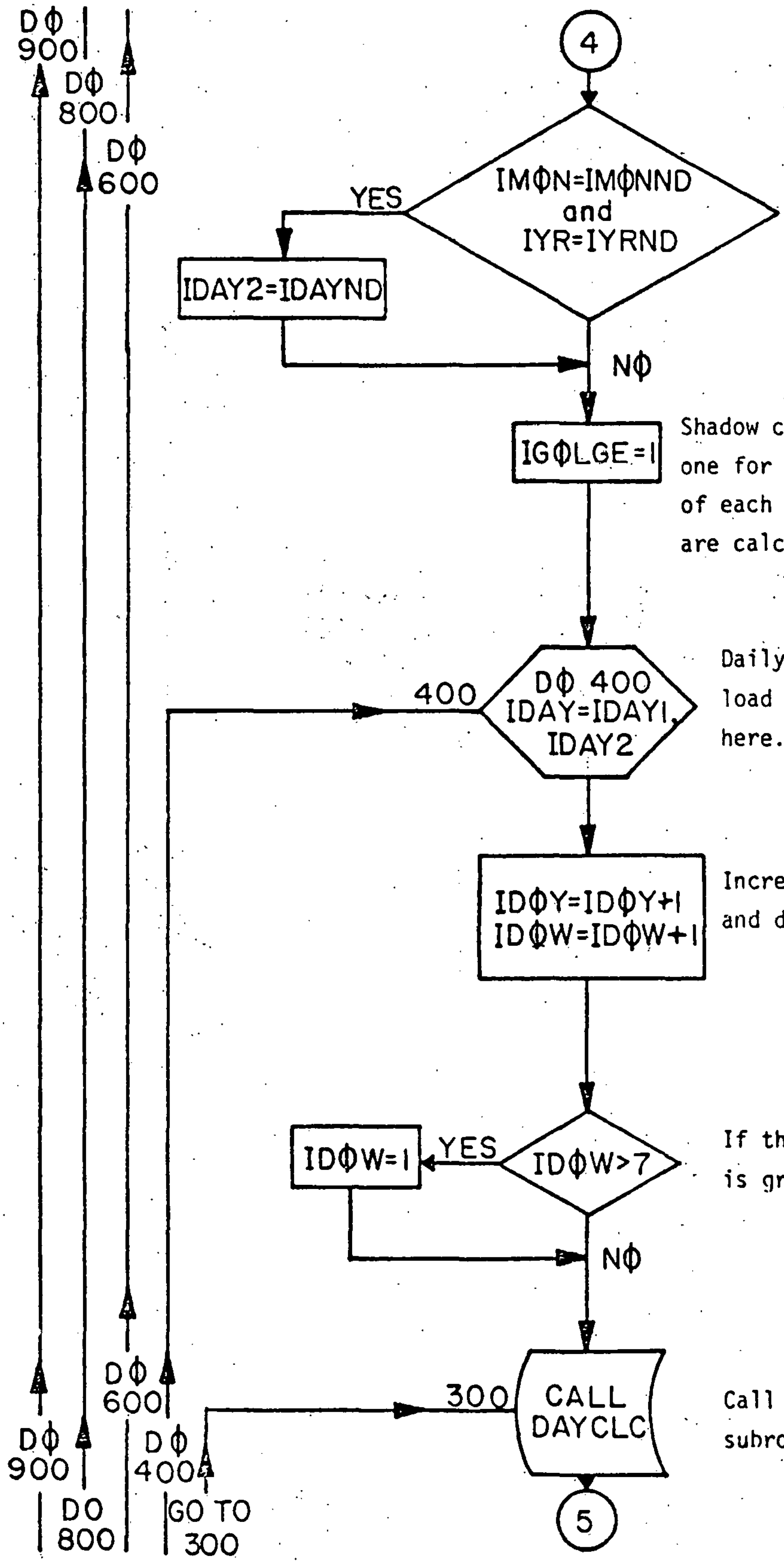
If the current year is the beginning year of the run period, set IMON1 to beginning month of run period.

Monthly LØADS run loop.

Initialize the first day of the run period for current month.

If it is the beginning year and beginning month of the run period, set IDAY to beginning day of run period.

The FUNCTION MONLEN will return the length of the month in days.



If it is the ending month and ending year of the run period, set IDAY2 to the ending day of the run period.

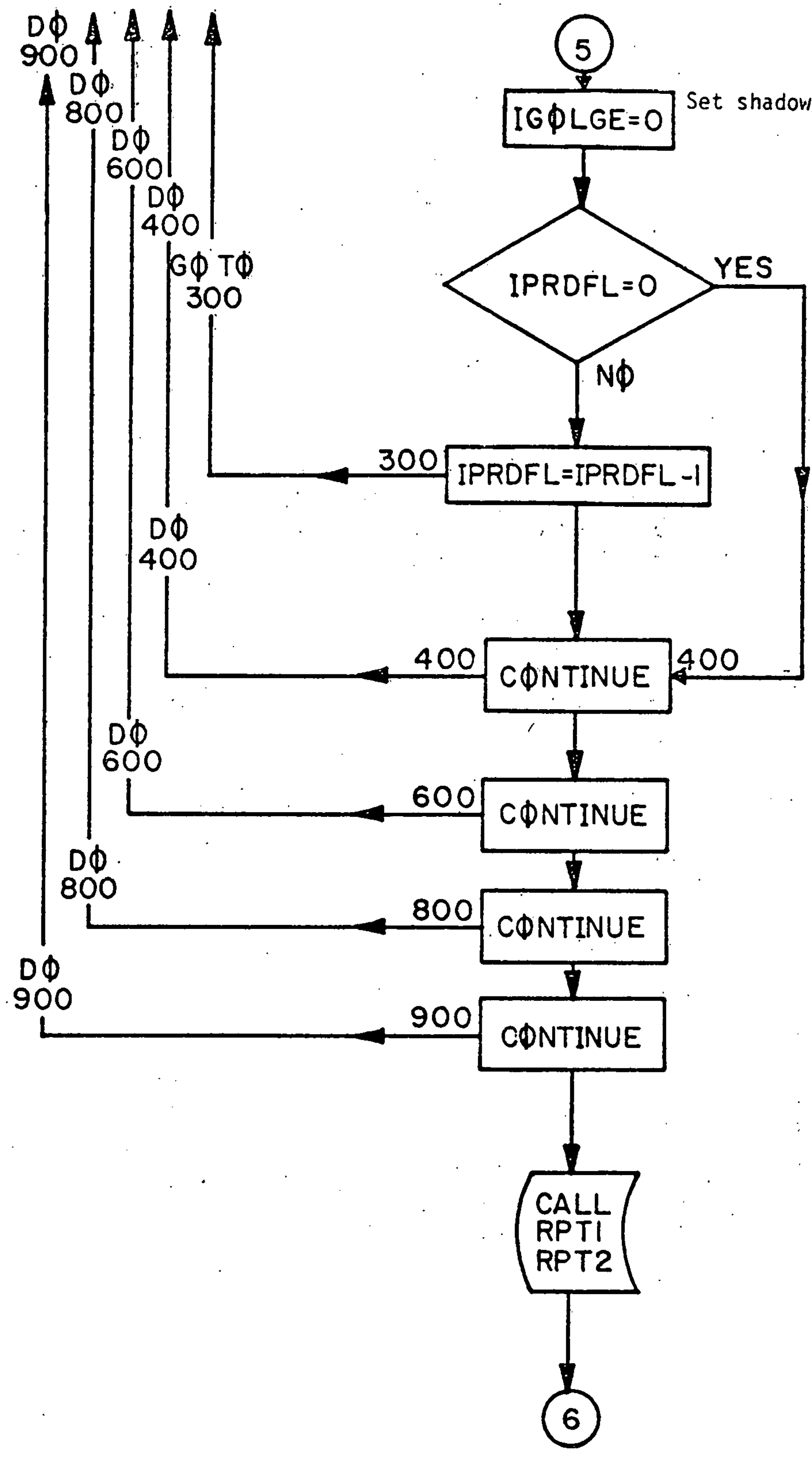
Shadow calculation flag equal to one for initial run and on first of each month. Shadow ratios are calculated only once per month.

Daily LOADS run loop. Hourly load calculations are performed here.

Increment the day of the year and day of the week counter.

If the day of the week counter is greater than seven, reset to one.

Call the daily loads calculation subroutine.



Set shadow calculation flag to zero.

Check to see if initial three-day run to stabilize weighting factors is complete.

Reduce counter by one.

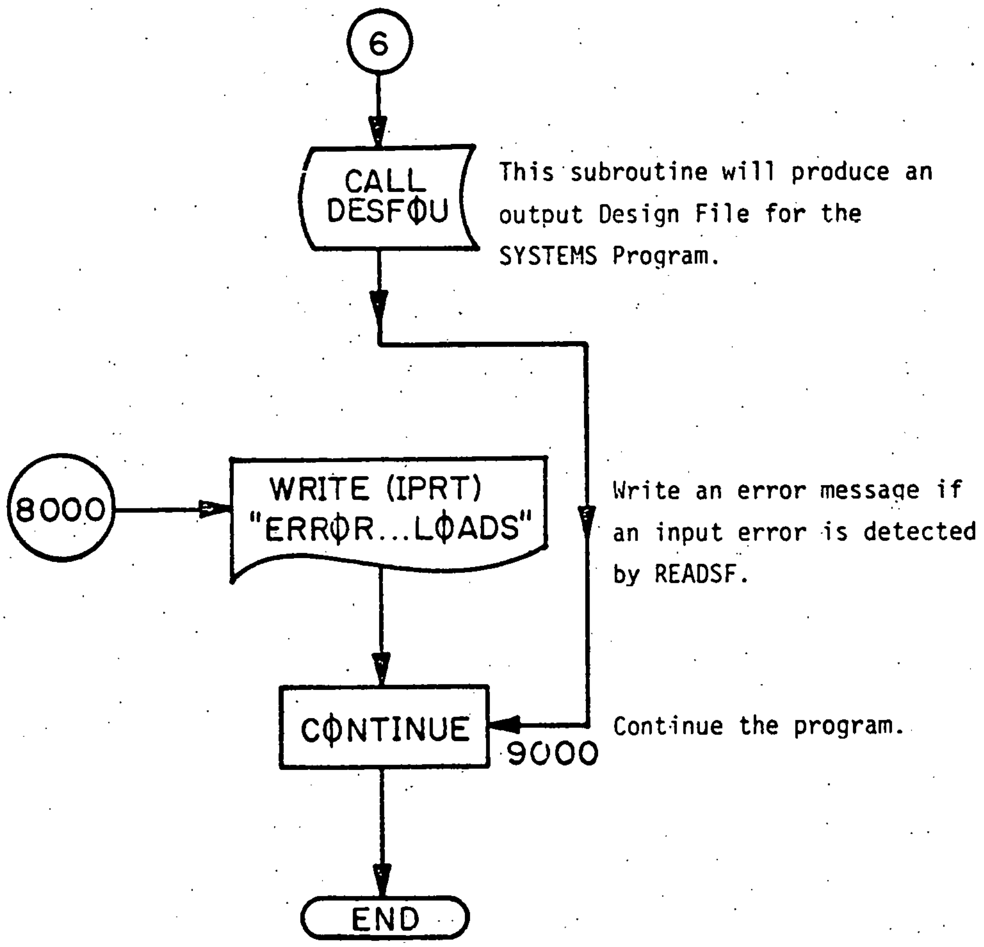
End of daily LOADS run loop.

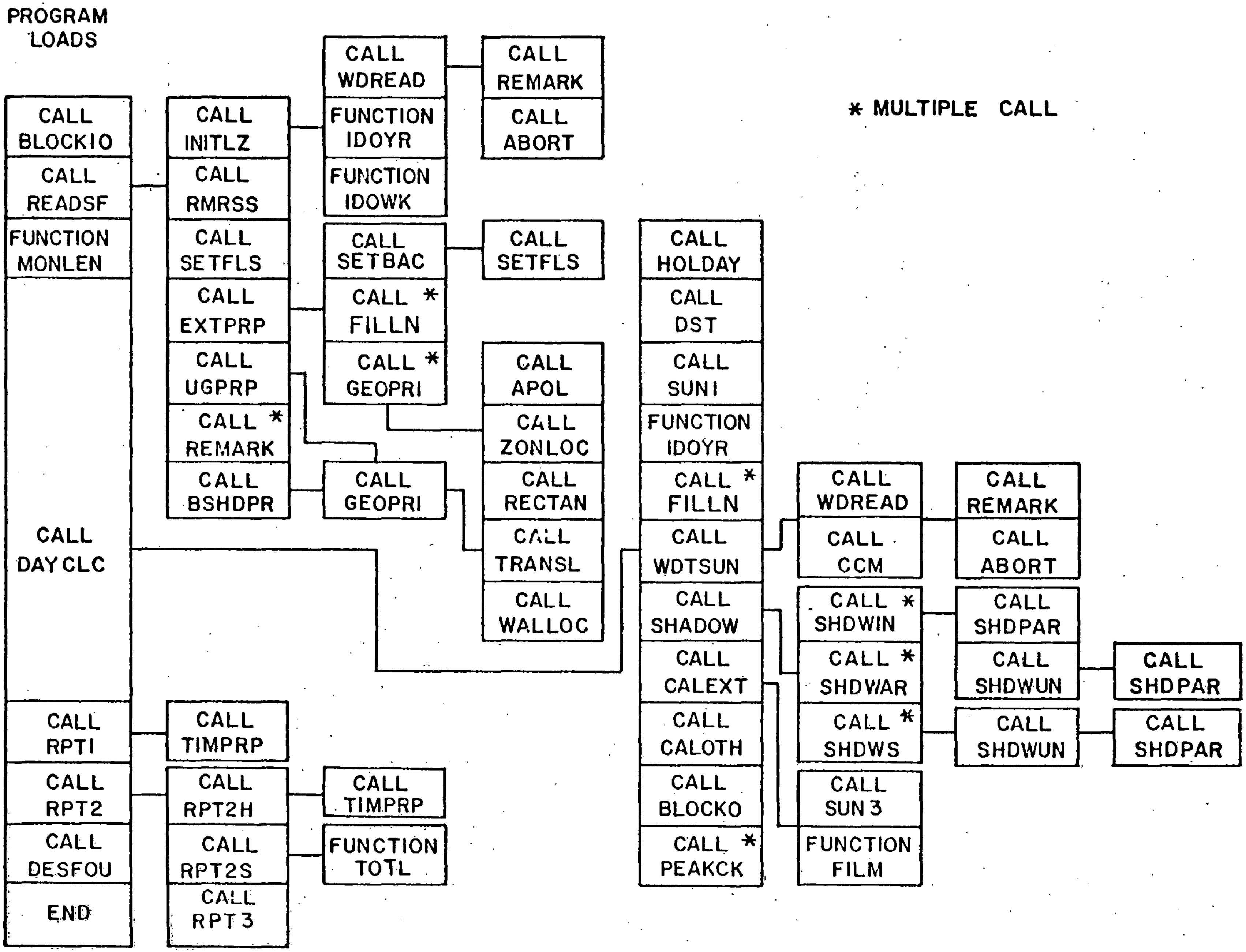
End of monthly LOADS run loop.

End of yearly LOADS run loop.

End of specified run period loop.

Call the LOADS output report subroutines.


6
CALL
DESFØU
This subroutine will produce an output Design File for the SYSTEMS Program.
8000
WRITE (IPRT)
"ERRØR...LØADS"
Write an error message if an input error is detected by READSF.
CØNTINUE
9000
Continue the program.
END



2. Cal-ERDA LOADS Subroutine Network.

3. Program LOADS Common Blocks.

Common Block	Definition of Contents
BLDPK	Contains information for building peak loads
BLDVA	Contains variables for total volume and area of building
CLOCK	Has the date and the hour, minute, second field as printed in the reports (2A40AL format)
CONST	Contains the constants used by the LOADS Program
DATE	Contains calendar and time data
DPTRS	Standard file data pointers
FILES	Names of files throughput to LOADS
FLAGS	Holiday and daylight savings time flags for international use
GPNTRS	Pointers to beginning of data blocks in standard file
INFPAR	Infiltration parameters
IWCPTR	IW pointers (used in READSF and DESFOU)
LOCALD	Contains the data pertinent to the locale of the building
PNTRS	Subpointers and lengths for zone data blocks
PTRBGN	Beginning pointers for each data block
REPORT	Contains data to be printed in reports
RUNPRD	The data for the length of the run period
RUNTIM	Beginning and ending dates of run period
SHADWF	Shadow subroutine call flags
SHDWC	Shadow data
SIZE	Contains building geometry data
SPCD	Contains the space peak loads and components
SUND	The hourly solar data, location, and intensity
SURFD	Contains surface data
TITLE	Title data
WEATHD	Local weather data
ZCOND	Total space conductance
ZNEWHR	Hourly space data
ZWALOC	Data for transforming surface data in building coordinates

Common Block	Contents
BLDPK	BLDPK(51)
BLDVA	BLDVOL, BLDARE
CLOCK	CLOCK(3)

CONST	DTOR, PIOVR2, PIOVR4
DATE	IYR, IMON, IDAY, IHR, IDOY, IDOW, ISCHR, ISCDAY, ILEAPF, IDSTF, IFSTHR, IPRDFL
FILES	ISTNDF, ISYSDS, ISYSHR, IWEATH, IPRT, IDEBUG
FLAGS	HOLFLG, DSTFLG
GPNTRS	MZONTB, NZONT, MXSTB, NXST, MGSHD, MSCTB, NSCT
INFPAR	PTWV, PSE
IWCPTR	MIWC1, MIWC2, NIWC, LIWC
LOCALD	ITIMZ, STALAT, SSTALA, CSTALA, TSTALA, STALON, BAZIM, SBAZIM, CBAZIM, BALTIT
PNTRS	MZ, MX, MI, MW, MF, MP, MA, MR, MG, MS, MY, MZLEN, MALEN, MPLEN, MRLEN, MULEN, MLAST
PTRBGN	IBSP, IBEW, IBIW, IBUG, IBAT, IBSH, IBPR, IBGE, IBSC, IBSV
REPORT	NREPRT, IREPRT(20)
RUNPRD	NRUNPR, IRUNPR(6,12), RUNSCA(12)
RUNTIM	IDAYBG, IMONGB, IYRBG, IDAYND, IMONND, IYRND
SHADWF	IGOLGE, ISDINF
SHDWC	NUXDIV, NUYDIV, DELTAX, DELTAY, AO, A, BO, B, SMALL, SMALP, ENDPNT(120,20), PRMPNT(120,20), NEP(120), XVERTF(20), YVERTF(20), ROTMAI(3,3), XO, YO, ZO, S, C
SIZE	LENGTH, LLLLLL, LSV, LSC, LPR, LGE, LSH, LAT, LUG, LIW, LEX, LSP, NSP, NEX, NSC, MZLN, MALN, MPLN, MRLN, MULN, MILN, ISZDUM(6)
SPCD	SPCD(51,50)
SUND	ISUNUP, GUNDOG, HORANG, TDECLN, EQTIME, SOLCON, ATMEXT, SKYDFF, RAYCOS(1), RAYCOS(2), RAYCOS(3), RDN, RTOT, RDIR, BSUN, BG, GAMMA, ETA
SURFD	NV, AREA, PERIM, AZIM, TILT, H, W
TITLE	ITITLE(8,5), NTITLE
WEATHD	IWDID(2), LWDID(2), LYR, LMON, LLAT, LLONG, LTIMZ, CLRNES, TGNDR, IPRECP, IWNDDR, ICLDTY, CLDAMT, WNDSPD, DBT, WBT, DPT, PATM, SOLRAD, DIRSOL, HUMRAT, ENTHAL, DENSTY, WNDDRR, CLDCOV, RDNCC, BSCC, SKYA, DBTR
ZCOND	ZCOND
ZNEWHR	TZONER, INFCOD, QDUVAR, QTAVAN, CAMCON, DDUVAR, QINTHT, QUGF, QPPLS, QUGW, QEQPS, QEQPS2, DTAVAN, QPPLL, QEQPL, QEQPL2, QINFL, SDIGER, SCAM, SLAMBA, QINFS, QLAMBA, CAMGUN QELECT, CFMINF, QODA, QDIGER, QZS, QZ1, QZTOT
ZWALOC	XZO, YZO, ZZO, AZO, SAZ, CAZ, XWO, YWO, ZWO, AWO, TWO, SAW, CAW, STW, CTW, XW, YW

4. LOADS Variable Master Index.

Variable Name	Variable Definition(s)
A	Maximum (XVERTF (i))/area (ft^2/azimuth angle, deg)
AA	General purpose data array
ADIFI	Inner absorption factor for diffuse solar radiation through glass
ADIFO	Outer absorption factor for diffuse solar radiation through glass
ADIRI	Inner absorption factor for direct solar radiation through glass
ADIRO	Outer absorption factor for direct solar radiation through glass
AGO	Weighting factor for solar heat gain through glass
AG1	Weighting factor for solar heat gain through glass
AISO	Weighting factor for heat gain caused by lights
AIS1	Weighting factor for heat gain caused by lights
AL	Solar altitude angle (radians)
ALARGP	Largest permeability value
AP	The j^{th} permeability point
APAREA	Area of a polygon (ft^2)
APAZIM	Azimuth angle of a polygon (radians)
APTILT	Tilt angle of a polygon (radians)
AP10	Diagonal of rotated polygon
AR	Unshaded area (ft^2)
AREA	Area of a surface (ft^2)
ARECI	Area of receiving polygon in shadow routines (ft^2)
AT	Local variable that contains both azimuth and tilt angle data
ATARE	Area of attachment (ft^2)
ATMEXT	Atmospheric extinction coefficient
ATXSM	Number of glass attachments
AUNSHA	Net area of attachment (ft^2)
AUNSH1	Net area of attachment (ft^2)
AWO	Azimuth angle (radians)
AXO	Weighting factor for heat gain by conductance through walls and floors
AX1	Weighting factor for heat gain by conductance through walls and floors
AZIM	Azimuth angle (radians)
AZO	Azimuth of space with respect to building coordinate system
AD	Minimum (XVERTF (i))

A32	Diagonal of a cube $(x^2 + y^2 + z^2)^{1/2}$
B	Maximum (YVERTF (i))
BALTIT	Building altitude (ft)
BAZIM	Building azimuth angle (radians)
BG	Ground reflectance (brightness of ground) (Btu/hr-ft^2)
BG1M	Weighting factor for solar heat gain through glass
BISIM	Weighting factor for heat gain caused by lights
BLDDTC	Temperature difference for building cooling load
BLDDTH	Temperature difference for building heating load
BLDPK	Building peak load summary (1-40)
BSCC	Brightness of sky with clouds (Btu/hr-ft^2)
BSUN	Brightness of sky without clouds (Btu/hr-ft^2)
BXIM	Weighing factor for heat gain through walls and floors
BO	Minimum (YVERTF (i))
C	Local variable, X-component of solar direction vector divided by Z-component
CA	Cosine of the azimuth angle, COS(AZIM)
CAMCON	Glass conductance (Btu/hr-ft^2)
CAMGUN	Solar gain to space through glass
CAW	COS(AWO)
CAZ	COS(AZO)
CBAZIM	Cosine of building azimuth angle, COS(BAZIM)
CC	Cloud-cover modifier coefficient
CD	Local variable, 1.-0.5 * TDECL2
CFM	Air flow (cfm)
CFMINF	Infiltration air flow (cfm)
CHCD	Local variable, COS(HORANG) * CD
CHCDSC	Local variable, CHCD * SSTALA
CLDCOV	Cloud-cover modifier coefficient, equivalenced to CC
CLOCK	Current values for time
CLRNES	Atmospheric clearness number
CLSD	Local variable, SD * CSTALA
CBTALA	Cosine of the latitude
CT	Cosine of the tilt angle
CTW	Cosine of the tilt angle, COS(TWO)
CWA	Cosine of wall azimuth angle, COS(WA)
C1	Local variable to convert to radians, COS(.01721*FLOAT (IDOY))

C2	Local variable, C1 * C1 - S1 * S1/weighted solar radiation effect
C3	Local variable, C1 * C2 - S1 * S2/weighted solar radiation effect
D	Setback/perimeter (ft)
DBT	Dry-bulb temperature (°F)
DBTC	Dry-bulb temperature at cooling load peak
DBTH	Dry-bulb temperature at heating load peak
DBTR	Dry-bulb temperature (°R)
DCA	Local variable for SETBAC calculations, D * CA
DDUVAR	Delayed wall load
DEBUGF	File name for variable IDEBUG
DELTAX	Strip width (ft)
DELTAY	Not currently used
DENSTY	Density (lb/ft^3)
DIRSOL	Not currently used
DNEW	Change in temperature history
DOTADJ	
DOTSLF	
DPT	Dew point temperature (°F)
DSA	Local variable for setback calculations, D * SA
DT	Temperature difference
DTAVAN	Delayed ceiling load
DTØR	Degree to radian conversion, 0.01745
ELCPWR	Not currently used
EN	The j^{th} endpoint of the i^{th} X-division
ENDPNT	The i^{th} strip, j^{th} Y-coordinate/boundary of rectangle
ENTH	Enthalpy of air (Btu/lb)
ENTHAL	Not currently used
EO	The previous value of EN
EQTIME	The solar equation of time (hrs)
ETA	Cosine of the solar angle of incidence
F	Not currently used
FI	Inside film coefficient for glass heat gain, (RO+RA) * U
FILMU	Heat transfer film coefficient ($Btu/hr\text{-}ft^2$ °F)
FO	Outside film coefficient for glass heat gain, RO * U

GAMMA	Cosine of the surface tilt angle
GUNDOG	The hour angle of sunrise and sunset (radians)
H	Height (ft)
HCT	Local variable for setback calculation, HH * CT
HH	Local variable for setback calculation, H + B
HORANG	Current hour angle of the sun (radians)
HRODA	Humidity ratio
HST	Local variable for setback calculation, HH * ST
HUMRAT	The humidity ratio
I	The ith unit or increment/packed binary word containing time data
IA	General purpose data array
IATT	The ith attachment
IBAT	Attachment data block beginning pointer
IBEW	External wall (EW) data block beginning pointer
IBGE	Geometry data block beginning pointer
IBIW	Internal wall (IW) data block beginning pointer
IBPR	Properties data block beginning pointer
IBSC	Schedule data block beginning pointer
IBSH	Shade data block beginning pointer
IBSP	Space data block beginning pointer
IBSV	Schedule vector data block beginning pointer
IBUG	Underground floors and walls (UGFW) data block beginning pointer
IC	Local variable in IDOWK, IY/100.
ICLD	Amount of cloud cover, equivalenced to CLDAMT
ICLDTY	Type of cloud, equivalenced to ICLTP
ICLTP	Type of cloud, equivalenced to ICLDTY
ID	The day of the month for report printing (1-31)
IDAY	The day of the month
IDAYBG	The beginning day of a LOADS run period
IDAYND	The ending day of a LOADS run period
IDAY1	First day of month or beginning day of run period
IDAY2	Last day of month or ending day of run period
IDEBUG	Debugging variable
IDIF	Local variable for counter, LDOY-IDOY + 1
IDOW	Day of the week
IDOY	Day of the year

IDSTF	Daylight savings time flag
IE	The $i^{\underline{th}}$ external surface
IERRF	Error flag
IERRFL	Error flag
IEXT	The $i^{\underline{th}}$ external surface in a zone
IE2	The $n^{\underline{th}}$ external surface
IFLG	Vertex translation flag (i.e., wall, attachment, shade)
IFMT	Not currently used
IFSTHR	First-hour flag
IGOLGE	Shading calculation flag, 1 = calculate shade ratios
IH	The hour of the day (1-24) for report printing
IHR	The current hour
IL	Light fixture type
ILEAPF	Leap-year flag, 1 = leap year
IM	The month of the year for report printing
IMON	The current month of the year
IMONBG	Beginning month of a LOADS run period
IMONND	Ending month of a LOADS run period
IMON1	January or beginning month of a run period
IMON2	December or ending month of a run period
IN	Packed binary word containing month, day, and hour data
INAM	Two words (names) of zone identification, (1) + (2)
INC	Local variable in setback calculation, 3 * (MHLEN+4*3+1) + 1
INFCOD	Infiltration method code, 1 = air change, 2 = crack
IOGLE	Label for one of the four quarters of a day (AM, NOON, PM, MIDNIGHT)
IOUT	Output array in TIMPRP containing report time data
IPRDFL	Flag to initialize each run by looping through DAYCLC three times to establish weighting factors
IPRECP	Not currently used
IPRT	Variable name for contents of LOADS output file
IR	Property name (i.e., absorption)/incremental counter
IREPRT	Variable name for values to be printed in reports
IRN	The number of LOADS run
IRUNPR	The $i^{\underline{th}}$ run period
IS	Exterior surface type index

ISC	The ith schedule
ISCDAY	Day of the week for schedule selection
ISCHR	DST corrected hour, IHR + IDST
ISDINF	Shading flag, 0 = surface facing away from sun
ISHADE	
ISHD	The ith shading surface
ISIZ	Dimensioned array in READSF containing common block /SIZE/
ISPCD	Zone identification (name)
ISTAT	Not currently used
ISTNDF	The standard file
ISUB	Function name
ISUNUP	Sun up flag
ISYSDS	LOADS output design file for SYSTEMS, variable name
ISYSHR	LOADS output hourly data for SYSTEMS, variable name
ISZDUM	Not currently used
IT	Not currently used
ITIM	Time data for reports
ITIMZ	Time zone
ITITLE	Five titles of eight words each, ITITLE(8,5)
IU	The ith underground (UG) surface
IW	Type of construction
IWDID	Standard file I.D. for weather data station (name)
IWEATH	Variable name for contents of weather file
IWNDDR	Wind direction
IW1	Not currently used
IY	Local variable, the year
IYR	The current year
IYRBG	Beginning year of LOADS run
IYRBGG	Beginning year of LOADS run
IYRND	Ending year of LOADS run
IYRNDD	Ending year of LOADS run
IZ	Zone number
IZONE	The ith zone
I2	Local variable, number of response factors minus one

J	Local variable in shadow routine, MX + IHR
JAY	Day of the month, equivalenced to IDAY
JJ	Incremental counter
JOL	Holiday flag
J2	Incremental counter
K	Incremental counter, the i^{th} attachment
KOSE	Number of vertices
KOSEF	Number of vertices
KS	Schedule data pointer (schedule name)
L	Data field length
LAT	Length of attachment vector data
LDAY	Local variable for IDOWK
LDBGCT	Debug-print flag
LDOY	Local variable in WDREAD to locate proper point on weather tape
LENGTH	Length of second standard file record
LEX	Length of external wall vector data
LFN	Not currently used
LGE	Length of geometry vector data
LIW	Length of internal wall vector data
LIWC	Number of words of IW data
LLAT	Not currently used
LLLLLL	Not currently used
LLONG	Not currently used
LMON	Not currently used
LPR	Length of properties vector data
LSC	Length of schedule vector data
LSH	Length of shade vector data
LSP	Length of space vector data
LSV	Length of schedule vector data (see LSC)
LTIMZ	Not currently used
LUG	Length of UGFW vector data
LUND	
LWDID	Not currently used
LXFER	Not currently used
LYR	Not currently used

M	Local variable for month in IDOWK
MA	Attachment data pointer (number of attachments)
MALEN	Attachment data entry length (length of MA)
MALN	Attachment data entry length (length of MA)
MD	Name of data block containing Julian date for first day of each month
MDBTC	Dry-bulb temperature for cooling load, integer
MDBTH	Dry-bulb temperature for heating load, integer
MF	Underground floor (UGF) surface pointer
MG	Geometry data pointer (number of vertices)
MGS	Shade geometry data pointer
MGSHD	Building shade data pointer in standard file
MH	Shading surface data pointer (number of shading surfaces)
MHA	Shading surface data pointer (number of shading surfaces)
MHLEN	Shading surface data entry length (length of MH)
MI	Internal surface pointer (number of internal surface)
MILN	Internal surface data entry length (length of MI)
MIWC1	Beginning location for IW information in AA array.
MIWC2	Ending location for IW information in AA array.
ML	Array containing the number of days in each month
MLAST	Last data pointer in common block
MO	Month of the year, equivalenced to IMON
MP	Properties data pointer (name)
MPLEN	Construction data entry length for each space attachment (length of MP)
MPLN	
MPLN1	Three times number of response factors
MR	Glass properties data pointer (name)
MRLEN	Glass properties data entry length (length of MR)
MRLN	Glass properties data entry length (length of MR)
MS	Schedule data pointer
MSCTB	Schedule table beginning pointer
MU	UGWF, data pointer (number of space UG surfaces)
MULN	UGWF data entry length (length of MU)
MW	Not currently used
MWBTC	Wet-bulb temperature for cooling load (integer)

MWBTH	Wet-bulb temperature for heating load (integer)
MX	External surface pointer (number of external surfaces in zone)
MXSTB	External surface table beginning pointer
MZ	Zone data pointer (name)
MZLEN	Space data entry length for each space attachment (length of MZ)
MZLN	Space data entry length for each space attachment (length of MZ)
MZONTB	Zone table beginning pointer
MZ1	
M10TBL	Daylight savings time flag data block for month of October
M4TBL	Daylight savings time flag data block for month of April
N	The number of i (N particles)
NA	Not currently used
NATT	Number of attachments (MA)
NDAY	Day of the week, equivalenced to IDOW
NE	
NEP	Number of boundary or endpoints
NEQ	Number of boundary or endpoints of the i^{th} area
NEX	Number of external surface blocks
NEXT	Number of external surfaces
NEXTS	Number of external surfaces in zone
NH	
NIW	Number of internal walls
NIWC	Number of internal walls
NREPRT	Not currently used
NRESF	Number of response factors
NRUNPR	Number of run periods
NS	
NSC	Number of schedules
NSCB	Current schedule vector pointer
NSCT	Number of schedules
NSHADE	Number of shading surfaces
NSHD	Number of shading surfaces
NSP	Number of zones (spaces)
NSUG	Number of underground space surfaces
NT	

NTITLE	Number of titles in use
NUXDIV	Number of X-divisions
NUYDIV	Number of Y-divisions
NV	Number of vertices
NVS	Number of vertices for shading surface
NXDIV	Number of X-divisions
NXST	Number of external surface blocks
NZONT	Number of zones
ONE	
ORNEW	
OUTPUT	LOADS outfile name (IPRT)
P	
PATM	Atmospheric pressure (in. Hg)
PCO	Infiltration coefficient accounting for wind direction, wind speed, and neutral pressure zone
PERCT	Per cent of heat, generated by lights, that remains in the space
PERIM	Perimeter (ft)
PIOVR2	Constant, Pi/2, 1.5707963
PIOVR4	Constant, Pi/4, 0.78539816
PK	The building peak load
PM	
PPA	People activity level *0.01
PPN	Number of people
PPWV	Partial pressure of water vapor
PRMPNT	Permeability end point, lowest and highest
PROJ	The projection of a polygon
PSE	Zone height to neutral pressure zone
PTWV	Infiltration coefficient, 0.000482*(1.153*VEL)**2 or 0.0006407849 * WNDSPD * WNDSPD
PUND	
P1	Local variable for calculating partial pressure of water vapor
P2	Local variable for calculating partial pressure of water vapor
P3	Local variable for calculating partial pressure of water vapor
P4	Local variable for calculating partial pressure of water vapor
Q	Value for simple load-type calculation, U * A * DT
QABS	Solar radiation absorbed by glass surface

QCMAX	Maximum cooling load
QCON	Conduction heat gain through glass surface
QDIF	Diffuse solar radiation striking a glass surface
QDIGER	Current hour heat flux through walls and floors to be weighted
QDIR	Direct solar radiation striking a glass surface
QDUVAR	Quick wall load
QELECT	Electric power demand
QEQPL	Equipment, latent heat
QEQPL2	Process, latent heat
QEQPS	Equipment, sensible heat
QEQPS2	Process, sensible heat
QGUNES	Total solar radiation acquired by glass surface
QHMAX	Maximum heating load
QINFL	Infiltration, latent heat
QINFS	Infiltration, sensible heat
QINTHT	Internal wall load
QLAMBA	Current hour lighting load to be weighted
QODA	Sum of all weighted loads
QPLEN	Weighted plenum load (Btu/hr)
QPPLL	People, latent heat
QPPLS	People, sensible heat
QQ	Zone lighting load
QTAVAN	Quick ceiling load
QTRANS	Solar radiation transmitted by glass surface
QUGF	Underground floors load
QUGW	Underground walls load
QYENI	Quick wall load
QZL	Latent heat zone load (Btu/hr)
QZS	Sensible heat zone load (Btu/hr)
QZTOT	Total zone load (Btu/hr)
RA	Glass heat transfer coefficient (Btu/hr-ft^2°F)
RAYCOS	Directional cosines of the sun (3)
RCPM	Weighting factor for immediate heat gain to plenum
RCPO	Weighting factor for immediate heat gain to plenum
RCP1	Weighting factor for immediate heat gain to plenum
RDIF	Diffuse solar radiation (Btu/hr-ft^2)
RDIR	Direct beam solar radiation on a surface (Btu/hr-ft^2)

RDN	Direct normal beam radiation on a clear day (Btu/hr-ft^2)
RDNCC	Direction normal beam radiation on a cloudy day (Btu/hr-ft^2)
RI	Inside heat transfer coefficient for glass (Btu/hr-ft^2°F)
RMRG	Weighting factor for solar gains through glass
RMRSI	Weighting factor for gains caused by lighting
RMRX	Weighting factor for gains caused by conduction
ROTMAT	Rotation matrix, ROTMAT (3,3)
RTOT	Total solar radiation (Btu/hr-ft^2)
RUNSCA	Not currently used
RO	Outside heat transfer coefficient for glass (Btu/hr-ft^2°F)
S	Local variable, Y-component of solar direction vector divided by the Z-component
SA	Sine of azimuth angle, SIN(AZIM)
SAW	SIN(AWO)
SAZ	SIN(AZO)
SBAZIM	Sine of building azimuth angle, SIN(BAZIM)
SCAM	Weighted solar gain through glass
SD	Local variable, TDECLN *(1.-0.1667*TDCL2)
SDELTA	External surface orientation for infiltration calculations
SDIGER	Loads after weighting
SHCD	Local variable, SIN(HORANG)*CD
SKYA	Sky view factor, 2 *(10-CLDAMT)
SKYDFF	Sky diffusivity
SLAMBA	Weighted lighting load
SMALL	Local variable, DELTAX * 0.000001
SMALP	Smallest permeability
SOLCON	Extraterrestrial solar constant (Btu/hr-ft^2)
SOLI	Amount of solar radiation striking a shaded wall
SOLRAD	Not currently used
SPCD	Space peak load for the i^{th} space (Btu/hr)
SQ	Space of ICLD
SSTALA	SIN(STALAT)
ST	Sine of the tilt angle
STALAT	Station latitude (radians)
STALON	Station longitude (radians)
STNDFL	Standard file

STW	SIN(TWO)
SUMATT	Sum of attachments, net area for shading
SUMXDT	Summation of temperature difference in X-direction after application of response factors
SUMYDT	Summation of temperature difference in Y-direction after application of response factors
SWA	Sine of surface (wall) azimuth angle
SWT	Sine of surface (wall) tilt angle
SYSDES	SYSTEMS design file
SYSHRL	SYSTEMS hourly loads file
S1	Local variable, SIN(.01721*FLOAT(IDOY))
S2	Local variable, 2. * S1 * C1
S3	Local variable, CL * S2 + S1 * C2
T	Temperature (°F/tilt angle, deg)
TDECLN	Tangent of the solar declination angle
TDECL2	Square of TDECLN
TDIF	Transmission factor for diffuse solar radiation through glass
TDIR	Transmission factor for direct solar radiation through glass
TEMP	Ambient temperature (°F)
TGNDR	Ground temperature (°F)
TILT	Tilt angle (radians)
TNEW	Temperature history (°F)
TOLD	Temperature history (°F)
TSOL	Effective solar temperature of external wall (°F)
TSTALA	Tangent of the latitude
TWO	Tilt angle (radians)
TZONER	Zone calculation temperature (°R)
TZ1	Zone temperature for heating (°F)
T1	
T2	
T3	
U	Surface heat transfer factor (U) (Btu/hr-ft^2)
UATT	Attachment U-factor
UXS	External surface U-factor
V	Wind speed (mph)
VOL	Volume (ft^3)

W	Width/weight of floor in RMRSS (ft or lb/ft^2)
WA	Surface (wall) azimuth angle (radians)
WBT	Wet-bulb temperature (°F)
WBTC	Wet-bulb temperature at peak cooling load
WBTH	Wet-bulb temperature at peak heating load
WCA	Local variable for setback calculation, WW * CA
WEATHR	File name for weather file
WNDDRR	Wind direction
WNDSPD	Wind speed
WSA	Local variable for setback calculation, WW * SA
WT	Surface (wall) tilt angle (radians)
WW	Local variable for setback calculation, W + B + B
W1	
X	X-coordinate
XCOMP	X-component of polygon
XF	X-coordinate of the i^{th} vertex of shadow polygon
XH	
XL	X-coordinate of the n^{th} vertex of shadow polygon
XMUL	
XO	X-coordinate of n^{th} vertex of polygon
XP	
XP1O	Difference in X-coordinate between origin and first vertex
XS	X-component of solar direction vector
XSMULT	Shade multiplier
XSQCMP	
XSXCMP	
XT	
XUNSHA	Gross area (wall) (ft^2)
XV	
XVERTF	X-vertex, rotated receiving polygon
XW	X-coordinate of attachment with respect to EW coordinate system
XWO	X-coordinate of EW with respect to space coordinate system
XX	
XYZ	Vertex location data array
XZO	X-coordinate of building with respect to building coordinate system

X0	Rectangle translation distance in X-direction
X1	X-coordinate of $i^{\underline{th}}$ vertex of polygon
X2	X-coordinate
X32	Difference in X-coordinate between second and third vertex
Y	Y-coordinate
YCOMP	Y-component of polygon
YF	
YH	
YL	
YO	Y-coordinate of $n^{\underline{th}}$ vertex of polygon
YP	
YP10	Difference in Y-coordinate between origin and first vertex
YS	Y-component of solar direction vector
YT	Y-coordinate of $i^{\underline{th}}$ vertex of polygon
YV	
YVERTF	Y-vertex, rotated receiving polygon
YW	Y-coordinate of attachment with respect to EW coordinate system
YWO	Y-coordinate of EW with respect to building coordinate system
YZO	Y-coordinate of building with respect to building coordinate system
Y0	Rectangle translation distance in Y-direction
Y1	
Y2	
Y32	
Z	Z-coordinate/local variable for temperature conversion, 273.16/T
ZCOMP	Z-component of polygon
ZCOND	Total space conductance (Btu/hr)
ZERO	
ZL	
ZMULT	Zone multiplier
ZO	Z-coordinate of $n^{\underline{th}}$ vertex of polygon
ZP	
ZP10	Difference in Z-coordinate between origin and first vertex
ZS	Z-component of solar direction vector
ZUGFUA	Value of heat transfer function (UA) for underground floors (Btu/hr-ft^2)

ZUGWUA	Value of heat transfer function (UA) for underground walls (Btu/hr-ft^2)
ZWO	Z-coordinate of EW with respect to building coordinate system
ZZO	Z-coordinate of space with respect to building coordinate system
Z1	Z-coordinate of $i^{\underline{th}}$ vertex of polygon
Z2	Z-coordinate
Z32	Difference in Z-coordinate between second and third vertex

C. LOADS Subroutine Descriptions

This subsection contains descriptions of each of the subroutines and functions used by the Cal-ERDA LOADS Program. Each subroutine or function write-up contains the following:

(1) Subroutine call statement

(2) Brief description of the purpose of the subprogram

(3) Location of calls to the subprogram being described

(4) Calls to other subprograms by the subprogram being described

(5) Common Blocks

a. List of Common Blocks employed by the subprogram

b. List of the variables in each Common Block that are used by the subprogram

c. List of the variables that are placed in the Common Block by the subprogram

(6) Declaration statements made by the subprogram. These include dimension statements, data statements, real and integer statements, equivalence statements, etc.

(7) A list of the required inputs and their definitions

(8) A list of the outputs and their definitions

(9) The calculation procedure of the subprogram

(10) ASHRAE verification or origin of methodology of each subprogram. A table shows the important Cal-ERDA variables in each subprogram and their counterparts and location in the NECAP program (Ref. 3) and in ASHRAE literature (Ref. 1). An equivalence statement and a discussion of algorithm differences are also provided.

Subroutine Name	Description
ABORT	An operating system subroutine that aborts the program if a weather tape error is found
APOL	Calculates the polygon area, tilt angle, and azimuthal angle
BLOCKI	Reads blocked binary file
BLOCKIO	Enables blocked binary file input/output
BLOCKO	Writes the blocked binary file
BLOCKR	Writes record mark
BSHDPR	Prepares the building shade data
CALEXT	Contains building external surface calculations
CALOTH	Contains calculations for other building surfaces
CCM	Hourly cloud-cover modifier
DAYCLC	Daily loads calculations
DESFOU	Provides the hourly loads output file for SYSTEMS

DST	Calculates daylight savings time flag
DUMP	An operating system subroutine for debugging that dumps a desired core content
GEOPR1	Prepares building geometry data
HOLDAY	Indicates the national holidays of the United States
IDOWK	Calculates the day of the week
IDOYR	Calculates the day of the year
INITLZ	Intializes the LOADS run
LOADS	Main program routine
MONLEN	Calculates the length of a given month
PEAKCK	Checks for peak heating and cooling loads
READSF	Reads the standard file
RECTAN	Transcribes two-dimensional rectangular coordinates in XYZ
REMARK	An operating system subroutine used in lieu of a write statement
RMRSS	Computes weighting factors
RPT1	LOADS output report No. 1
RPT2	LOADS output report No. 2
RPT2S	LOADS report for individual components of sensible and latent heat
SETBAC	Calculations for setback surfaces
SETFLS	An operating system subroutine used to set field lengths
SHADOW	Calculates shading on external building surfaces
SHDPAR	Calculates the area of a polygon in shadow computations
SHDWAR	Calculates the unshaded area
SHDWIN	Initializes shadow calculation arrays with receiving polygon
SHDWPR	Prints the shadow picture
SHDWS	Processes shading surfaces and clips portions of shading polygons below the receiving polygon
SHDWUN	Unions of polygons for shading
SUN1	Calculates daily solar data
SUN3	Calculates hourly incident solar radiation for a particular surface orientation
TIMPRP	Prepares time data for LOADS reports
TOTL	Sums any set of values from one to N
TRANSL	Translates the vertices XYZ(...) by the values XO, YO, and ZO
TYPLES	Returns given argument without conversion of type
UGPRP	Processes underground space data
WALLOC	Locates wall position in zone
WDREAD	Reads Cal-ERDA weather file

WDTSUN	Calculates hourly weather variables and available solar radiation
ZONLOC	Locates zone data in global coordinates

1. Subroutine APOL.

SUBROUTINE APOL (N, XYZ, APAREA, APAZIM, APTILT)

Description

This subroutine calculates the surface area, tilt angle, and azimuth angle given the X,Y,Z coordinates of a polygon of known vertices.

Subprograms Calling This Subroutine

SUBROUTINE GEOPR1

Subprograms Called By This Subroutine

None

Common Blocks

None

Declarations

DIMENSION XYZ (3,N)

Input

Name	Description
N	Number of vertices
XYZ(3,N)	The coordinates of the vertices (ft)

Output

Name	Description
APAREA	Area of the polygon (ft^2)
APAZIM	Azimuth angle (radians)
APTILT	Tilt angle (radians)

Calculation Procedure

1. Dimension the coordinate input data array XYZ

   DIMENSION XYZ (3,N) ,

   where N = number of vertices.

2. Initialize the X, Y, and Z components of the polygon.

   XCOMP = 0.0
   YCOMP = 0.0
   ZCOMP = 0.0

3. Set the starting point to be the nth vertex of the polygon.

   XO = XYZ (1,N)
   YO = XYZ (2,N)
   ZO = XYZ (3,N)

4. Calculate the X, Y, and Z components of the polygon by using a DO loop.

   ```
   DO 200 I = 1,N
   ```

5. Get the X, Y, and Z coordinates of the $i^{\underline{th}}$ vertex from the input data array.

   ```
   X1 = XYZ (1,I)
   Y1 = XYZ (2,I)
   Z1 = XYZ (3,I)
   ```

6. Determine the X, Y, and Z components of the polygon.

   ```
   XCOMP = XCOMP + YO*Z1 - Y1*ZO
   YCOMP = YCOMP + ZO*X1 - Z1*XO
   ZCOMP = ZCOMP + XO*Y1 - X1*YO
   ```

7. Increment the XO, YO, and ZO terms for the next iteration and continue.

   ```
   XO = X1
   YO = Y1
   ZO = Z1
   200 CONTINUE
   ```

8. Calculate the area, tilt angle and projection of the polygon

   ```
   A = SQRT (XCOMP*XCOMP + YCOMP*YCOMP + ZCOMP*ZCOMP)/2.
   TILT = ACOS (ZCOMP/(2.*A))
   PROJ = SQRT (XCOMP*XCOMP + YCOMP*YCOMP)
   ```

9. Compare the projection and the area of the polygon. If PROJ << A then AZIM = 0.0; but, if projection is appreciable compared to A, use the proper equation according to the signs of XCOMP and YCOMP. Table III.1 lists the equations and the actual coding is next.

TABLE III.1

EQUATIONS FOR THE CALCULATION OF AZIM

		SIGN OF XCOMP	
		-	0 or +
SIGN OF YCOMP	-	$\Pi + \sin^{-1}$ (-XCOMP/PROJ)	$\Pi/2 + \sin^{-1}$ (-YCOMP/PROJ)
	0 or 1	$1.5\Pi + \sin^{-1}$ (YCOMP/PROJ)	$\sin^{-1}$ (XCOMP/PROJ)

```
IF(PROJ - .0001*A)  110, 110, 120
110 AZIM = 0.0
GO TO 900
120 IF (XCOMP)160, 130, 130
130 IF (YCOMP)150, 140, 140
```

```
140 AZIM = ASIN (XCOMP/PROJ)
GO TO 900
150 AZIM = 1.5708 + ASIN (-YCOMP/PROJ)
GO TO 900
160 IF (YCOMP) 170, 180, 180
170 AZIM = 3.1416 + ASIN (-XCOMP/PROJ)
GO TO 900
180 AZIM = 4.7124 + ASIN (YCOMP/PROJ)
900 CONTINUE
```

10. Set the output variables and terminate.

```
APAREA = A
APAZIM = AZIM
APTILT = TILT
RETURN
END
```

ASHRAE Verification

There is no equivalent ASHRAE algorithm (Ref. 1) for the subroutine APOL. It is, however, virtually identical to NECAP subroutine APOL,and a further description of the procedures may be found in the NECAP documentation (Ref. 3). Table III.1, for example, is taken from this manual.

2. Subroutine BSHDPR.

Description

This subroutine prepares the building shade data for use by the LOADS Program. It does not perform shade effect calculations.

Subprograms Calling This Subroutine

SUBROUTINE READSF

Subprograms Called By This Subroutine

SUBROUTINE GEOPR1

Common Blocks	Variables Obtained from Common Block	Variables Placed in Common Blocks
/CONST/	DTOR	None Placed
/GPNTRS/	MGSHD	MGSHD
/LOCALD/	Not currently used here	
/PNTRS/	None obtained	MG, MH
/PTRBGN/	IBSH, IBGE	None Placed
/SIZE/	Not currently used here	

/SURFD/	None Obtained	H, W
/ZWALOC/	SAZ, CAZ, XWO, YWO, ZWO, AWO, TWO, SAW, CAW, STW, CTW	XZO, YZO, ZZO, AZO

Declarations

```
COMMON AA(1000)
INTEGER IA(1000)
EQUIVALENCE (IA(1), AA(1))
```

Input

Name	Description
MGSHD	Building shade data pointer in standard file
IBSH	Shade data block beginning pointer
IA(MH)	Number of shading surfaces
IA(MH+2)	Shade geometry data pointer
IBGE	Geometry data block beginning pointer
AA(MG+1)	X-coordinate and/or height
AA(MG+2)	Y-coordinate and/or width
AA(MG+3)	Z-coordinate and/or azimuth and tilt angles
DTOR	Conversion factor for degrees to radians

Output

Name	Description
XWO	X-coordinate of external wall
YWO	Y-coordinate of external wall
ZWO	Z-coordinate of external wall
H	Height (ft)
W	Width (ft)
AWO	Azimuth angle (radians)
TWO	Tilt angle (radians)
SAW	Sine of azimuth angle
CAW	Cosine of azimuth angle
STW	Sine of tilt angle
CTW	Cosine of tilt angle

Calculation Procedure

1. Check to see if shade data is available,

```
IF (MGSHD .EQ. 0) GO TO 9000,
```

where 9000 is the statement number for RETURN.

2. Initialize variables.

```
XZO = YZO = ZZO = AZO = 0.
SAZ = 0.
CAZ = 1.
```

Note that these variables are not used by this subroutine.

3. Locate the shade data.

```
MGSHD = MGSHD + IBSH
MH = MGSHD
```

4. Determine the number of shading surfaces for which data is available.

```
NSHD = IA(MH)
```

5. Produce the required output shade data for each shade (NSHD) using a DO loop.

```
DO 2000 ISHD = 1, NSHD
MG = IA(MH+2) = IA(MH+2) + IBGE
```

6. Obtain the X, Y, and Z coordinates from the AA array.

```
XWO = AA(MG+1)
YWO = AA(MG+2)
ZWO = AA(MG+3)
```

7. Increment the MG pointer for next set of data.

```
MG = MG + 3
```

8. Obtain the height, width, and azimuth/tilt angle from the incremented AA array.

```
 H = AA(MG+1)
 W = AA(MG+2)
AT = AA(MG+3)
```

where AT is a local variable containing both the azimuth and tilt angle data in combination.

9. Reset the AA array MG pointer.

```
MG = MG - 3
```

10. Differentiate the azimuth and tilt angle data.

```
T = AMOD (AT, 1000.)
A = (AT -T)/1000.,
```

where

T = Tilt angle (deg) and
A = Azimuth angle (deg).

11. Convert the azimuth and tilt angle from degrees to radians and calculate their respective sines and cosines.

```
AWO = A*DTOR
TWO = T*DTOR
SAW = SIN(AWO)
CAW = COS(AWO)
```

```
STW = SIN(TWO)
CTW = COS(TWO)
```

12. Call the geometry data preparation subroutine GEOPR1 and set the mode flag, IFLG, equal to two for shade data.

```
CALL GEOPR1 (2)
```

13. Increment the shade data pointer and return.

```
MH = MH + 2
2000 CONTINUE
9000 CONTINUE
RETURN
END
```

ASHRAE Verification

There is no equivalent ASHRAE algorithm for subroutine BSHDPR.

3. Subroutine CALEXT.

Description

The Cal-ERDA subroutine CALEXT performs the thermal load calculations for all external surfaces, both quick and delayed, including solar gains and infiltration.

Subprograms Calling This Subroutine

SUBROUTINE DAYCLC

Subprograms Called by This Subroutine

SUBROUTINE SUN3

FUNCTION FILM

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/CONST/	PIOVR2, PIOVR4	None Placed
/DATE/	IHR	None Placed
/INFPAR/	PTWV, PSE	None Placed
/PNTRS/	MX, MP, MA, MR, MALEN	MX, MP, MA, MR
/SUND/	GUNDOG, HORANG, RTOT, RDIR, BG, GAMMA, ETA	GAMMA
/WEATHD/	WNDDRR, BSCC, SKYA, DBTR	None Placed
/ZNEWHR/	TZONER, INFCOD, QDUVAR, QTAVAN, CAMCON, DDUVAR, DTAVAN, CAMGUN, CFMINF	QDUVAR, QTAVAN, CAMCON, DDUVAR, DTAVAN, CAMGUN, CFMINF

Declarations

```
DIMENSION QYEN1 (15)
EQUIVALENCE (QYENI, QDUVAR)
```

```
COMMON AA(1000)
INTEGER IA(1000)
EQUIVALENCE (IA(1), AA(1))
```

Input

Name	Description
IA(MX)	Number of external surfaces
IA(MX+4)	External surface properties data pointer
AA(MX+7)	External surface multiplier
AA(MX+18)	Cosine of surface tilt angle
HORANG	Current hour angle of the sun (radians)
RTOT	Total solar radiation (Btu/hr-ft^2)
RDIR	Direct beam solar radiation on a surface (Btu/hr-ft^2)
AA(J+20)	Shadow ratio
IA(MP+6)	Surface roughness index
INFCOD	Infiltration Calculation Code
AA(MX+19)	Surface azimuth angle (radians)
WNDDRR	Wind direction (radians)
PIOVR2	Pi/2
PTWV	Theoretical wind velocity pressure
PSE	Stack effect pressure (height to neutral pressure zone)
IA(MP+7)	Number of response factors
AA(MP+8)	External surface U-factor
AA(MP+4)	External surface absorption factor
DBTR	Dry-bulb temperature (°R)
SKYA	Sky-view factor
TZONER	Zone calculation temperature (°R)
AA(MX+12)	Area of surface (ft^2)
AA(MX+16)	Surface tilt angle (radians)
PIOVR4	Pi/4
AA(MP+3)	Infiltration flow coefficient
AA(MX+20)	Surface perimeter (ft)
AA(MX+49)	Wall temperature (°F) (history)
AA(J+49)	Wall temperature (°F) (history)
AA(IR+12)	Last X value of response factor
AA(IR+13)	Last Y value of response factor

AA(MX+47)	Previous value for X-component heat flux
AA(MP+9)	Response factor common ratio
AA(MP+12)	Present X-component response factor
AA(MX+45)	Previous value for temperature history summation
AA(MX+48)	Previous value for Y-component heat flux
AA(MX+46)	Previous value for temperature history summation
AA(MP+13)	Present Y-component response factor
IA(MX+5)	Attachments data pointer
IA(MA)	Number of attachments
IA(MR+6)	Number of panes of glass
BSCC	Diffuse solar radiation (Btu/hr-ft^2) (brightness of sky)
BG	Ground-reflected solar radiation (Btu/hr-ft^2)
AA(MR+8)	Glass coefficient for transmission of direct solar radiation
AA(MR+9)	Glass coefficient for transmission of direct solar radiation
AA(MR+10)	Glass coefficient for transmission of direct solar radiation
AA(MR+11)	Glass coefficient for transmission of direct solar radiation
AA(MR+12)	Glass coefficient for absorption of direct solar radiation
AA(MR+13)	Glass coefficient for absorption of direct solar radiation
AA(MR+14)	Glass coefficient for absorption of direct solar radiation
AA(MR+15)	Glass coefficient for absorption of direct solar radiation
AA(MR+16)	Glass coefficient for transmission of diffuse solar radiation
AA(MR+17)	Glass coefficient for absorption of diffuse solar radiation
AA(MR+18)	Glass coefficient for absorption of direct solar radiation
AA(MR+19)	Glass coefficient for absorption of direct solar radiation
AA(MR+20)	Glass coefficient for absorption of direct solar radiation
AA(MR+21)	Glass coefficient for absorption of direct solar radiation
AA(MR+22)	Glass coefficient for absorption of direct solar radiation
AA(JJ+12)	Shadow data
AA(MA+11)	Sky form factor
AA(MA+12)	Ground form factor
AA(MA+9)	Net area of glass (ft^2)
AA(MR+5)	Shade coefficient
AA(MR+3)	Infiltration flow coefficient
AA(MA+10)	Perimeter (ft)
AA(MA+7)	Attachment (window) multiplier

MALEN	Length of attachment data table
IA(MX+11)	Length of external surface data table

Output

Name	Description
QDUVAR	Quick wall load (Btu/hr)
QTAVAN	Quick ceiling load (Btu/hr)
DDUVAR	Delayed wall load (Btu/hr)
DTAVAN	Delayed ceiling load (Btu/hr)
CFMINF	Infiltration flow (cfm)
CAMGUN	Solar heat flux through glass (Btu/hr)

Calculation Procedure

1. Initialize the external surface index and loop through each: The external surface index loop is the outermost DO loop of CALEXT.

```
NEXT = IA(MX)
DO 49900 IEXT = 1, NEXT
```

2. Set the external surface properties index and the surface multiplier.

```
MP = IA(MX+4)
XSMULT = AA(MX+7)
```

3. Initialize the solar data and determine if the sun is up. If so, call SUN3 and calculate the amount of solar radiation incident on a surface.

```
200 SOLI = RDIR = 0.
GAMMA = AA(MX+18)
IF(ABS(HORANG).GT.ABS(GUNDOG)) GO TO 300
CALL SUN3
J = IHR + MX
SOLI = RTOT - RDIR * AA(J+20)
```

4. Calculate the outside film coefficient.

```
FILMU = FILM(IA(MP+6))
```

5. Initialize the infiltration calculations and check the infiltration code. If crack method is set, continue calculations.

```
CFM = 0.
IF (INFCOD.NE.2) GO TO 600
```

6. Calculate wind direction angle for surface and then the total outside pressure.

```
SDELTA = AMIN1(ABS(AA(MX+19)-WNDDRR), PIOVR2)
PCO = PTWV * COS(SDELTA) + PSE
PCO = ABS(PCO)
```

7. Determine if surface is quick or delayed type.

```
600 IF(IA(MP+7).NE.0) GO TO 3000
```

Opaque Quick Surface Calculations

8. Determine surface heat transfer factor U; calculate the external surface temperature and the quick surface energy flux Q.

```
U = AA(MP+8)
TSOL = (SOLI*AA(MP+4)  + FILMU * DBTR-GAMMA * SKYA + U*TZONER)/(FILMU+U)
Q = U*(TSOL-TZONER)**AA(MX+12)
```

9. Sum energy flux under walls if vertical or ceiling if not.

```
IF (AA(MX+16).LT. PIOVR4) GO TO 1400
QDUVAR = QDUVAR + Q*XSMULT
GO TO 1500
1400 QTAVAN = QTAVAN + Q*XSMULT
```

10. If infiltration code is set for crack method, calculate infiltration rate (opaque quick surface).

```
1500 IF(INFCOD.EQ.2) CFM = AA(MP+3) * AA(MX+20) * (PCO**.5)
GO TO 4000
```

Opaque Delayed Surface Calculations

11. Calculate outside surface coefficients using film coefficients, temperature difference, solar radiation intensity, surface orientation, surface absorptivity, and surface response factor.

```
C2 = FILMU + AA(MP+12)
C3 = GAMMA * SKYA-AA(MP+12) * TZONER-FILMU * DBTR-AA(MP+4) * SOLI
```

12. Initialize the surface temperature history data. Loop through the response factor calculations and determine the current temperature history and heat balance.

```
SUMXDT = SUMYDT = 0.
TNEW = AA(MX+49)
IR = MP
J = MX
12 = IA(MP+7)-1
DO 3400 I = 2, I2
J = J+1
TOLD = AA(J+49)
AA(J+49) = TNEW
DNEW = TNEW-TZONER
TNEW = TOLD
IR = IR+3
SUMXDT = SUMXDT + AA(IR+12) * DNEW
SUMYDT = SUMYDT + AA(IR+13) * DNEW
3400 CONTINUE
```

13. Calculate the current exterior and interior energy flux components through the surface.

```
IR = IR+3
DT = TOLD-TZONER
XSXCMP = AA(MX+47) = AA(MP+9) * (AA(MX+47) - AA(MX+45)) + SUMXDT
         + AA(IR+12) * DT
XSQCMP = AA(MX+48) = AA(MP+9) * (AA(MX+48) - AA(MX+46)) + SUMYDT
         + AA(IR+13) * DT
```

14. Reset temperature history summation for next hour.

```
AA(MX+45) = SUMXDT
AA(MX+46) = SUMYDT
```

15. Calculate new external surface temperature and space heat gain or loss.

```
T = AA(MX+49) = - (C3+XSXCMP)/C2
Q = (XSQCMP + AA(MP+13) * (T-TZONER)) * AA(MX+12)
```

16. Calculate current delayed wall or delayed ceiling load according to surface tilt angle.

```
IF(AA(MX+16).LT.PIOVR4) GO TO 3600
DDUVAR = DDUVAR + Q * XSMULT
GO TO 3700
3600 DTAVAN = DTAVAN + Q * XSMULT
```

17. Calculate infiltration rate for opaque delayed surfaces.

```
3700 IF(INFCOD.EQ.2) CFM = AA(MP+3) * AA(MX+12) * (PCO**.8)
4000 CFMINF = CFMINF + CFM * XSMULT
```

Surface Attachments

18. Initialize and check for surface attachments.

```
MA = IA(MX+5)
IF (MA.EQ.0) GO TO 40000
NATT = IA(MA)
```

19. Loop through attachments, check the attachment type (1 = glass, 2 = door, 3 = hole),and go to relevant routine section.

```
DO 39900 IATT = 1,NATT
MR = IA(MA+4)
IF (IA(MA+5) -2) 31000, 32000, 33000
```

Glass Attachment Calculations

20. Determine glass coefficients and calculate overall U-factor.

```
31000 RO = 1./FILM(6)
RI = .71
RA = 0.
```

If the number of panes is greater than one, reset RA.

```
IF (IA(MR+6).GT.1) RA = 1.6
U = 1./(RO+RA+RI)
```

21. Check the amount of solar radiation available. Do not perform glass calculations if zero.

```
IF((RDIR.EQ.0).AND.(BSCC .EQ.0).AND.(BG.EQ.0)) GO TO 31300
```

22. Calculate the interior and exterior direct and diffuse absorption coefficients and the direct and diffuse transmission coefficients for the glass.

```
TDIR = AMAX1(0., AA(MR+8) + ETA * (AA(MR+9) + ETA * (AA(MR+10)
       + ETA * AA(MR+11))))
ADIRO = AA(MR+12) + ETA * AA(MR+13) + AA(MR+14)/(ETA+AA(MR+15))
```

```
TDIF = AA(MR+16)
ADIFO = AA(MR+17)
```

If more than one pane, ADIRO must be reset, and ADIRI and ADIFI are nonzero.

```
IF (IA(MR+6).NE.1) GO TO 31100
ADIRI = ADIFI = 0.
GO TO 31200
ADIRO = AMAX1(ADIRO, 0.)
ADIRI = AMAX1(0., AA(MR+18) + ETA ** AA(MR+19) + AA(MR+20)/
        (ETA + AA(MR+21)))
ADIFI = AA(MR+22)
```

Glass Heat Gain

23. Calculate the interior and exterior film coefficients.

```
FI = (RO+RA) * U
FO = RO * U
```

24. Calculate the direct and diffuse energy available to the glass, the amounts transmitted and absorbed, and the total heat gain of the glass without shading considered.

```
JJ = MA+IHR
QDIR = (1.-AA(JJ+12)) * RDIR
QDIF = BSCC * AA(MA+11) + BG * AA(MA+12)
QTRANS = QDIF * TDIF + QDIR * TDIR
QABS = QDIF * (FO * ADIFO + FI * ADIFI) + QDIR * (FO * ADIRO + FI * ADIRI)
QGUNES = (QTRANS + QABS) * AA(MA + 9)
```

25. Add shading effect if available.

```
IF (AA(MR+5).GT.0) QGUNES = QGUNES * AA(MR+5)
GO TO 31400
31300 QGUNES = 0.
```

26. Calculate the glass conductance gains and infiltration rate around window.

```
31400 QCON = U * (DBTR-TZONER) * AA(MA+9)
CFM = 0.
IF (INFCOD.EQ.2) CFM = AA(MR+3) * AA(MA+10) * (PCO**.66)
```

27. Sum the glass heat gains caused by infiltration, radiation, and conduction.

```
ATXSM = AA(MA+7) * XSMULT
CFMINF = CFMINF + CFM * ATXSM
CAMGUN = CAMGUN + QGUNES * ATXSM
CAMCON = CAMCON + QCON * ATXSM
GO TO 39000
```

28. Increment pointers and return to start of loops.

```
32000 CONTINUE
33000 CONTINUE
39000 MA = MA + MALEN
39900 CONTINUE
40000 MX = MX + IA(MX+11)
49900 CONTINUE
RETURN
```

ASHRAE Verification

Several different types of loads are calculated in the subroutine CALEXT. The first is the infiltration flow rate using the crack method. Table III.2 lists the variables used to calculate the infiltration rate and their respective counterparts in NECAP (Ref. 3) and ASHRAE (Ref. 1).

TABLE III.2
INFILTRATION CALCULATIONS

Cal-ERDA Name	NECAP Name	NECAP Location	ASHRAE Name	ASHRAE Location	Equivalent
SDELTA	SDELTA	LOADS	X	INFIL	YES
COS(SDELTA)	PTKO	LOADS	PTKO	INFIL	YES
PTWV	PTWV	LOADS	PTWV	INFIL	YES
PSE	PSE	LOADS	PSE	INFIL	YES/NO
PCO	PCO	LOADS	PCO	INFIL	YES/NO
CFM	CFMD,CFMW,CFMQ	LOADS	LEAK	INFIL	YES/NO
CFMINF	CFMINF	LOADS	0	INFIL	YES/NO

The Cal-ERDA calculation is identical to the NECAP method. Both procedures begin to vary from ASHRAE in the computation of the stack effect pressure, PSE. The ASHRAE method employs the equation

$$(PSE)_k = -0.52*PO*HT/TO \qquad \text{(p. 149. Ref. 1)}$$

where

PO = 0.4910*barometric pressure (in. Hg.),

TO = Dry-bulb temperature (°R), and

*HT = Height to neutral zone (ft).

Both Cal-ERDA and NECAP, however, use the equation

$$PSE = .25532 * PATM * AA(MZ+39)/(DBTR-TZONER),$$

where

AA(MZ+39) = Height to neutral zone (ft),

DBTR = Dry-bulb temperature (°R), and

TZONER = Zone temperature (°R).

* Note that the ASHRAE documentation (Ref. 1) does not state that this is a height-to-neutral zone measurement. It is erroneously given as simply the height of the floor.

The apparent difference then is that Cal-ERDA and NECAP use a temperature difference in the denominator, while ASHRAE employs only a single temperature.

The second variable that differs from the ASHRAE documentation (Ref. 1) is the total outside pressure, PCO. The ASHRAE equation for this calculation is

$$(PCO)_k = (PAWV)_k * (PSE)_k,$$

where

$$(PAWV)_k = (PTKO)_k * (PTKN)_k * (PTWV)_k,$$

and

PTKO = Wind velocity pressure correction factor for winds oblique to the surface,

PTKN = Wind velocity pressure correction factor for winds normal to the surface, and

PTWV = Theoretical wind velocity pressure.

Therefore, $(PCO)_k = (PTKO)_k * (PTKN)_k * (PTWV)_k * (PSE)_k$.

The Cal-ERDA counterpart to this equation is

PCO = PTWV * COS(SDELTA) * PSE,

where

COS(SDELTA) = PTKO.

The difference is that the factor PTKN is not present in the Cal-ERDA equation.

The third variable in question is the leak-rate calculation for each surface. The ASHRAE equation used to approximate this is

LEAK = 4000 * A * K * (DP**N)
= C *(DP**N),

where

A = Opening area (ft^2),

K = Flow coefficient (demensionless),

DP = Pressure difference,

N = Pressure exponent, and

C = Equivalent flow coefficient (per foot of linear crack length).

The equivalent Cal-ERDA equation is

CFM = AA(MP+3) ** AA(MX+20) * (PCO**N),

where

AA(MP+3) = C,

AA(MX+20) = Linear crack length (perimeter), and

N = 0.5 for quick surfaces,

0.66 for glass surfaces, and

0.8 for delayed surfaces.

It is evident that the Cal-ERDA variable CFM is identical in method to the ASHRAE variable LEAK, the values of C and N corresponding to the values given in Table A-17 of Ref. 1. However, the Cal-ERDA term PCO is not purely ASHRAE-derived, as stated earlier, and thus the variable CFM is given a YES/NO equivalence.

The final variable CFMINF is simply the sum of the values of CFM and, therefore, is also given a YES/NO equivalence.

The next load calculated by CALEXT is the load caused by heat gain through opaque quick surfaces (e.g. steel doors, etc.). These are based on the simple steady-state equation,

$$Q = UA(T_o - T_i).$$

The exterior temperature is computed as an apparent outside temperature and considers the solar radiation, sky-view factor, surface orientation, film coefficient, and dry-bulb temperature. This is in agreement with ASHRAE, although a specific algorithm for quick surfaces is not given in the documentation (Ref. 1). It is identical to the NECAP subroutine HQ (Ref. 3).

Transient heat transfer loads are computed next for opaque delayed surfaces. These involve the use of response factors. The response factors are calculated in the BDL Program for each wall (see Sec. A.2) and employ the ASHRAE methodology. There is no library of values for predefined walls, each being constructed layer by layer. The application of these response factors follows the ASHRAE method as outlined in HEATW (Ref. 1) using a recursive summation technique developed by M. Lokmanhekim. Table III.3 gives the Cal-ERDA output variables and their equivalents in NECAP (Ref. 3) and ASHRAE.

TABLE III.3

	Cal-ERDA Name	NECAP Name	NECAP Location	ASHRAE Name	ASHRAE Location	Equivalent
1.	T	TO(1)	HD	TOS_t	HEATW	Yes
2.	Q	Q	HD	$HEAT_t$	HEATW	Yes

The final energy calculation that takes place in subroutine CALEXT involves the energy gain through glass surfaces. The Cal-ERDA procedure follows that of ASHRAE subroutine SHG (Ref. 1), with the exception that the coefficients for glass absorption and transmission are taken from tables in the Cal-ERDA BDL Program, while ASHRAE calculates them separately in subroutine TAR. Also, the Cal-ERDA method separates the heat gain into absorption and transmission components, while ASHRAE separates the energy into direct and diffuse components. The only important variable to mention is the total glass heat gain called QGUNES in Cal-ERDA and SHG in ASHRAE; they are equivalent.

4. Subroutine CALOTH.

Description

This subroutine calculates the internal zone loads for the building. Weighted loads because of lights, internal walls, and solar are calculated, as well as sensible and latent loads for people, equipment, and infiltration. Underground surfaces are also accounted for.

Subprograms Calling This Subroutine

SUBROUTINE DAYCLC

Subprograms Called 3y This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/DATE/	IFSTHR, ISCHR	None placed
/PNTRS/	MZ, MS	MS
/WEATHD/	WNDSPD, TGNDR, HUMRAT, DENSTY, DBT, DBTR	None placed
/ZNEWHR/	TZONER, QDUVAR, QTAVAN, CAMCON, CAMGUN, DDUVAR, QINTHT, QUGF, QUGW, QEQPS, QEQPS2, QEQPL, QEQPL2, DTAVAN, QPPLL, QINFL, SDIGER, SCAM, SLAMBA, QINFS, QLAMBA, CFMINF, QODA, QDIGER, QZS, QZL, QPPLS	QINTHT, QUGF, QUGW, QEQPS, QEQPS2, QEQPL, QEQPL2, QPPLL, QINFL, SDIGER, SCAM, QINFS, QLAMBA, CFMINF, QODA, QDIGER, QZS, QZL, QZTOT, QPPLS

Declarations

```
DIMENSION QYENI(15)
EQUIVALENCE (QYENI, QDUVAR)
COMMON AA(1000)
INTEGER IA(1000)
EQUIVALENCE (IA(1), AA(1))
```

Input

Name	Description
AA(MZ+41)	Zone internal surface heat gain
AA(MZ+22)	Zone underground wall heat transfer function, U * A

TGNDR	Ground temperature (°R)
TZONER	Zone calculation temperature (°R)
AA(MZ+26)	Zone underground floor heat transfer function, U * A
IA(MZ+19)	People schedule
IA(KS+3)	Current schedule vector pointer
ISCHR	Daylight savings time corrected hour
AA(MS-1)	Schedule for current hour
AA(MZ+42)	People sensible heat without schedule
AA(MZ+21)	People latent heat without schedule
IA(MZ+27)	Equipment schedule
AA(MZ+28)	Equipment zone load total, sensible (Btu/hr)
AA(MZ+29)	Equipment load, per cent sensible (Btu/hr)
AA(MZ+30)	Equipment load, per cent latent (Btu/hr)
AA(MZ+28)	Equipment power (kW)
IA(MZ+31)	Special equipment schedule
AA(MZ+33)	Special equipment load, per cent sensible (Btu/hr)
AA(MZ+34)	Special equipment load, per cent latent (Btu/hr)
IA(MZ+23)	Lighting schedule
AA(MZ+24)	Lighting power (kW)
AA(MZ+25)	Zone lighting load (Btu/hr)
IFSTHR	First-hour flag
CAMGUN	Solar heat gain to zone through glass
QDIGER	Current hour heat flux through floors and walls to be weighted
QLAMBA	Current light load to be weighted
AA(MZ+43)	Weighting factor for heat gain through walls and floors
AA(MZ+44)	Weighting factor for heat gain through walls and floors
AA(MZ+45)	Weighting factor for heat gain through walls and floors
AA(MZ+13)	Previous hour heat flux through walls and floors
AA(MZ+10)	Previous hour heat flux through walls and floors
AA(MZ+46)	Weighting factor for solar heat gain through glass
AA(MZ+47)	Weighting factor for solar heat gain through glass
AA(MZ+48)	Weighting factor for solar heat gain through glass
AA(MZ+5)	Previous hour heat flux through glass
AA(MZ+9)	Previous hour heat flux through glass
AA(MZ+49)	Weighting factor for heat gain from lights

AA(MZ+18)	Zone furniture weight
AA(MZ+50)	Weighting factor for heat gain from lights
AA(MZ+51)	Weighting factor for heat gain from lights
AA(MZ+11)	Previous hour heat flux for lighting
AA(M+54)	
AA(M+65)	
AA(MZ+52)	Weighting factor for immediate heat gain to plenum from lighting
AA(MZ+53)	Weighting factor for immediate heat gain to plenum from lighting
AA(MZ+54)	Weighting factor for immediate heat gain to plenum from lighting
AA(MZ+12)	Previous hour heat gain to plenum
AA(MZ+11)	Previous hour light load (QLAMBA)
IA(MZ+35)	Infiltration calculation code
AA(MZ+15)	Zone volume (ft^3)
AA(MZ+36)	Number of air changes
WNDSPP	Wind speed
AA(MZ+38)	Exhaust air
IA(MZ+37)	Infiltration schedule
HUMRAT	Humidity ratio
DBT	Dry-bulb temperature (°F)
DENSTY	Air density

Output

Name	Description
CFMINF	Infiltration flow rate (cfm)
QINFS	Infiltration sensible load (Btu/hr)
QINFL	Infiltration latent load (Btu/hr)
QZL	Zone latent load (Btu/hr)
QZS	Zone sensible load (Btu/hr)
QZTOT	Total zone load (Btu/hr)

Calculation Procedure

1. Calculate the internal and underground heat gains.

```
QINTHT = AA(MZ+41)
QUGW = AA(MZ+22) * (TGNDR-TZONER)
QUGF = AA(MZ+26) * (TGNDR-TZONER)
```

2. Calculate people loads, both sensible and latent. If no schedule is available, set to zero and continue.

```
QPPLS = 0.
QPPLL = 0.
KS = IA(MZ+19)
IF (KS.EQ.0) GO TO 1200
MS = IA(KS+3) + ISCHR
QPPLS = AA(MS-1) * AA(MZ+42)
QPPLL = AA(MS-1) * AA(MZ+21)
1100 CONTINUE
```

3. Calculate equipment loads, including sensible, latent, and electrical. If no schedule is available, set to zero and continue.

```
QEQPS = 0.
QEQPL = 0.
QELECT = 0.
KS = IA(MZ+27)
IF (KS.EQ.0) GO TO 1200
MS = IA(KS+3) + ISCHR
QEQPS = AA(MS-1) * (AA(MZ+28) + AA(MZ+29)
QEQPL = AA(MS-1) *  AA(MZ+30)
QELECT = AA(MS-1) * AA(MZ+28)
1200 CONTINUE
```

4. Calculate the sensible and latent components of process heat. If no schedule is available, set to zero and continue.

```
QEQPS2 = 0.
QEQPL2 = 0.
KS = IA(MZ+31)
IF (KS.EQ.0) GO TO 1300
MS = IA(KS+3) + ISCHR
QEQPS2 = AA(MS-1) * AA(MZ+33)
QEQPL2 = AA(MS-1) * AA(MZ+34)
```

5. Calculate the lighting load. If no schedule is available, continue.

```
KS = IA(MZ+23)
IF (KS.EQ.0) GO TO 1400
MS = IA(KS+3) + ISCHR
QELECT = QELECT + AA(MS-1) * AA(MZ+24)
QLAMBA = AA(MS-1)
1400 CONTINUE
```

Weighting Factor Calculations

6. Initialize the output variables.

```
QDIGER = QEQPS + QDUVAR + QTAVAN + DDUVAR + DTAVAN + QUGF + QUGW + QINTHT
         + QPPLS + CAMCON + QEQPS2
```

7. If it is the first hour of the run period, initialize each type of load for a previous value. For example, assume the previous hour glass-heat gain to be the same as the current hour.

```
IF (IFSTHR.EQ.0) GO TO 2300
AA(MZ+9) = AA(MZ+5) = CAMGUN
AA(MZ+10) = AA(MZ+13) = QDIGER
AA(MZ+11) = QLAMBA
AA(MZ+18) = .85 * QLAMBA
AA(MZ+12) = .15 * QLAMBA
DO 2200 I = 1,11
M = MZ+I
AA(M+54) = AA(M+65) = QYENI(I)
2200 CONTINUE
```

8. Perform the weighting factor calculations.

```
2300 BX1M = AA(MZ+43)
AXO = AA(MZ+44)
AX1 = AA(MZ+45)
SDIGER = AA(MZ+13) = BXIM * AA(MZ+13) + AXO * QDIGER + AX1 * AA(MZ+10)
SCAM = AA(MZ+5) = AA(MZ+46) * AA(MZ+5) + AA(MZ+47) * CAMGUN + AA(MZ+48)
                     * AA(MZ+9)
SLAMBA = AA (MZ+18) = AA(MZ+49) * AA(MZ+18) + AA(MZ+50) * QLAMBA + AA(MZ+51)
                         * AA(MZ+11)
```

9. Increment history data for each.

```
DO 2400 I=1,11
M = MZ+I
AA(M+64) = BX1M * AA(M+64) + AXO * QYENI(I) + AX1 * AA(M+54)
AA(M+54) = QYENI(I)
QYENI(I) = AA(M+65)
2400 CONTINUE
```

10. Store current hour data to be used as previous hour data for the next hour.

```
QODA = SDIGER + SCAM + SLAMBA
AA(MZ+12) = AA(MZ+52) * AA(MZ+12) + AA(MZ+53) * QLAMBA + AA(MZ+54) * AA(MZ+11)
AA(MZ+11) = QLAMBA
AA(MZ+10) = QDIGER
AA(MZ+9) = CAMGUN
```

11. Calculate sensible and latent loads caused by infiltration.

```
IF (IA(MZ+35)-1) 3100, 3200, 3300
3100 QINFS = QINFL = 0.
GO TO 3500
3200 CFMINF = AA(MZ+15) * AA(MZ+36) * WNDSPD/691.8
GO TO 3400
3300 CFMINF = AMAX1(0.,CFMINF-.335*(-AA(MZ+38)))
3400 CFMINF = CFMINF + AA(MZ+38)
IF(IA(MZ+37).EQ.0) GO TO 3420
KS = IA(MZ+37)
MS = IA(KS+3) + ISCHR
CFMINF = CFMINF * AA(MS-1)
3420 HRODA = HUMRAT
IF(DBT.GE.50) HRODA = (53.2+.245*(DBT-50.))/7000.
QINFS = 14.4 * DENSTY * CFMINF *(DBTR-TZONER)
QINFL = 63000. * DENSTY * CFMINF *(HUMRAT-HRODA)
3500 CONTINUE
QINFL = AMAX1(QINFL,0.)
```

12. Sum the zone loads for output.

```
    QZL = QPPLL + QEQPL + QEQPL2
    QZS = QODA + QINFS
    QZTOT = QZL + QZS
    RETURN
```

5. Subroutine CCM.

Description

This subroutine calculates the cloud-cover coefficients for modifying solar radiation intensity for a clear atmosphere as a function of cloud type, cloud amount, and solar altitude angle.

Subprograms Calling This Subroutine

SUBROUTINE WDTSUN

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/SUND/	RAYCOS(3)	None placed
/WEATHD/	CLDAMT, ICLDTY	CLDCOV

Declarations

```
    REAL ICLD
    EQUIVALENCE (ICLD, CLDAMT), (ICLTP, ICLDTY), (CLDCOV, CC), (AL, RAYCOS(3))
```

Input

Name	Description
AL	Altitude angle of the sun (radians)
ICLD	Total amount of cloud cover (0-10) (dimensionless)
ICLTP	Cloud type index for lowest cloud layer (0 = cirrus cirrostratus; 1 = stratus; and 2 = other)

Output

Name	Description
CLDCOV	Cloud cover modifier (CC) defined as the ratio of the radiation on a horizontal surface for the hour with cloud cover to the radiation on a horizontal surface for the hour for a clear sky

Calculation Procedure

The value of CLDCOV is determined by a set of equations that are curve-fits for data supplied by Ref. 10. These values are given in Table III.4.

TABLE III.4

CLOUD COVER MODIFIER, CLDCOV

ICLTP	Stratus		Cirrus, Cirrostratus	
ICLD	AL < 45°	AL > 45°	AL < 45°	AL > 45°
1	0.60	0.88	0.84	1.0
2	0.60	0.88	0.83	1.0
3	0.58	0.88	0.83	1.0
4	0.58	0.87	0.82	1.0
5	0.57	0.85	0.80	0.99
6	0.53	0.83	0.77	0.98
7	0.49	0.79	0.74	0.95
8	0.43	0.73	0.67	0.90
9	0.35	0.61	0.60	0.84
10	0.27	0.46	0.49	0.74

The equations used are:

$SQ = ICLD * ICLD$

ICLTP = 1 (stratus clouds)

$AL \leq 45°$

$CC \overline{=} 0.598 + 0.00026 * ICLD + 0.00021 * SQ - 0.00035 * ICLD * SQ$

$AL > 45°$

$CC = 0.908 - 0.03214 * ICLD + 0.0102 * SQ - 0.00114 * ICLD * SQ$

ICLTP = 0 (cirrus, cirrostratus)

$AL \leq 45°$

$CC \overline{=} 0.849 - 0.01277 * ICLD + 0.0036 * SQ - 0.0059 * ICLD * SQ$

$AL > 45°$

$CC = 1.01 - 0.01394 * ICLD + 0.00553 * SQ - 0.00068 * ICLD * SQ$

For cloud types other than cirrus, cirrostratus and stratus, an average value of CLDCOV for ICLTP = 0 and 1 must be used. These equations are given below.

ICLTP = 2 (other cloud types)

$AL \leq 45°$

$CC \overline{=} 0.724 - 0.00625 * ICLD + 0.00191 * SQ - 0.0047 * ICLD * SQ$

$AL > 45°$

$CC = 0.959 - 0.02304 * ICLD + 0.00787 * SQ - 0.00091 * ICLD * SQ$

ASHRAE Verification

The Cal-ERDA subroutine CCM is identical to the subroutine CCM in the NECAP program (Ref. 3). It does not, however, bear any resemblance to the ASHRAE cloud-cover subroutine CCF (Ref. 1). The ASHRAE subroutine divides the cloud cover into four layers with each layer identified as to cloud type and amount. These values are summed and one-half of their value is subtracted from the total cloud amount to produce a value for cloud cover, CC. This number is then employed with a set of coefficients for each of the four seasons to produce a cloud cover factor, CCF.

In contrast, the Cal-ERDA and NECAP method assumes only a single layer; cloud type for the lowest layer and the total amount of cloud cover (all layers) are used. Curve-fit equations to empirical data are then used.

6. Subroutine DAYCLC.

Description

Subroutine DAYCLC is the driver routine for the calculation of daily and hourly loads.

Subprograms Calling This Routine

Program LOADS

Subprograms Called by This Routine

Subroutine BLOCKO
Subroutine CALEXT
Subroutine CALOTH
Subroutine DST
Subroutine FILLN
Subroutine HOLDAY
Function IDOYR
Subroutine PEAKCK
Subroutine SHADOW
Subroutine SUN1
Subroutine WDTSUN

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/BLDPK/	None obtained	BLDPK array
/DATE/	IDOY, ISCDAY, IHR, IDSTF, IPRDFL	ISCHR
/FILES/	ISYSHR	None placed
/GPNTRS/	NSCT, MZONTB, NZONT	None placed
/WEATHD/	PATM, DBTR	None placed
/INFPAR/	None obtained	PSE
/PNTRS/	MZLEN	MZ
/SHADWF/	IGOLGE	None placed
/SPCD/	SPCD array	SPCD array

/SUND/	None obtained	None placed
/TITLE/	ITITLE(1,3)	None placed
/ZNEWHR/	None obtained	QDUVAR, QTAVAN, CAMCON, DDUVAR, QINTHT, QUGF, QPPLS, QUGW, QEQPS, DTAVAN, QPPLL, QEQPL, QEQPL2, QINFL, QINFS, QLAMBA, CAMGUN, QELECT, CFMINF, QZS, QZL, QZTOT

Declarations

```
    DIMENSION QYENI(15)
    EQUIVALENCE (QYENI, QDUVAR)
    COMMON AA(1000)
    INTEGER IA(1000)
    EQUIVALENCE (IA(1), AA(1))
    DIMENSION BLDDTH(17), BLDDTC(17)
    DIMENSION XX(8)
    DATA LDBGCT/-1/
    DATA SPCD/2550*0./
    DATA BLDPK/51*0.0/
```

Input

Name	Description
NSCT	Number of schedules
IDOY	Day of the year
IA(KS+5)	Schedule day of year for ending current block
IA(KS+6)	Current schedule block pointer
ISCDAY	Day of week for schedule selection
IHR	The current hour
IDSTF	Daylight savings time flag
IPRDFL	Flag that allows DAYCLC to be run three times at the beginning of the run period to allow weighting factors to stabilize
IGOLGE	Shading calculation flag, 1 = calculate shade ratios
MZONTB	Zone table beginning pointer
MZLEN	Space data entry length (length of MZ)
PATM	Atmospheric pressure (in. Hg)
DBTR	Dry-bulb temperature (°R)
AA(MZ+39)	Zone height to neutral zone
SPCD array	Previously established peak hourly zone loads
BLDPK array	Previously established peak building loads
ISYSHR	LOADS output file for hourly data for SYSTEMS

Output

Name	Description
SPCD array	New peak loads if any hourly loads this day surpassed previously established peaks
BLDPK array	Building peak loads if previous peak loads are exceeded

The following hourly data are sent to the SYSTEMS input file:

QZS	Sensible zone heat load
QZL	Latent zone heat load
QELECT	Electric power demand
QPLEN	Length of saved data array
QEQPS	Sensible equipment heat
QEQPL	Latent equipment heat
CFMINF	Infiltration air flow (cfm)

Calculation Procedure

1. CALL HOLDAY to identify holidays and set schedule index number.
   CALL DST to calculate daylight savings time flag.
   CALL SUN1 to calculate daily data on solar radiation.

2. Prepare schedule vector pointers for the current day.

   If number of schedules, NSCT, = 0, go to Step 3.

   a. Enter a DO loop, ISC = 1 to number of schedules.
   b If (IDOY.LE.IA(KS+5)), that is day-of-year < current day-of-year for ending current block, go to Step 2(e).
   c. Otherwise, move the schedule pointer to the next block:

      IS = IA(KS+6) = IA(KS+6) + 10

   d. Julian date of the current day of year for current block is reset by calling function IDOYR with this block ending month, day, and leap year flag.

   e. Current day's schedule vector pointer is calculated.

      IS = IA(KS+6) + ISCDAY
      IA(KS+3) = IA(IS+2)
      KS = IA(KS+4)

   f. Continue loop at 2(a)

3. Enter the hourly calculation loop, performing Steps 4 through 27 as IHR runs from 1 to 24.

4. CALL FILLN (0., BLDDTH(1), 17)
   CALL FILLN (0., BLDDTC(1), 17)
   FILLN will fill the BLDDTH and BLDDTC arrays with zeros for initialization.

5. Calculate corrected hour = current hour minus 1 if daylight savings time. If ISCHR = 0, it is set to 24.

```
ISCHR = IHR-IDSTF
IF(ISCHR.EQ.0) ISCHR = 24
```

6. Call WDTSUN.

   Subroutine WDTSUN will perform this hour's calculations for solar position and intensity.

7. Check IPRDFL. If it is not equal to zero, DAYCLC is being run one of three consecutive times at the beginning of the run period to allow weighting factors to stabilize. In this case, the program skips to Step 9.

8. If ITITLE(1,3) is not "DEBUG", skip to Step 9.
   Otherwise, a debugging print is made of hour, day, month, daylight savings time corrected hour, sunrise hour, dry- and wet-bulb temperature, and cloud cover. On the basis of LDBGCT, a print is made every sixth hour, i.e., four times a day, of certain headings for the above data.

9. If IGOLGE ≠ 0, shade ratios are to be calculated and Subroutine SHADOW is called.

10. The MZ pointer is set to MZONTB-MZLEN.

11. A loop is entered, IZONE = 1, NZONT, number of zones, performing Steps 12 through 26.

12. Increment MZ=MZ+MZLEN.
    Pointers MX (zone external surface pointer) and MI (zone internal surface pointer) are set.

    ```
    MZ = IA(MZ+40)
    MI = IA(MZ+41)
    ```

    Temperature of zone, Rankine, is assigned: TZONER = AA(MZ+8) and infiltration calculations code.

    ```
    INFCOD = IA(MZ+35)
    ```

13. Zone height to neutral pressure zone, AA(MZ+39), is adjusted for atmospheric pressure (PATM), dry-bulb temperature (°R), and zone temperature (°R).

    ```
    PSE = .25532 * PATM * (1./DBTR-1./TZONER)* AA(MZ+39)
    ```

14. Initialize output variables.

    ```
    QEQPS = QDUVAR = DDUVAR = QUGF = QUGW = QINTHT = QPPLS = 0.
    CAMCON = QTAVAN = DTAVAN = QPPLL = CAMGUN = QEQPL = ELCPWR = 0.
    CFMINF = QINFS = QINFL = QLAMBA = 0.
    QELECT = 0.
    ```

15. If MX, external wall data pointer ≠ 0 CALL CALEXT for thermal load calculations for external walls. Then CALL CALOTH internal zone loads.

16. Statistics are calculated.

    ```
    QYENI(1)  = QYENI(1) + QYENI(4) (quick plus delayed wall loads)
    QYENI(4)  = SCAM (weighted solar gain through glass)
    QYENI(2)  = QYENI(2) + QYENI(11) (quick plus delayed ceiling loads)
    QYENI(11) = QINFS (infiltration, sensible heat)
    QYENI(6)  = QYENI(6) + QYENI(8) (underground floors and walls loads)
    QYENI(8)  = SLAMBA (weighted lighting load)
    ```

17. If QZTOT, total zone load, > 0, or if QZS, sensible heat load, > SPCD (1,IZONE), space peak load for this hour, go to Step 19.

    a. Otherwise, reset peak load for this hour:
    SPCD(1,IZONE) = QZS
    Set SPCD(2,IZONE) to a packed binary notation of current month, day, and hour of peak load;
    SPCD(2,IZONE) = SHIFT(IMON,12).OR.SHIFT(IDAY,6).OR.IHR
    Set SPCD(3,IZONE) = DBT, dry-bulb temperature
    SPCD(4,IZONE) = WBT, wet-bulb temperature

    b. Enter a DO loop, I=1,15
    SPCD(I+14,IZONE) = QYENI(I)

18. If total zone load, QZTOT, < SPCD(5,IZONE), previous peak load, go to Step 20.

    Otherwise,

    a. SPCD(5,IZONE) = QZTOT
    SPCD(6,IZONE) = SHIFT(IMON,12).OR.SHIFT(IDAY,6).OR.IHR, packed date data
    SPCD(7,IZONE) = QZS, sensible heat load
    SPCD(8,IZONE) = DBT, dry-bulb temperature
    SPCD(9,IZONE) = WBT, wet-bulb temperature

    b. A DO loop is entered, I=1,15
    SPCD(I+29,IZONE) = QYENI(I)

19. If QZS, zone sensible load, < SPCD(10,IZONE), previous peak load, go to Step 21.

    Otherwise,
    SPCD(10,IZONE) = QZS
    SPCD(11,IZONE) = SHIFT(IMON,12).OR.SHIFT(IDAY,6).OR.IHR
    SPCD(12,IZONE) = QZL, zone latent heat load
    SPCD(13,IZONE) = DBT, dry-bulb temperature
    SPCD(14,IZONE) = WBT, wet-bulb temperature

20. ZMULT = AA(MZ+7), zone multiplier, i.e., number of identical zones.

21. If QZTOT > 0, go to Step 23.

    Otherwise,

    a. Enter a DO loop, I=1,15
    BLDDTH(I+2) = BLDDTH(I+2) + QYENI(I) * ZMULT

    where BLDDTH = temperature difference for building heating load.

    b. BLDDTH(1) = BLDDTH(1) + QZTOT * ZMULT

    c. BLDDTH(2) = BLDDTH(2) + QXS * ZMULT

    d. Go to Step 24.

22. As there is no heating load, cooling load calculations are done instead.

    a. Enter a DO loop, I=1,15
    BLDDTC(I=2) = BLDDTC(I+2) + QYENI(I) * ZMULT

    where BLDDTC = temperature difference for cooling load.

    b. BLDDTC(1) = BLDDTC(1) + QZTOT * ZMULT

    c. BLDDTC(2) = BLDDTC(2) + QZS * ZMULT

23. QPLEN = AA(MZ+12), length of saved data array.

24. Set up the XX array.

```
XX(1) = QZS
XX(2) = QZL
XX(3) = QELECT
XX(4) = QPLEN
XX(5) = QEQPS
XX(6) = QEQPL
XX(7) = CFMINF
XX(8) = 0
```

25. Call BLOCKO(ISYSHR, XX, 8) to send calculated hourly data to the SYSTEMS file.

26. Call subroutine PEAKCK for heating and cooling building load calculations.

```
CALL PEAKCK (BLDDTH, BLDPK)
CALL PEAKCK (BLDDTC, BLDPK)
IFSTHR = 0., first-hour flag  is off
```

27. RETURN

ASHRAE Verification

ASHRAE documentation (Ref. 1) has no equivalent algorithm to subroutine DAYCLC.

7. Subroutine DESFOU.

Description

This subroutine produces an output design file for SYSTEMS.

Subprograms Calling This Subroutine

PROGRAM LOADS

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/FILES/	None obtained	ISYSDS
/GPNTRS/	MZONTB	None placed
/IWCPTR/	Not currently used here	
/PNTRS/	MZ, MZLEN	MZ
/RUNPRD/	NRUNPR, IRUNPR(6,12) RUNSCA(12)	None placed
/SIZE/	NSP	None placed
/SPCD/	None obtained	SPCD(51,50)
/TITLE/	ITITLE(8,5), NTITLE	None placed
/WEATHD/	IWDID(2)	None placed

Declarations

```
INTEGER ISPCD (51,25)
EQUIVALENCE (ISPCD, SPCD)
COMMON AA(1000)
INTEGER AA(1000)
EQUIVALENCE (IA(1), AA(1))
```

Input

Name	Description
MZONTB	Zone table beginning pointer
NSP	Number of zones
SPCD(1,IZ)	Maximum heating load
SPCD(10,IZ)	Maximum cooling load
IA(MZ+1)	Zone identification (name)
IA(MZ+2)	Second word of zone identification
AA(MZ+41)	Zone internal surface heat gain
AA(MZ+26)	UA value for underground floors
AA(MZ+22)	UA value for underground walls
AA(MZ+6)	Zone conductance
AA(MZ+17)	Zone floor weight
AA(MZ+16)	Zone plenum volume
AA(MZ+14)	Zone floor area
AA(MZ+15)	Zone volume
AA(MZ+20)	Maximum number of people
AA(MZ+8)	Zone temperature, heating
AA(MZ+7)	Zone multiplier
MZLEN	Space data entry length for each attachment

Output

Name	Description
QHMAX	Maximum heating load
QCMAX	Maximum cooling load
ISPCD(1,IZ)	Zone identification (name)
ISPCD(2,IZ)	Second word of zone identification
SPCD(3,IZ)	Zone internal surface heat gain
SPCD(4,IZ)	Sum of UA values for underground floors and walls
SPCD(5,IZ)	Zone conductance
SPCD(6,IZ)	Zone floor weight
SPCD(7,IZ)	Zone plenum volume

SPCD(8;IZ)	Zone floor area
SPCD(9,IZ)	Zone volume
SPCD(10,IZ)	Maximum number of people
SPCD(11,IZ)	Zone temperature, heating
SPCD(12,IZ)	Maximum cooling load
SPCD(13,IZ)	Maximum heating load
SPCD(14,IZ)	Zone multiplier
SPCD(15,IZ)	0. (not currently used)
ISYSDS	The output design file for SYSTEMS

Calculation Procedure

1. Locate zone data beginning point.

```
MZ = MZONTB
```

2. Loop through zones and locate required data.

```
DO 20000 IZ  = 1,NSP
QHMAX        = SPCD(1,IZ)
QCMAX        = SPCD(10,IZ)
ISPCD(1,IZ)  = IA(MZ+1)
ISPCD(2,IZ)  = IA(MZ+2)
SPCD(3,IZ)   = AA(MZ+41)
SPCD(4,IZ)   = AA(MZ+26) + AA(MZ+22)
SPCD(5,IZ)   = AA(MZ+6)
SPCD(6,IZ)   = AA(MZ+17)
SPCD(7,IZ)   = AA(MZ+16)
SPCD(8,IZ)   = AA(MZ+14)
SPCD(9,IZ)   = AA(MZ+15)
SPCD(10,IZ)  = AA(MZ+20)
SPCD(11,IZ)  = AA(MZ+8)-460.
SPCD(12,IZ)  = QCMAX
SPCD(13,IZ)  = QHMAX
SPCD914,IZ)  = AA(MZ+7)
SPCD(15,IZ)  = 0.
```

3. Increment zone pointer.

```
MZ = MA + MZLEN
```

4. Loop return

```
20000 CONTINUE
```

5. Create the output design file for SYSTEMS.

```
WRITE (ISYSDS)    ((ITITLE(J,I), J = 1,8),
                 I = 1,5), NTITLE, NRUNPR,
                 ((IRUNPR (J,I), J = 1,6),
                 RUNSCA (I),I = 1, NRUNPR),
                 IWDID (I), I = 1,2), NSP,
                 ((SPCD (J,I), J = 1,15),
                 I = 1, NSP)
```

ASHRAE Verification

There is no equivalent ASHRAE algorithm for the Cal-ERDA LOADS subroutine DESFOU.

8. Subroutine DST.

Description

The DST subroutine is a simple algorithm used to calculate the daylight savings time flag IDSTF.

Subprograms Calling This Subroutine

SUBROUTINE DAYCLC

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/DATE/	IMON, IDAY, IDOW	IDSTF

Declarations

```
DIMENSION M4TBL (7), M1OTB (7)
DATA      M4TBL/24, 30, 29, 28, 27, 26, 25/, M1OTBL/25, 31, 30, 29, 28,
          27, 26/
```

Input

Name	Description
IMON	The month of year, 1-12 (dimensionless)
IDAY	The day of the month, 1-31 (dimensionless)
IDOW	The day of the week, 1-7 (dimensionless)

Output

Name	Description
IDSTF	The daylight savings time flag; an index number that is either zero (0) for standard time or one (1) for daylight savings time (dimensionless)

Calculation Procedure

1. Check for the month of year, IMON. If IMON is less than 4 (April), set ISDTF equal to zero.
2. If IMON is greater than 4 (April) and less than 10 (October), set IDSTF equal to one.
3. If IMON is greater than 10 (October), set IDSTF equal to zero.
4. For IMON equal to 4 (April):
   a. If IDAY is less than 24, set IDSTF equal to zero.
   b. If IDAY is equal to 24 and it is a Sunday, set IDSTF equal to one.
   c. If IDAY is equal to 24, but it is not a Sunday, set IDSTF equal to one on the first following Sunday.

5. For IMON equal to 10 (October):

a. If IDAY is less than 24, set IDSTF equal to one.
b. If IDAY is equal to 25 and it is a Sunday, set IDSTF equal to zero.
c. If IDAY is equal to 24 and it is not a Sunday, set IDSTF equal to zero on the first Sunday to follow.

ASHRAE Verification

The sample subroutine DST in Cal-ERDA is identical to the DST subroutine of NECAP (Ref. 3) and the ASHRAE subroutine DST (Ref. 1) in that daylight savings time is initiated on the 24th of April if it is a Sunday or the 1st Sunday to follow and is discontinued on October 25th if it is a Sunday or the 1st Sunday to follow. The only difference is that ASHRAE DST calls the subroutine WKDAY to determine the appropriate day of the week, and Cal-ERDA uses the variable IDOW carried in the common block /DATE/.

9. Subroutine EXTPRP.

Description

This subroutine prepares data for the LOADS Program pertinent to external surfaces.

Subprograms Calling This Subroutine

SUBROUTINE READSF

Subprograms Called by This Subroutine

SUBROUTINE SETBAC
SUBROUTINE FILLN
SUBROUTINE GEOPR1

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/CONST/	DTOR	None placed
/LOCALD/	BAZIM	None placed
/PNTRS/	MZ, MX, MP, MA, MR, MALEN	MA, MR, MG, MH
/PTRBGN/	IBEW, IBAT, IBSH, IBPR, IBGE	None placed
/SIZE/	Not currently used here	
/SURFD/	AREA, PERIM, AZIM, TILT	None placed
/WEATHD/	BDTR	None placed
/ZCOND/	None obtained	ZCOND
/ZWALOC/	None obtained	XWO, YWO, ZWO, AWO, TWO, SAW, CAW, STW, CTW, WX, YW

Declarations

```
COMMON AA(1000)
INTEGER IA(1000)
EQUIVALENCE (IA(1), AA(1))
```

Input

Name	Description
IA(MZ+40)	Zone external surface pointer
IBEW	External wall data block beginning pointer
IA(MX)	Number of external surfaces in zone
IA(MX+3)	Geometry table pointer
IBGE	Geometry data block beginning pointer
IA(MX+4)	Property data table pointer
IBPR	Properties data block beginning pointer
IA(MX+5)	Attachments data table pointer
IBAT	Attachments data block beginning pointer
IA(MX+6)	Shading surfaces pointer
IBSH	Shading data block beginning pointer
AA(MX+12)	X-origin of surface
AA(MX+13)	Y-origin of surface
AA(MX+14)	Z-origin of surface
AA(MX+15)	Azimuth angle of surface (deg)
AA(MX+16)	Tilt angle of surface (deg)
DTOR	Degree to radian conversion factor
AREA	Area of surface (ft^2)
TILT	Tilt angle of surface (radians)
AZIM	Azimuth angle of surface (radians)
BAZIM	The building azimuth angle (radians)
PERIM	Perimeter (ft)
AA(J+20)	Shadow ratios
IA(MP+7)	Number of response factors
AA(MP+10)	Response factor sum (X)
AA(MP+11)	Response factor sum (Y)
AA(MP+9)	Response factor common ratio
AA(IR+12)	
AA(IR+13)	
AA(MX+7)	External surface multiplier
AA(MP+8)	External surface U-factor
MA	Number of attachments

IA(MA)	Number of attachments
IA(MA+3)	Attachment geometry data pointer
IA(MA+4)	Attachment properties data pointer
IA(MA+6)	Attachment shading data pointer
AA(MA+13)	Attachment X-coordinate on wall
AA(MA+14)	Attachment Y-coordinate on wall
AA(JJ+12)	Shadow data
IA(MR+6)	Number of panes of glass
AA(MA+7)	Attachment multiplier
IA(MX+11)	Length of external surface data entry

Output

Name	Description
XWO	X-origin of surface
YWO	Y-origin of surface
ZWO	Z-origin of surface
AWO	Azimuth angle of surface (radians)
TWO	Tilt angle of surface (radians)
SAW	Sine of surface azimuth angle
CAW	Cosine of surface azimuth angle
STW	Sine of surface tilt angle
CTW	Cosine of surface tilt angle
AA(MX+10)	Gross area of surface (ft^2)
AA(MX+12)	Net area of surface (ft^2)
AA(MX+20)	Perimeter (ft)
NRESF	Number of response factors
XSMULT	External surface multiplier
UXS	External surface U-factor
ATARE	Area of surface attachments (ft^2)
NATT	Number of attachments
XW	Attachment X-coordinate on wall
YW	Attachment Y-coordinate on wall
VATT	Attachment U-factor
ZCOND	Total space conductance (Btu/hr)

Calculation Procedure

1. Determine pointers for number of surfaces, surface geometry, surface properties, number of attachments, attachment geometry, and attachment properties.

```
MX = IA(MS+40) = IA(MZ+40) + IBEW
IE2 = IA(MX)
DO 4000 IE = 1,IE2
MG = IA(MX+3) = IA(MX+3) IBGE
MGS = MG
MP = IA(MX+4) = IA(MS+4) + IBPR
IF (IA(MX+5) .NE. 0) IA(MX+5) = IA(MX+5) + IBAT
MA = IA(MX+5)
IF (IA(MX+6) .NE. 0) IA(MX+6) = IA(MX+6) + IBSH
MH = IA(MX+6)
```

2. Determine the coordinates, azimuth and tilt angles, and their sines and cosines.

```
XWO = AA(MX+12)
YWO = AA(MX+13)
ZWO = AA(MX+14)
AWO = AA(MX+15) * DTOR
TWO = AA(MX+16) * DTOR
SAW = SIN (AWO)
CAW = COS (AWO)
STW = SIN (TWO)
CTW = COS (TWO)
```

3. Call the geometry preparation subroutine GEOPR1 and set for wall. Return with data.

```
CALL GEOPR1 (0)
AA(MX+10) = AREA
AA(MX+12) = AREA
AA(MX+16) = TILT
AA(MX+17) = SIN (TILT)
AA(MX+18) = COS (TILT)
AA(MX+15) = AZIM
AA(MX+13) = SIN (AZIM)
AA(MX+14) = COS (AZIM)
AA(MX+19) = AMOD (BAZIM + AZIM, 6.2831853)
AA(MX+20) = PERIM
```

4. Call the subroutine FILLN and place the 24 hrs of shadow ratio data in the array.

```
J = MX + 1
CALL FILLN (0., AA(J+20), 24)
```

5. Preinitialize the response factors to provide a relevant starting position for the beginning of the LOADS run period. From this point the first day of the run will cycle three times to stabilize the response factor values before proceeding.

```
NRESF = IA(MP+7)
IF (NRESF .EQ. 0) GO TO 700
CALL FILLN (DBTR, AA(MX+49), NRESF)
IR = MP + NRESF * 3 - 3
DT = DBTR - 535.
AA(MX+47) = (AA(MP+10) + AA(IR+12)/(1. - AA(MP+9))) * DT
AA(MX+48) = (AA(MP+11) + AA(IR+13)/(1. - AA(MP+9))) * DT
AA(MX+46) = AA(MP+11) * DT
700 CONTINUE
```

6. Determine the number of external surfaces and the U-factor.

```
XSMULT = AA(MX+7)
UXS = AA(MP+8)
```

7. If there are no response factors available, recalculate UXS.

```
IF (IA(MP+7) .EQ. 0) UXS = UXS/(1. + .333 * UXS)
```

8. Initialize and prepare the attachments data.

```
ATARE = 0.
IF (MA .EQ. 0) GO TO 1920
NATT = IA(MA)
DO 1900 K = 1,NATT
MG = IA(MA+3) = IA(MA+3) + IBGE
IA(MA+4) = IA(MA+4) + IBPR
MR = IA(MA+4)
IF (IA(MA+6) .NE. 0) IA(MA+6) = IA(MA+6) + IBSH
```

9. Determine the X and Y coordinates of the attachment location.

```
XW = AA(MA+13)
YW = AA(MA+14)
```

10. Call the geometry preparation routine and set for attachment calculation and, if the attachment is inset, call the setback routine.

```
CALL GEOPR1 (1)
IF (AA(MA+10) .GT. 0.) CALL SETBAC
AA (MA+9) = AREA
AA(MA+10) = PERIM
```

11. Fill shadow data array.

```
JJ = MA + 1
CALL FILLN (0., AA(JJ+12), 24)
```

12. Determine the attachment U-factor, area, and total zone conductance values.

```
UAT = 1.1
IF (IA(MR+6) .GT. 1) UATT = 0.6
ZCOND = ZCOND + AREA * UATT * XSMULT * AA(MA+7)
ATARE = ATARE + AREA * AA(MS+7)
IF (AA(MA+7) .EQ. 0) ATARE = ATARE + AREA
MA = MA + MALEN
1900 CONTINUE
IF (AA(MX+12) .GE. ATARE) GO TO 1920
```

13. Print any error messages.

```
PRINT 901, IA(MX+1), IA(MX+2)

901 FORMAT(*___ ERROR IN LOADS DATA ___ AREA OF SURFACE * 2R8 * IS NEGATIVE
*)

IERRFL = 1

1920 CONTINUE
```

14. Reset zone conductance value.

```
AA(MX+12) = AMAX1 (0., AA(MX+12) - ATARE)
ZCOND = ZCOND + AA(MX+12) * UXS * XSMULT
MX = MX + IA(MX+11)

400 CONTINUE
RETURN
```

ASHRAE Verification

This is a data preparation routine and has no equivalent in the ASHRAE documentation (Ref. 1).

10. Subroutine FILLN.

SUBROUTINE FILLN (C, X, N)

Description

FILLN is a utility routine used to fill an array with a desired constant. Blanks and array errors are therefore easily located.

Subprograms Calling This Subroutine

SUBROUTINE EXTPRP
SUBROUTINE DAYCLC

Subprograms Called By This Subroutine

None

Common Blocks

None

Declarations

DIMENSION X(N)

Input

Name	Description
C	Constant to be placed in array
X	Array name
N	Size of the array

Output

Name	Description
X(N)	The filled array

Calculation Procedure

1. Dimension the filled output array.

```
DIMENSION X(N)
```

2. Fill the array with the constant desired using a DO loop.

```
DO 200 I = 1,N
X(I) = C
```

```
200 CONTINUE
RETURN
END
```

ASHRAE Verification

FILLN has no equivalent ASHRAE algorithm.

11. Function FILM.

FUNCTION FILM (IS)

Description

This function determines the outside surface heat transfer coefficient as a function of wind velocity and the type of surface construction.

Subprograms Calling This Subroutine

SUBROUTINE CALEXT

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/WEATHD/	WSDSPD	None placed

Declarations

EQUIVALENCE (V,WNDSPD)

Input

Name	Description
V	Wind speed (knots)
IS	Exterior surface type index (dimensionless)

Output

Name	Description
FILM	The outside surface heat transfer coefficient (Btu/hr-ft^2-°F)

Calculation Procedure

$FILM = A * V^2 + B * V + C$

where

A, B, and C are given in Table III.5 according to the variable IS as selected by the user for a particular exterior surface.

TABLE III.5

ALGORITHMS COEFFICIENTS

Exterior Wall Type	IS	A(IS)	B(IS)	C(IS)
Stucco	1	0.	0.535	2.04
Brick and rough plaster	2	0.001329	0.369	2.20
Concrete	3	0.	0.380	1.90
Clear pine	4	-0.002658	0.363	1.45
Smooth plaster	5	0.	0.281	1.80
Glass, white paint on pine	6	-0.001661	0.302	1.45

ASHRAE Verification

Listed below are the two important variables contained in FILM and their counterparts and locations in ASHRAE (Ref. 1) and NECAP (Ref. 3). One is a user-selected input and the other is computed by the function.

Cal-ERDA Name	NECAP Name	NECAP Location	ASHRAE Name	ASHRAE Location	Equivalent
IS	IS	FILM	IS	FO	Yes
FILM	FILM	FILM	FO	FO	Yes

The only difference between the ASHRAE procedure and that of Cal-ERDA is in the conversion of the wind speed from knots to mph. The ASHRAE procedure first changes the value to mph and then computes the film coefficient, while the Cal-ERDA method uses the value in knots with modified equation coefficients. Thus, the algorithms are equivalent.

12. Subroutine GEOPR1.

SUBROUTINE GEOPR1 (IFLG)

Description

This subroutine prepares the necessary geometry data, depending on the type of surface input. The surface type is indexed by the variable IFLG.

Subprograms Calling This Subroutine

SUBROUTINE BSHDPR
SUBROUTINE EXTPRP

Subprograms Called by This Programs

SUBROUTINE APOL
SUBROUTINE ZONLOC
SUBROUTINE RECTAN

SUBROUTINE TRANSL
SUBROUTINE WALLOC

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/PNTRS/	MG	None placed
/PTRBGN/	Not currently used here	
/SURFD/	None obtained	NV, AREA, PERIM, AZIM, TILT, H, W
/ZWALOC/	AZO, AWO, TWO, XW, YW,	None placed

Declarations

```
COMMON AA(1000)
INTEGER IA(1000)
EQUIVALENCE (IA(1), AA(1))
```

Input

Name	Description
IFLG	Surface type index and calculation flag; 0 = wall, 1 = attachment translation, 2 = shade data
IA(MG)	Number of surface vertices
AA(MG+1)	X-coordinate
AA(MG+2)	Y-coordinate
AWO	Azimuth angle (radians)
AZO	Azimuth of space with respect to building coordinate system
TWO	Tilt angle (radians)

Output

Name	Description
AREA	Area of surface (ft^2)
PERIM	Perimeter of surface (ft)
KOSE	Number of vertices
H	Height of surface (ft)
W	Width of surface (ft)
AZIM	Azimuth angle of surface (radians)
TILT	Tilt angle of surface (radians)

Calculation Procedure

1. Determine number of vertices

```
NV = IA(MG)
```

2. Perform calculations according to number of vertices listed.

```
IF(NV-2) 4000, 5000, 6000
```

If the value of NV is less than two, it is an attachment translation only and the coordinates of a single vertex are passed.

```
4000 AREA = AA(MG+1)
PERIM = AA(MG+2)
GO TO 5800
```

If the data being prepared is for a wall or a shading surface, set the number of vertices to four and calculate data.

```
5000 IA(MG) = 4
KOSE = 4
MG = MG+3
H = AA(MG+1)
W = AA(MG+2)
MG = MG-3
AREA = H*W
PERIM = 2.*(H+W)
```

3. Convert two-dimensional coordinates into XYZ system.

```
Call RECTAN (H, W, AA(MG+1))
```

4. Determine azimuth and tilt angles.

```
5800 AZIM = AZO + AWO
TILT = TWO
GO TO 9000
```

5. According to the value of IFLG, call the various location and/or translation routines for the various surfaces.

```
9000 IF (IFLG.EQ.1) CALL TRANSL (XW, YW, 0., AA(MG+1), KOSE)
CALL WALLOC (AA(MG+1), KOSE)
IF (IFLG, NE.2) CALL ZONLOC (AA(MG+1), KOSE)
IF (IFLG. LT.2).AND. (NV. GT. 2)) CALL APOL (NV, AA(MG+1), AREA, AZIM,
  TILT)
RETURN
END
```

ASHRAE Verification

There is no equivalent ASHRAE energy calculation algorithm to the Cal-ERDA subroutine GEOPR1.

13. Subroutine HOLDAY.

Description

This subroutine identifies the national holidays of the United States for the year and sets a schedule index number.

Subprograms Calling This Subroutine

SUBROUTINE DAYCLC

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/DATE/	IDOY, ILEAPF, IDAY, IDOW, IMON	ISCDAY

Declarations

EQUIVALENCE (MO, IMON), (JAY, IDAY), (NDAY, IDOW)

Input

Name	Description
IDOY	The day of the year using Julian date, 1-366 (dimensionless)
ILEAPF	Leap year flag, either zero (0) or one (1) (dimensionless)
IMON	The month of the year, 1-12 (dimensionless)
IDAY	The day of the month, 1-31 (dimensionless)
IDOW	The day of the week, 1-7 (dimensionless)

Output

Name	Description
ISCDAY	The value of this variable sets the schedules to be used by the LOADS Program. The value may be from one to eight, where 1 - Sunday 2 - Monday 3 - Tuesday 4 - Wednesday 5 - Thursday 6 - Friday 7 - Saturday 8 - A holiday and is dimensionless.

Calculation Procedure

1. Initialize the local variable JOL.

   JOL = 0

   This local flag determines whether a specific day will be identified as a holiday. If JOL is greater than zero, the day is a holiday.

2. If the following days are encountered (holidays), the value of JOL is set greater than zero and the output variable ISCDAY is set equal to eight, which indicates a holiday.

   a. January 1
   b. January 2 if it is a Monday
   c. February 21 if it is a Friday
   d. February 22
   e. February 23 if it is a Monday
   f. May 29 if it is a Friday
   g. May 30
   h. May 31 if it is a Monday
   i. July 3 if it is a Friday

j. July 4
k. July 5 if it is a Monday
l. First Monday in September
m. Fourth Thursday in November
n. December 24 if it is a Friday
o. December 25
p. December 26 if it is a Monday
q. December 25 through December 27
r. December 31 if it is a Friday

3. If the above holidays are not encountered, the output variable is set equal to IDOW.

   ISCDAY = IDOW

   The schedules employed by LOADS are then selected according to their value.

   2-6 = Weekday
   7 = Saturday
   1 = Sunday

ASHRAE Verification

The Cal-ERDA subroutine HOLDAY is equivalent to both the NECAP subroutine HOLDAY (Ref. 3) and the ASHRAE algorithm HOLDAY (Ref. 1) in basic methodology. A difference does occur, however, in the list of holidays to be identified. The ASHRAE algorithm HOLDAY recognizes Columbus Day (second Monday of October) and Veteran's Day (fourth Monday of October) as national holidays, while the Cal-ERDA subroutine HOLDAY does not. Also, the Cal-ERDA version always recognizes December 24 through 27 as holidays, while the ASHRAE method does not.

14. Function IDOWK.

FUNCTION IDOWK (IYR, IMON, IDAY)

Description

This function calculates the day of the week and assigns a numerical value to each.

Subprograms Calling This Function

SUBROUTINE INITLZ (IERRF)

Subprograms Called By This Function

None

Common Blocks

None

Declarations

None

Input

Name	Description
IYR	The year (A.D.)
IMON	The month of the year, 1-12 (dimensionless)
IDAY	The day of the month, 1-31 (dimensionless)

Output

Name	Description
IDOWK	Index number for the day of the week, 1-7 (dimensionless) 1 = Sunday 2 = Monday 3 = Tuesday 4 = Wednesday 5 = Thursday 6 = Friday 7 = Saturday

Calculation Procedure

1. Set the local variables M and IY.

```
M = IMON - 3
IY = IYR
```

2. If IMON is past February, skip the next two calculations and set the local variable IC. If IMON is January or February, reset the local variables M and IY and set IC.

```
IF (IMON.GT.2) GO TO 100
M = M + 12
IY = IY - 1
100 IC = IY/100
```

3. Depending on the path taken in item 2 above, either initialize or reset the local variable IY.

```
IY = IY - 100 * IC
```

4. Calculate the local variable LDAY, which is a function of IC, IY, M, and IDAY.

```
LDAY = 21 * IC/4 + 5 * IY/4 + (13 + M + 2)/5 + IDAY + 2
```

5. Calculate the output variable IDOWK, which is a function of LDAY calculated in item 4 above.

```
IDOWK = 1 + LDAY - (LDAY/7) * 7
```

ASHRAE Verification

The Cal-ERDA function IDOWK provides the same output as the NECAP subroutine NDOW (Ref. 3) and the ASHRAE subroutine WKDAY (Ref. 1). It does not use the same method to calculate the day of the week. The Cal-ERDA function is based on a similar algorithm contained in a bibliography of algorithms compiled by the Association for Computing Machinery as listed in Ref. 11.

15. Function IDOYR.

Function IDOYR (IMON, IDAY, ILEAPF)

Description

This function calculates the day of the year (Julian date), 1-366, as a function of the month, the day of the month, and the leap year index.

Subprograms Calling This Function

SUBROUTINE INITLZ (IERRF)

Subprograms Called By This Function

None

Common Blocks

None

Declarations

```
DIMENSION MD (12)
DATA MD/0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334/
```

Input

Name	Description
IMON	The month of the year, 1-12 (dimensionless)
IDAY	The day of the month, 1-31 (dimensionless)
ILEAPF	The leap year index, 0 or 1 (dimensionless)

Output

Name	Description
IDOYR	The day of the year (Julian date), 1-366 (dimensionless)

Calculation Procedure

1. Set up a data array to provide a series of numbers that indicate the Julian date on which each successive month must end (January is initiated by zero.

   ```
   DIMENSION MD (12)
   DATA MD/0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334/
   ```

2. The day of the year, IDOYR, is then calculated by adding the current day of the month, IDAY, for the current month, IMON, to the Julian date of the last day for the previous month. This number is provided in DATA MD as shown in item 1 above.

   IDOYR = MD (IMON) + IDAY

   For example, March 6 would be

   IDOYR = 59 + 6 = 65 (Julian date).

3. If the month is greater than two, the leap year index, ILEAPF, is added.

```
IF (IMON. GT. 2) IDOYR = IDOYR + ILEAPF
```

ASHRAE Verification

The Cal-ERDA function IDOYR simply increments a Julian calendar by one for each new day analyzed. A subroutine of equivalent function does not exist in either NECAP (Ref. 3) or ASHRAE (Ref. 1). However, the data array MD of IDOYR on which the algorithm is keyed is identical to the data provided in the ASHRAE Subroutine DST.

16. Subroutine INITLZ.

SUBROUTINE INITLZ(IERRF)

Description

This subroutine is called to initialize each Program LOADS run. If a single continuous year is to be used, INITLZ(IERRF) will be called once.

Subprograms Calling This Subroutine

SUBROUTINE READSF (IERRF)

Subprograms Called By This Subroutine

FUNCTION IDOYR
FUNCTION IDOWK
SUBROUTINE WDREAD

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/CONST/	None obtained	DTOR, PIOVR2, PIOVR4,
/DATE/	None obtained	IYR, IMON, IDAY, IHR, IDOY, IDOW, ILEAPF
/LOCALD/	STALAT, BAZIM	SSTALA, CSTALA, TSTALA, SBAZIM, CBAZIM
/RUNTIM/	IDAYBG, IMONBG, IYRBG	None placed
/WEATHD/	Not currently used here	

Declarations

```
DATA DTOR/.0174532925/, PIOVR2/1.5707963/, PIOVR4/.78539816/
```

Input

Name	Description
IYRBG	Beginning year of LOADS run period
IMONBG	Beginning month of LOADS run period
IDAYBG	Beginning day of LOADS run period
STALAT	Station latitude (radians)
BAZIM	Building azimuth angle (radians)

Output

Name	Description
IYR	The calendar year
ILEAPF	Leap year flag, 1 = leap year
IDOY	The day of the year
IDOW	The day of the week
IMON	Beginning month of run period
IDAY	Beginning day of run period
IHR	The hour
SSTALA	Sine of latitude
CSTALA	Cosine of latitude
SBAZIM	Sine of building azimuth angle
CBAZIM	Cosine of building azimuth angle
IERRF	Error flag, 0 = no error

Calculation Procedure

1. Set data array.

```
DATA DTOR/.0174532925/, PIOVR2/1.5707963/, PTIOVR4/.78539816/
```

2. Initialize calendar for run period and leap year.

```
IYR = IYRBG
ILEAPF = 0
IF ((IYR/4)*4 .EQ. IYR) ILEAPF = 1
```

3. Initialize day, week, month, and hour values.

```
IDOY = IDOYR (IMONBG, IDAYBG, ILEAPF) -1
IDOW = IDOWK (IYR, IMONBG, IDAYBG) -1
IMON = IMONBG
IDAY = IDAYBG
IHR = 1
```

4. Initialize location data.

```
SSTALA = SIN (STALAT)
CSTALA = COS (STALAT)
TSTALA = SSTALA/CSTALA
SBAZIM = SIN (BAZIM)
CBAZIM = COS (BAZIM)
```

5. Read the weather data for the current day.

```
IDOY = IDOY + 1
CALL WDREAD
IDOY = IDOY - 1
```

6. Set the error flag if no error is encountered in weather file, and return.

```
IERRF = 0
RETURN
END
```

ASHRAE Verification

This is a simple subroutine to initialize the LOADS Program for each run. It has no equivalent in the ASHRAE literature for energy calculations.

17. Function MONLEN.

```
FUNCTION MONLEN (IMON, ILEAPF)
```

Description

The Cal-ERDA function MONLEN is a very simple algorithm to calculate the number of days in each month.

Program Calling This Function

PROGRAM LOADS

Subprogram Called By This Function

None

Common Blocks

None

Declarations

```
DIMENSION ML (12)
DATA ML/31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31/
```

Input

Name	Description
IMON	The month of the year, 1-12 (dimensionless)
ILEAPF	The leap year index, 0 or 1 (dimensionless)

Output

Name	Description
MONLEN	The number of days in the current month given by IMON (28-31 days)

Calculation Procedure

1. Set up a data array that contains the number of days for each of the 12 months of a nonleap year.

```
DIMENSION ML (12)
DATA ML/ 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31/
```

2. Determine number of days for each month, MONLEN, given by IMON, based on the data array ML shown in item 1 above.

```
MONLEN = ML (IMON)
```

3. If the month is February (IMON = 2), add the leap year index, ILEAPF, to the output variable MONLEN.

```
If (IMON .EQ. 2) MONLEN = MONLEN + ILEAPF
```

ASHRAE Verification

This is an extremely simple algorithm that merely extracts a number for a particular month from a data array. There are no comparable algorithms listed in either NECAP (Ref. 3) or ASHRAE (Ref. 1).

18. Subroutine READSF

SUBROUTINE READSF (IERRF)

Description

The subroutine READSF reads the LOADS standard file and prepares the data and data block pointers. It also produces an error flag (IERRF) if the standard file is not correct. This error flag will terminate the LOADS Program before any calculations are made.

Programs Calling This Subroutine

PROGRAM LOADS

Subprograms Called By This Subroutine

SUBROUTINE INITLZ
SUBROUTINE RMRSS
SUBROUTINE SETFLS
SUBROUTINE EXTPRP
SUBROUTINE UGPRP
SUBROUTINE BSHDPR
SUBROUTINE REMARK

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/FLAGS/	None obtained	HOLFLG, DSTFLG
/BLDVA/	None obtained	BLDVOL, BLDARE
/CLOCK/	None obtained	CLOCK array
/CONST/	None obtained	DTOR
/FILES/	ISTNDF	None placed
/GPNTRS/	None obtained	MGSHD, NZONT, NXST, NSCT, MSCTB, MZONTB, MXSTB
/IWCPTR/	None obtained	MIWC1, MIWC2, NIWC, LIWC
/LOCALD/	None obtained	STALAT, STALON, ITIMZ, BAZIM, BALTIT,
/PNTRS/	None obtained	MZLEN, MALEN, MPLEN, MRLEN, MULEN, MLAST, MP
/PTRBGN/	None obtained	IBSV, IBSC, IBPR, IBGE, IBSH, IBAT, IBEW, IBSP, IBUG, IBIW
/REPORT/	None obtained	None placed

/RUNPRD/	None obtained	NRUNPR, IRUNPR array, RUNSCA array
/RUNTIM/	None obtained	IDAYBG, IMONBG, IYRBG, IDAYND, IMONND, IYRND
/SIZE/	None obtained	LENGTH, LLLLLL, LSV, LSC, LPR LGE, LSH, LAT, LUG, LIW, LEX, LSP, NSP, NEX, NSC, MZLN, MALN, MPLN, MRLN, MULN, MILN, (ISZDUM(I), I=1,4)
/TITLE/	None obtained	ITITLE array, NTITLE
/WEATHD/	None obtained	IWDID array
/ZCOND/	None obtained	ZCOND
/ZWALOC/	None obtained	XZO, YZO, ZZO, AZO, SAZ, CAZ

Declarations

```
    DIMENSION ISIZ(25)
    EQUIVALENCE (ISIZ(1), LENGTH)
    COMMON AA(1000)
    INTEGER IA(1000)
    EQUIVALENCE (IA(1), AA(1)
```

Input

Name	Description
ISTNDF	LOADS standard file

Output

READSF reads the LOADS standard file, reading values into the variables in common blocks shown under "Common Blocks" above, and reading or calculating a number of values to the AA(IA) array. For a description of these variables, see the calculations described below and the LOADS Variable Master Index, Sec. III.A.4.

Calculation Procedure

1. Initialize: DTOR (degrees to radians) = 0.01745.

   IERRF = 0
   REWIND ISTNDF (standard file)

2. Read from the standard file CLOCK, ITITLE, NTITLE, NRUNPR, IRUNPR, RUNSCA, IWDID, STALAT, STALON, ITIMZ, BAZIM, BALTIT, HOLFLG, DSTFLG, ISIZ, MGSHD, IREPRT. ISIZ is a 25-word array indirectly equivalenced to the common block/SIZE/. Therefore, when ISIZ is read, all the variables in the common block are input. (See the description of these variables in the variable definition list, Sec. III.A.4.)

3. Check to see if the end of the standard file has been reached. If it has, go to an error-printing statement; call REMARK to print error message, set error flag to one, and return.
   IF (EOF(ISTNDF).NE.0) GO TO 80000

```
.
.
.
80000 PRINT 80001
80001 FORMAT (*0---ERROR IN LOADS---STANDARD FILE MISSING.*/)
CALL REMARK (43H-----ERROR-----LOADS STANDARD FILE MISSING)
GO TO 80900
.
.
.
80900 IERRF = 1
RETURN
```

4. Otherwise, check the length of the second record in the standard file. If it is less than 1000, default storage length will serve; if equal to or greater than 1000 the file storage length must be increased. System routines LOCF and SETFLS are used to establish this file length.

```
IF (LENGTH.LT.1000) GO TO 300
L = LOCF(AA) + LENGTH
CALL SETFLS(L)
300 CONTINUE
```

5. Read the second record of the standard file.

```
READ (ISTNDF) (IA(I), I=1, LENGTH)
```

If the end of the standard file has been reached, it is treated as in Step 3.

6. Some values read in Step 2 are assigned to other variables:

```
MZLEN = MZLN (length of zone data)
MALEN = MALN (length of attachment data)
MPLEN = MPLN (length of construction data)
MRLEN = MRLN (length of glass properties data)
MULEN = MULN (length of underground floors and walls data)
IDAYBG = IRUNPR(1,1) (beginning day of LOADS run)
IMONBG = IRUNPR(2,1) (beginning month)
IYRBG = IRUNPR(3,1) (beginning year)
IDAYND = IRUNPR(4,1) (ending day)
IMONND = IRUNPR(5,1) (ending month)
IYRND = IRUNPR(6,1) (ending year)
```

7. Station-related data are converted from input data in degrees to radians.

```
STALAT = STALAT  * DTOR (station latitude, degrees to radians)
STALON = STALON * DTOR (station longitude, degrees to radians)
BAZIM = BAZIM * DTOR (building azimuth angle, degrees to radians)
```

8. Subroutine INITLZ is called to perform initialization functions. If a nonzero error flag, IERRF, is returned from INITLZ, the program prints an error message, calls REMARK to write an error message, and returns.

```
CALL INITILZ (IERRF)
IF (IERRF.NE.0) GO TO 80200
.
.
.
80200 PRINT 80002
```

```
80002 FORMAT(80---ERROR IN LOADS-----BAD DATA IN STANDARD FILE*/)
CALL REMARK (39H-----ERROR----BAD LOADS STANDARD FILE)
80900 IERRF = 1
RETURN
```

9. Assign values to "number" variables.

```
NZONT = NSP (number of zones)
NXST = NEX (number of external surface  blocks)
NSCT = NSC (number of schedules)
```

10. Calculate pointers to the beginning of data blocks.

```
IBSV = 0                 (vector data block pointer)
IBSC = IBSV + LSV        (schedule data block pointer)
IBPR = IBSC + LSC        (properties data block pointer)
IBGE = IBPR + LPR        (geometry data block pointer)
IBSH = IBGE + LGE        (shade data block pointer)
IBAT = IBSH + LSH        (attachment data block pointer)
IBEW = IBAT + LAT        (external wall data block pointer)
IBSP = IBEW + LEX        (space data block pointer)
IBUG = IBSP + LSP        (underground floors and walls data block pointer)
IBIW = IBUG + LUG        (internal wall data block pointer)
MLAST = IBIW + LIW + 1   (last data pointer in common)
MSCTB = IBSC
MZONTB = IBSP
MXSTB = IBEW + 1
IBSC = IBSC - 1
```

11. KS is assigned the pointer to the beginning of the schedule data block.

```
KS = MSCTB
```

12. If the next entry in the IA array is zero, there is no more schedule data, and the program jumps to Step 13.

Otherwise, NSCB, current schedule vector pointer, is assigned IA(KS+3), and values for IA(KS+4,5, and 6) are calculated. KS is set to IA(KS+4), and the program returns to the beginning of this step.

```
2200 IF(IA(KS+1).EQ.0) GO TO 3000
NSCB = IA(KS+3)
IA(KS+4) = KS + 6 + 10*NSCB
IA(KS+6) = KS + 6 -10
IA(KS+5) = 0
KS = IA(KS+4)
GO TO 2200
300 CONTINUE
```

13. To process properties data, the variable MP is first assigned the properties data beginning pointer value, plus one.

```
MP = IBPR + 1
```

14. Another loop is entered that will continue until there are no more properties data (IA(MP).EQ.0).

   a. IF IA(MP+2).NE.1, i.e., property type is glass, the process skips to 14(b). Otherwise, MPLN1 = 3 * IA(MP+7) (number of response factors*3) MP is incremented by MPLEN,

```
MP = MP + MPLEN.
```

MPLN1 is checked; if it is greater than zero, the MP pointer is incremented.

```
IF (MPLN1.GT.0) = MP + MPLN1 + 3
```

Return to the beginning of the loop.

b. MP is simply incremented by MRLEN, length of glass properties entry, and returned to the beginning of the loop.

```
MP = MP + MRLEN
```

15. Space data are now adjusted. First BLDVOL, BLDARE, and MZ are initialized.

```
BLDVOL = BLDARE = 0.
MZ = MZONTB
```

16. a. A loop is entered on number of space data. Zone number is set to the loop index. ZCOND is initialized to zero.

```
DO 1000 = IZ = 1,NSP
IA(MZ+4) = IZ
ZCOND = 0.
```

b. The X, Y, and Z coordinates of the zone are assigned to XZO, YZO, and ZZO. The zone azimuth angle (in radians) and its sign and cosine are calculated. Value of heat transfer functions for underground floors and walls are set to zero.

```
XZO = AA(MZ+43)
YZO = AA(MZ+44)
ZZO = AA(MZ+45)
AZO = AA(MZ+46) * DTOR
SAZ = SIN(AZO)
CAZ = COS(AZO)
ZUGFUA = 0.
ZUGWUA = 0.
```

c. If this zone has external surfaces, EXTPRP is called to prepare the external surface data.

```
IF (IA(MZ+40).GT.0) CALL EXTPRP
```

d. If this zone has underground floors or walls, UGPRP is called to prepare that data and to return the heat transfer function values for those floors or walls.

```
IF (IA(MZ+42).GT.0) CALL UGPRP (ZUGFUA, ZUGWUA)
```

e. A running total of building area and of building volume are updated with the area and volume of this zone.

BLDARE = BLDARE + AA(MZ+7) * AA(MZ+14),

where AA(MZ+7) = number of zone multiplier and
AA(MZ+14) = zone floor area.

BLDVOL = BLDVOL + AA(MZ+7) * AA(MZ+15),

where AA(MZ+7) is as above, and AA(MZ+15) = zone volume.

f. Zone input CFM * zone is calculated.

A(MZ+38) = AA(MZ+49) * AA(MZ+14)

g. Zone lighting load, QQ (AA(MZ+25)), is calculated as (zone lights (Btu)) + [lights (W/ft^2) * zone floor area)/1000. * 3413 (Btu/kW-hr)]

QQ = AA(MZ+24) = (AA(MZ+24) + AA(MZ+48) * AA(MZ+14)/1000) * 3413.

h. Zone equipment load, i.e., total sensible load (Btu), is calculated as input equipment load plus (equipment load (W/ft^2) times square feet/1000.) * 3413 (Btu/kW-hr).

AA(MZ+28) = (AA(MZ+28) + AA(MZ+47) * AA(MZ+14)/1000.) * 3413

i. Subroutine RMRSS is called to supply the weighting factors to delay the heat transfer between this space and the building HVAC equipment. Variables sent to RMRSS are lighting fixture type, weight of the floor (lbs/ft^2), and the percentage of the heat generated by the lights that is retained in the space as a load. RMRSS returns an array of data equivalent to BX1M, AX0, AX1, BG1M, AG0, AG1, BIS1M, AIS0, AIS1, RCPM, RCP0, and RCP1. (See Sec. III.A.4, LOADS Variable Master Index, or the write-up of subroutine RMRSS for a definition of these variables.

CALL RMRSS (IA(MZ+25), AA(MZ+17), AA(MZ+26), AA(MZ+43))

j. Zone temperature for heating is converted from Fahrenheit (input) to Rankine, and TZONER is set to Rankine temperature.

AA(MZ+8) = AA(MZ+8) + 460.
TZONER = AA(MZ+8)

k. PPN = AA(MZ+20) (number of people)
PPA = AA(MZ+21) * 0.01 (latent activity level of people)

Sensible heat from people, AA(MZ+42) = number of people, PPN, times the quantity, (28. + PPA * (266.4-10.25 * PPA) + (TZONER - 460.) * (1.2 - PPA *(3.07 - 0.128 * PPA)))

Latent heat from people, AA(MZ+21), is then recalculated as number of people, PPN, times the quantity (206. - PPA *(214.9 - 13.8 * PPA) - (TZONER - 460.) * (6.7 - PPA * (4.44 - 0.222 * PPA)))

(These equations are taken directly from the NECAP Program.)

Zone lighting, AA(MZ+25) = QQ * AA(MZ+26)/100.

l. If there is a schedule for occupants, the length of that schedule is incremented by IBSC, schedule pointer. If there is a schedule for lighting, the length of that schedule is incremented by IBSC. Similarly, schedule lengths for equipment, special equipment, and infiltration are incremented.

```
IF(IA(MZ+19).NE.0) IA(MZ+19) = IA(MZ+19) + IBSC
IF(IA(MZ+23).NE.0) IA(MZ+23) = IA(MZ+23) + IBSC
IF(IA(MZ+27).NE.0) IA(MZ+27) = IA(MZ+27) + IBSC
IF(IA(MZ+31).NE.0) IA(MZ+31) = IA(MZ+31) + IBSC
IF(IA(MZ+37).NE.0) IA(MZ+37) = IA(MZ+37) + IBSC
```

m. Heat transfer values for underground floors and walls are assigned to the appropriate array elements; zone internal surface heat gain is initialized to zero; and total space conductance is incremented by underground floor and wall values for this zone, That value is assigned to the appropriate array element.

```
AA(MZ+26) = ZUGFUA
AA(MZ+22) = ZUGWUA
AA(MZ+4) = 0.
ZCOND = ZCOND + ZUGFUA + ZUGWUA
AA(MZ+6) = ZCOND
```

The MZ pointer is reset, and the loop continues at 16a.

```
MZ = MZLEN
10000 CONTINUE
```

17. Subroutine BSHDPR is called to prepare the building shade data for use by LOADS.

18. Internal wall data is now prepared. The pointer to this data is set.

```
MI = IBIW + 1
NIW = IA(MI), number of internal walls in the building
```

Other pointers are set.

```
NIWC = NIW
MIWC1 = MI
LIWC = MIWC1-1
```

19. If number of internal walls is zero, skip to Step 20. Otherwise,

a. Loop IW = 1 to number of internal walls.

b. MP, property pointer, = IA(MI+6) + IBPR (pointer to beginning of properties data).

c. U-factor of the surface is multiplied by surface area and the number of identical surfaces ("surface multiplier"),

```
C = AA(MZ+7) * AA(MI+5) * AA(MP+8)
```

and an array element is assigned this value.

```
AA(LIWC+3) = C
```

d. Two more data pointers are set.

```
MZ1 = MZ = IA(MI+3) + IBSP - 1
```

e. TZ1 = AA(MZ+8) (zone heating temperature)
IA(LIWC+2) = IA(MZ+4) (property table data pointer)

f. Q = C * AA(MZ+8) (new zone temperature) - TZ1 (old zone temperature) and AA(MZ+41) = AA(MZ+41) - Q (zone internal surface heat gain)

g. MZ is reset to MZ1; and AA(MZ+41), with this new MZ pointer, is set back to its original value.

```
AA(MZ+41) = AA(MZ+41) + Q
```

h. MI and LIWC pointers are incremented.

```
MI = MI + MILN
LIWC = LIWC + 3
```

i. The loop continues at Step 19a.

20. A pointer is incremented.

LIWC = LIWC + 1.

An array value is set to zero to indicate end of data;

IA(LIWC) = 0

and, other pointers are reset;

MIWC2 = LIWC
LIWC = LIWC = MIWC1 + 1

and return.

ASHRAE Verification

Since subroutine READSF is a data preparation routine for the use of LOADS data, there is no applicable ASHRAE algorithm.

19. Subroutine RECTAN.

SUBROUTINE RECTAN (H, W, XYZ)

Description

The Cal-ERDA subroutine RECTAN puts two-dimensional rectangular coordinates into an XYZ system given the height (H) and the width (W). These may be walls or attachments to walls.

Subprograms Calling This Subroutine

SUBROUTINE GEOPR1

Subprograms Called by This Subroutine

None

Common Blocks

None

Declarations

DIMENSION XYZ(3,4)

Input

Name	Description
H	Height of the polygon (ft)
W	Width of the polygon (ft)

Output

Name	Description
XYZ	A 3x4 array containing the X, Y, and Z coordinates for each of the four vertices of a rectangle

Calculation Procedure

1. Dimension of XYZ output array.

   DIMENSION XYZ(3,4)

2. Using the inputs H and W, set the coordinates of each of the vertices. The vertices are set on an imaginary XY plane with the second vertex being at the origin. Z is zero in each case. See Fig. III.1 for detail.

3. RETURN

```
XYZ(1,1) = 0.
XYZ(2,1) = H            Vertex 1
XYZ(3,1) = 0.

XYZ(1,2) = 0.
XYZ(2,2) = 0.           Vertex 2 (origin)
XYZ(3,2) = 0.

XYZ(1,3) = W
XYZ(2,3) = 0.           Vertex 3
XYZ(3,3) = 0.

XYZ(1,4) = W
XYZ(2,4) = H            Vertex 4
XYZ(3,4) = 0.
```

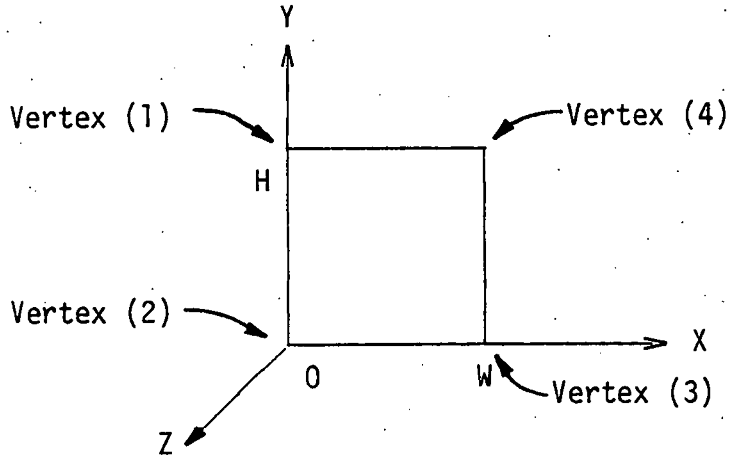


Fig. III.1. XYZ coordinate system.

ASHRAE Verification

There is no algorithm in the ASHRAE reference (Ref. 1) similar to the Cal-ERDA subroutine RECTAN.

20. LOADS Reports.

a. RPT1. This is the LOADS LO1 report titled "Space Peak Loads" and is the default report provided by the program if no other option is specified. It provides a summary of the space peak loads, giving total heating load, total cooling load, and their respective times of occurrence.

b. RPT2. This is the LOADS LO2 report titled "Space Peak Load Components." It is not a default report and must be requested by the user. It provides, for each space, the individual components of the loads (sensible and latent) including dry-bulb and wet-bulb temperatures, time of occurrence, and the area and volume of the space. The individual load components reported are walls, ceilings, glass (conduction and solar), internal surfaces, underground surfaces, occupants, lights, equipment, processes, and infiltration. It does not include outside ventilation air. The routine calls RPT2H, RPT2S, and RPT3.

c. RPT3. This routine is similar to RPT1 except that the total building peak heating and cooling loads are given instead of individual space loads.

d. RPT2H. This is simply a printing routine similar to RPT2S.

e. RPT2S. This subroutine is called by RPT2 to print the titles and values of the space peak load components.

21. Subroutine RMRSS

SUBROUTINE RMRSS (IL, W, PERCT, A)

Description

The subroutine RMRSS supplies the weighting factors to delay the heat transfer between a space and the building HVAC equipment.

Subprograms Calling This Subroutine

SUBROUTINE READSF

Subprograms Called by This Subroutine

None

Common Blocks

None

Declarations

DIMENSION A(12)

```
DIMENSION RMRG(3,3), RMRX(3,3), RMRST(3,4,3)
DATA RMRG/0.224, 0.197, 0.187, -0.044, -0.067, -0.097, 0.82, 0.87, 0.91/
DATA RMRX/0.703, 0.681, 0.676, -0.523, -0.551, -0.586, 0.82, 0.87, 0.91/
DATA RMRSI/0.53, 0.53, 0.53, 0.59, 0.59, 0.59, 0.87, 0.87, 0.87, 0.50, 0.50,
           0.50, -0.35, -0.40, -0.44, -0.41, -0.46, -0.50, -0.69, -0.74,
           -0.78, -0.32, -0.37, -0.41, -0.82, 0.87, 0.91, 0.82, 0.87,
           0.91, 0.82, 0.87, 0.91, 0.87, 0.91, 0.82/
```

Input

Name — Description

IL — The lighting fixture type

= 1, Fluorescent fixtures recessed into a suspended ceiling, plenum is not vented.

= 2, Fluorescent fixtures recessed into a suspended ceiling, return air through ceiling plenum.

= 3, Fluorescent fixtures recessed into a suspended ceiling, supply and return air through fixtures.

= 4, Incandescent lights exposed in the room air.

(Fig. III.2 details these different lighting fixture types (IL).)

W — The weight of the floor (lbs/ft^2)

PERCT — The percentage of the heat generated by the lights that is retained in the space as a load with (100-PERCT) of the heat energy to the return plenum

Output

Name — Description

A — A scalar array of 12 that returns the weighting factors for solar gains through glass, conduction gains through walls and floors, and gains caused by lighting within the space.

A(1) = BX1M = weighting factor for heat gain through walls and floors

A(2) = AX0 = weighting factor for heat gain through walls and floors

A(3) = AX1 = weighting factor for heat gain through walls and floors

A(4) = BG1M = weighting factor for solar heat gain through glass

A(5) = AG0 = weighting factor for solar heat gain through glass

A(6) = AG1 = weighting factor for solar heat gain through glass

A(7) = BAS1M = weighting factor for heat gain from lights

A(8) = AIS0 = weighting factor for heat gain from lights

A(9) = AIS1 = weighting factor for heat gain from lights

A(10) = RCPM = weighting factor for immediate heat gain to plenum

A(11) = RCP0 = weighting factor for immediate heat gain to plenum

A(12) = RCP1 = weighting factor for immediate heat gain to plenum

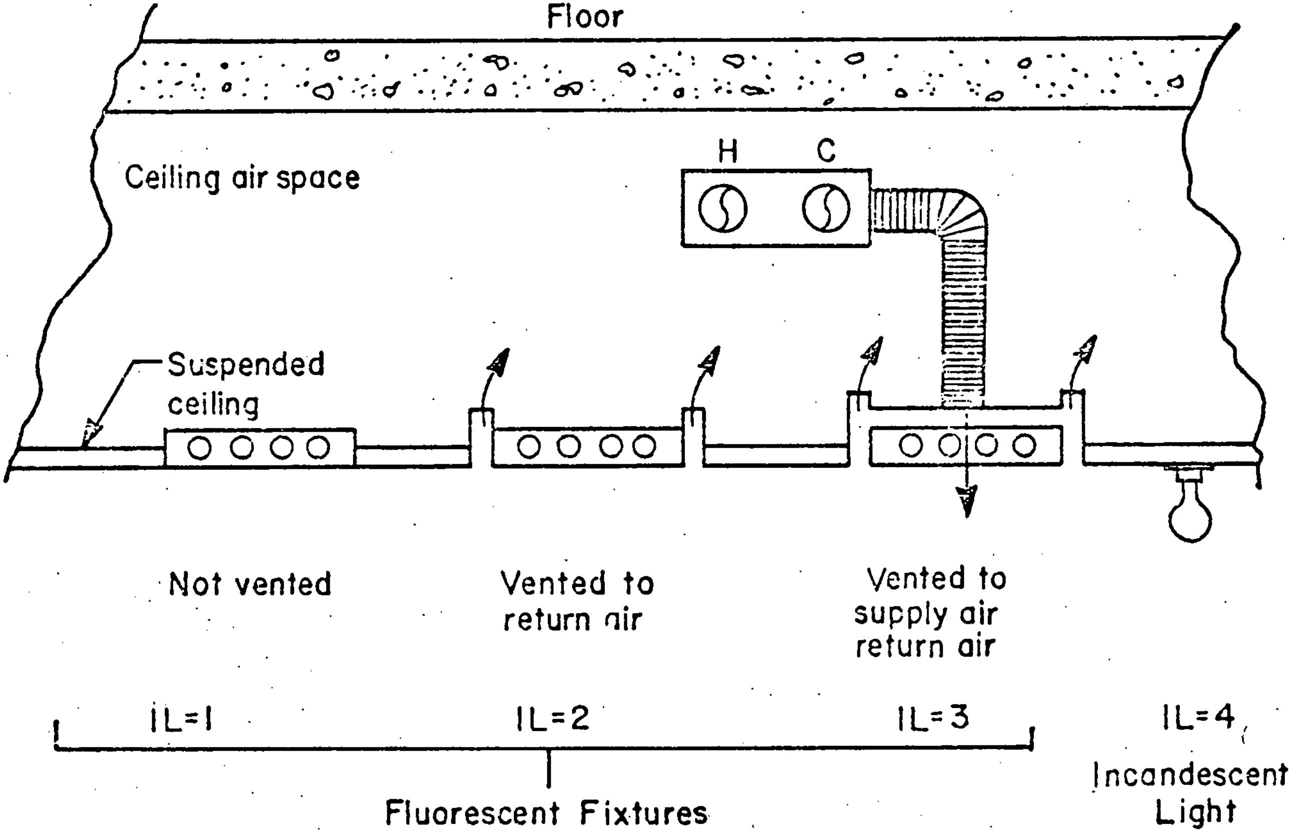


Fig. III.2 Lighting fixture type detail.

Calculation Procedure

1. Dimension the scalar output array A.

2. Dimension the weighting factor data arrays RMRG, RMRX and RMRSI.

   DIMENSION RMRG(3,3), RMRX(3,3), RMRSI(3,4,3)

3. Set data arrays for weighting factor coefficients.

   DATA RMRG/....(see Declarations section above)
   DATA RMRX/....
   DATA RMRSI/....

   Tables III.6, III.7, and III.8 give the coefficients according to type (glass solar gain, conduction, and lights) and construction-type index.

TABLE III.6

WEIGHTING FACTOR COEFFICIENTS FOR SOLAR GAIN THROUGH GLASS

WEIGHTING FACTOR SYMBOL	TYPE OF CONSTRUCTION		
	LIGHT	MEDIUM	HEAVY
AGO	0.224	0.197	0.187
AG1	-0.044	-0.067	-0.097
BGIM	0.82	0.87	0.91

TABLE III.7

WEIGHTING FACTOR COEFFICIENTS FOR CONDUCTION TYPE

WEIGHTING FACTOR SYMBOL	TYPE OF CONSTRUCTION		
	LIGHT	MEDIUM	HEAVY
AXO	0.703	0.681	0.676
AXI	-0.523	-0.551	-0.586
BXIM	0.82	0.87	0.91

TABLE III.8

WEIGHTING FACTOR COEFFICIENTS FOR LIGHTING TYPE

WEIGHTING FACTOR SYMBOL	TYPE OF CONSTRUCTION		
	LIGHT IW = 1	MEDIUM IW = 2	HEAVY IW = 3
IL = 1 - Fluorescent fixtures recessed into suspended ceiling; ceiling plenum not vented.			
AISO	0.53	0.53	0.53
AIS1	-0.35	-0.40	-0.44
BISIM	0.82	0.87	0.91
IL = 2 - Fluorescent fixture recessed into suspended ceiling; return air through ceiling plenum.			
AISO	0.59	0.59	0.59
AIS1	-0.41	-0.46	-0.50
BISIM	0.42	0.87	0.91
IL = 3 - Fluorescent fixture recessed into suspended ceiling; supply and return air through fixtures.			
AISO	0.87	0.87	0.87
AIS1	-0.69	-0.74	-0.78
BISIM	0.82	0.87	0.91
IL = 4 - Incandescent lights exposed in the room air			
AISO	0.50	0.50	0.50
AIS1	-0.32	-0.37	-0.41
BISIM	0.82	0.87	0.91

4. Set the construction type index by floor weight.

```
IW = 3
IF (W .LE. 100) IW = 2
IF (W .LE. 50) IW = 1
```

5. Set the weighting factors for solar heat gain through glass.

```
AGO = RMRG (IW,1)
AG1 = RMTG (IW,2)
BG1M = RMRX (IW,3)
```

6. Set the weighting factors for heat gains through walls and floors.

```
AXO = RMRX (IW,1)
AX1 = RMRX (IW,2)
BX1M = RMRX (IW,3)
```

7. Check for limits on light-fixture types.

```
IF ((IL .LT. 1) .OR. (IL. GT. 4)) IL = 1
```

8. Check for nonvented lights.

```
IF ((IL .EQ. 1) .OR. (IL .EQ. 4)) PERCT = 100.0
```

9. Set the weighting factors for interior space heat gain caused by lights.

```
AISO = (PERCT/100.0) * RMRSI (IW,IL,1)
AIS1 = (PERCT/100.0) * RMRSI (IW,IL,2)
BIS1M = RMRSI (IW,IL,3)
```

10. Set the weighting factors for interior space heat gain to plenum.

```
IF ((IL .EQ. 1) .OR. (IL .EQ. 4)) GO TO 100
RCPO = ((100.0-PERCT)/100.0)* RMRSI(IW,IL,1)
RCP1 = ((100.0-PERCT)/100.0)* RMRSI(IW,IL,2)
RCPM = RMRSI (IW,IL,3)
GO TO 900
```

If no plenum,

```
100 RCPO = 0.0
RCP1 = 0.0
RCPM = 0.0
```

11. Produce the scalar output array A.

```
900 CONTINUE
A(1) = BX1M
A(2) = AXO
A(3) = AX1
A(4) = BG1M
A(5) = AGO
A(6) = AG1
A(7) = BIS1M
A(8) = AISO
A(9) = AIS1
A(10)= RCPM
A(11)=RCPO
A(12)=RCP1
```

```
RETURN
END
```

ASHRAE Verification

The Cal-ERDA subroutine RMRSS is virtually identical to the ASHRAE algorithm HLC (Ref. 1) and is exactly the same as the subroutine RMRSS in NECAP (Ref. 3). Table III.9 lists the important variables in Cal-ERDA subroutine RMRSS and their equivalents in ASHRAE and NECAP

TABLE III.9

IMPORTANT VARIABLES IN CAL-ERDA SUBROUTINE RMRSS AND THEIR EQUIVALENTS IN ASHRAE AND NECAP.

Cal-ERDA Name	NECAP Name	NECAP Location	ASHRAE Name	ASHRAE Location	Equivalent
AGO	RMRG1	RMRSS	AG_0	HLC	YES
AG1	RMRGC	RMRSS	AG_1	HLC	YES
AGIM	RATRG	RMRSS	$(-)BG_1$	HLC	YES
AXO	RMRX1	RMRSS	AX_0	HLC	YES
AX1	RMRXC	RMRSS	AX_1	HLC	YES
BXIM	RATRX	RMRSS	$(-)BX_1$	HLC	YES
AISO	RMRIS1	RMRSS	AIS_0	HLC	YES
AIS1	RMRISC	RMRSS	AIS_2	HLC	YES
BISIM	RATRIS	RMRSS	$(-)BIS_1$	HLC	YES
RCPO	RMRPS1	RMRSS	N/A	-	-
RCP1	RMRPSC	RMRSS	N/A	-	-
RCPM	RATRPS	RMRSS	N/A	-	-

Note that the terms $(-)BG_1$, $(-)BX_1$, and (-)BIS indicate that Cal-ERDA subroutine RMRSS is employing the negative value of the ASHRAE equivalent.

There are two variations from the ASHRAE documentation (Ref. 1) in Cal-ERDA RMRSS. The first concerns the calculation of the variables RCPO, RCP1, and RCPM. These values are based on the above weighting factors and determine the amount of energy being passed to the plenum system from the lights. These simple calculations are mentioned, but the details are not shown in ASHRAE (see footnote p. 131, Ref. 1).

The second variation involves the number of weighting factors employed for each type of heat gain considered. ASHRAE sets four for each type, while Cal-ERDA and NECAP use only three of these. The missing factors are BG_0, BX_0, and AIS. In each case, the factor is a constant for all three structure types, light, medium, and heavy, with $BG_0 = 1.00$, $BX_0 = 1.00$, and AIS = 0.87.

The procedure employed by ASHRAE and, therefore, by Cal-ERDA and NECAP, was developed by G. P. Mitalas and D. G. Stephenson (Refs. 7 and 8). A detailed description of the technique may be found in the appendix of Ref. 1.

22. Subroutine SETBAC.

Description

Subroutine SETBAC calculates the vertex coordinates for the vertices of three added shading surfaces surrounding a window and the sides and top surfaces created by the setback of the window from the wall surface. The window must be a rectangle, and only windows are handled by this routine.

Subprograms Calling This Subroutine

SUBROUTINE EXTPRP

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/PNTRS/	MLAST, MA, MX, MG	MG
/SIZE/	None obtained	LENGTH
/SURFD/	NV, H, W	None placed

Declarations

DATA MHLEN/2/

Input

Name	Description
NV	Number of vertices
MLAST	Last data pointer in common block
MA	Attachment data pointer
MX	External surface pointer
MG	Geometry data pointer
AA(MA+10)	Setback depth
AA(MX+13)	Sine of the azimuth angle
AA(MX+14)	Cosine of the azimuth angle

AA(MX+17)	Sine of the tilt angle
AA(MX+18)	Cosine of the tilt angle
AA(MG+1)	X-coordinate
AA(MG+2)	Y-coordinate
AA(MG+3)	Z-coordinate
H	Height (ft)
W	Width (ft)

Output

Name	Description
IA(MA+6)	Shade surface pointer
IA(MH)	Number of shading surface
AA(MH+1)	Transmissivity of shading surface
IA(MH+2)	Shade geometry data pointer
IA(MG)	Number of vertices
AA(MG+1)	X-coordinate
AA(MG+2)	Y-coordinate
AA(MG+3)	Z-coordinate

Calculation Procedure

1. If number of vertices ≠ 2, return.

```
IF(NV.NE.2) GO TO 9000
```

2. Set intermediate variable INC equal to (shading surface data entry length, plus 4(vertices), times 3(coordinates), plus 1) times 3(surfaces) plus 1. This figure is added to the last data pointer in the common block and is used in operating system subroutine SETFLS to set field length.

```
INC = 3 *(MHLEN + 4 * 3 + 1)+1
L = MLAST + INC
CALL SETFLS (LOCF (AA(L)))
```

3. Determine LENGTH, MH, and IA(MA+6). MH and IA(MA+6) will contain the old value of the MLAST pointer in common, while MLAST itself and LENGTH are incremented by the field length that will be needed to contain the newly calculated vertex data.

```
MH = MLAST
MLAST = MLAST + INC
LENGTH = MLAST
IA(MA+6) = MH
```

4. Set intermediate variables

```
D = AA(MA+10) (setback)
B = 0.        (border)
SA = AA(MX+13)(Y-origin)
CA = AA(MX+14)(Z-origin)
```

ST = AA(MX+17) (Sine of tilt angle)
CT = AA(MX+18) (Cosine of tilt angle)

5. Increment MG by three to point to first vertex.
Set X, Y, and Z coordinates.

X2 = AA(MG+1)
Y2 = AA(MG+2)
Z2 = AA(MG+3)

6. Window height is incremented by B and width by 2B.

HH = H + B
WW = W + B + B

(Since B has been set to zero, no change is made. In the NECAP program, the equivalent variable represents a border around the glass, In subroutine SETBAC of Cal-ERDA, however, no border (B = 0.) is taken into account.)

HST = H + ST (height + SIN(tilt angle))

7. Set X, Y, and Z coordinate to true value, taking into account the width of the border, B. (Since B=0, this step is redundant.)

X = X2 + B * CA
Y = Y2 - B * SA
Z = Z2

8. Calculate height and width of window and setback in terms of the building coordinate system, using sine and cosine of the azimuth angle of the surface.

HCT = HH * CT(height * COS(tilt angle))
DSA = D * SA(setback depth * SIN(azimuth angle))
DCA = D * CA(setback depth * COS(azimuth angle))
WCA = WW * CA(width * COS(azimuth angle))
WSA = WW * SA(width * SIN(azimuth angle))

9. Set number of shading surfaces, IA(MH) = 3.
Set MG, geometry data pointer, to shading surface data pointer plus three times the needed array length, plus 1.

MG = MH + 3 * MHLEN + 1

10. Set transmissivity of these shading surfaces to zero; set shade geometry data pointer to the new geometry data pointer; and set number of vertices to 4.

AA(MH+1) = 0.
IA(MH+2) = MG
IA(MG) = 4

11. a. The coordinates of the first vertex of the first of the three shading surfaces are the input corner of the window at the wall surface.

AA(MG+1) = X
AA(MG+2) = Y
AA(MG+3) = Z

b. Increment MG data pointer to next vertex.

MG = MG + 3

The second vertex is at the same height as the first, but set back from the wall surface. Setback is expressed in terms of the building coordinate system.

AA(MG+1) = X + DSA (X plus (±) setback)
AA(MG+2) = Y + DCA (Y plus (∓) setback)
AA(MG+3) = Z

c. Increment MG pointer by 3.
Calculate the coordinates of the third vertex. This vertex is above or below the second, with allowance made for the tilt of the window from vertical.

AA(MG+1) = X + DSA - HCT * SA
AA(MG+2) = Y + DCA - HCT * CA
AA(MG+3) = Z + HST

d. Increment MG pointer by 3.
Calculate coordinates of the fourth vertex that is at the same height as the third, but at the wall surface.

AA(MG+1) = X - HCT * SA
AA(MG+2) = Y - HCT * CA
AA(MG+3) = Z + HST

12. Increment MG pointer by 4, and increment MH pointer to the next shading surface. Set shade geometry data pointer to MH. Set transmissivity of the new shading surface to 0, and indicate four vertices for the new surface.

MG = MG+4
MH = MH + MHLEN
IA(MH+2) = MG
AA(MH+1) = 0.
IA(MG) = 4

This shading surface is horizontal. MG is incremented by 3 for each vertex.

a. and b. The coordinates of the first and second vertices of the second shading surface are calculated precisely the same as the last two vertices of the previous surface.

c. and d. The coordinates of the third and fourth vertices are at the same height as the second. The third vertex is setback from the wall, and the fourth is at the wall surface.

MG = MG + 3
AA(MG+1) = X - HCT * SA + DSA - WCA
AA(MG+2) = Y - HCT * CA + DCA + WSA
AA(MG+3) = Z + HST
MG = MG+3
AA(MG+1) = X - HCT * SA - WCA
AA(MG+2) = Y - HCT * CA + WSA
AA(MG+3) = Z + HST

13. MG, MH, IA(MH+2), AA(MH+1), and IA(MG) are reset as in Step 12.

a. The first vertex of the third shading surface is at the wall surface, above or below the vertex calculated in Step 12d.

AA(MG+1) = X - WCA
AA(MG+2) = Y + WSA
AA(MG+3) = Z

b. The second vertex of this surface is located as the one in Step 13a, except that it is setback from the wall surface.

MG = MG + 3
AA(MG+1) = X - WCA + DSA
AA(MG+2) = Y + WSA + DCA
AA(MG+3) = Z

c. and d. The last two vertices of the surface are calculated precisely the same as those in Steps 12c and 12d.

Figure III.3 shows an example of these calculations, using the lower-left-hand corner of the window as the input vertex and the glass as not vertical.

ASHRAE Verification

The Cal-ERDA subroutine SETBAC is identical to the NECAP subroutine SETBAC (Ref. 3) and simply generates setback data for windows. As such, it has no direct equivalent in the ASHRAE documentation (Ref. 1). This same type of information is produced in the ASHRAE algorithm SHADOW1.

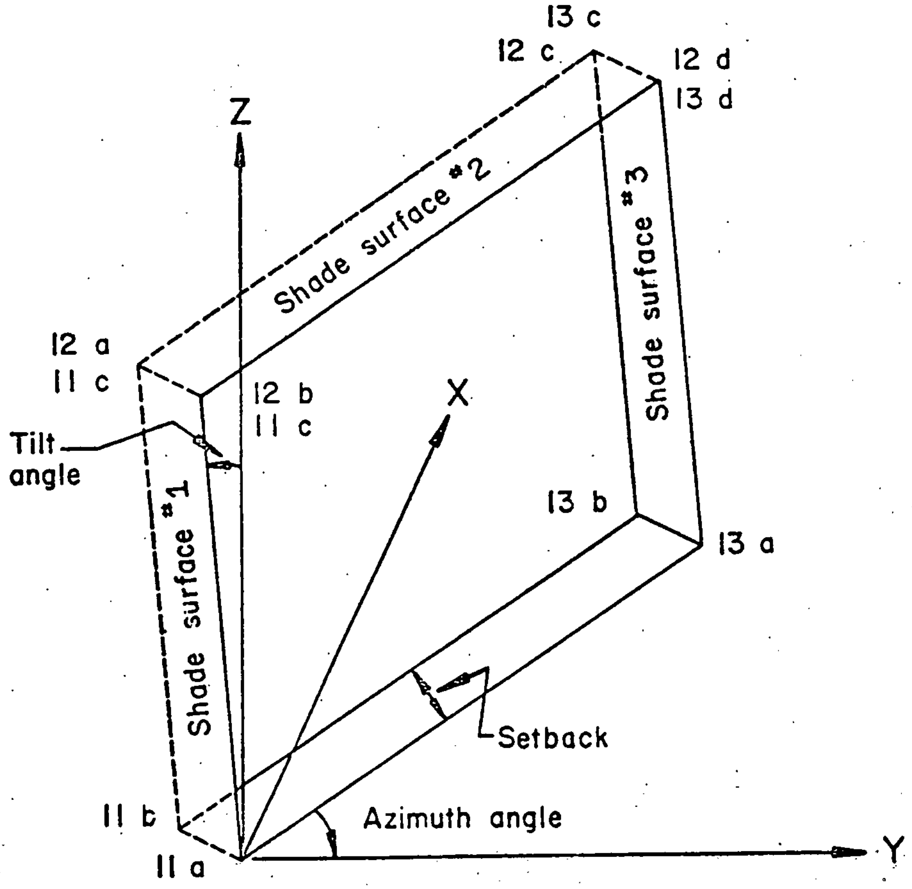


Fig. III.3. Subroutine SETBAC calculations.

23. Shadow Calculations.

The Cal-ERDA Program can account for shading effects on the building loads calculations. A set of seven subroutines is employed to produce shadow ratios on the building external surfaces and its attachments. These are:

SHADOW - Control routine, initializes and outputs data

SHDWIN - Initializes the shadow calculation arrays with the receiving polygon

SHDWAR - Calculates the unshaded area

SHDPAR - Calculates the area of a polygon in the shadow calculations

SHDWS - Clips portions of shading polygons below the receiving polygon and unions it with the shaded surface.

SHDWUN - Unions polygon into shadow arrays ENDPNT and PRMPNT (polygon strip length and permeability values)

SHDWPR - Prints shadow picture

This group of shadow routines is not based on either NECAP (Ref. 3) or ASHRAE (Ref. 1) algorithms. It was developed by Consultants Computation Bureau (CCB) to provide accurate, fast shadow ratio calculations. Although the basic analytical geometry is the same in any of the procedures, the Cal-ERDA routines use a completely different methodology for determination of the cast shadow. The previous methods broke an area up into a set of squares, each with a centerpoint. A shadow line above or below that point would then determine if that square was shaded or clear. The result was a shadow ratio that produces a step-like shade line on an area. The Cal-ERDA method, however, employs a series of lines or strips applied to the area in question with an infinite number of points that the shadow line may cross. The result is a shade line that closely approximates the actual shading present on a surface. Figure III.4 shows the two methods graphically. It can be seen that Cal-ERDA produces a more realistic picture.

These shadow ratios are calculated once each month for each surface required. Normally, this is done on the first of the month, but will also be done on the first day of a run period if it does not coincide with the first of the month.

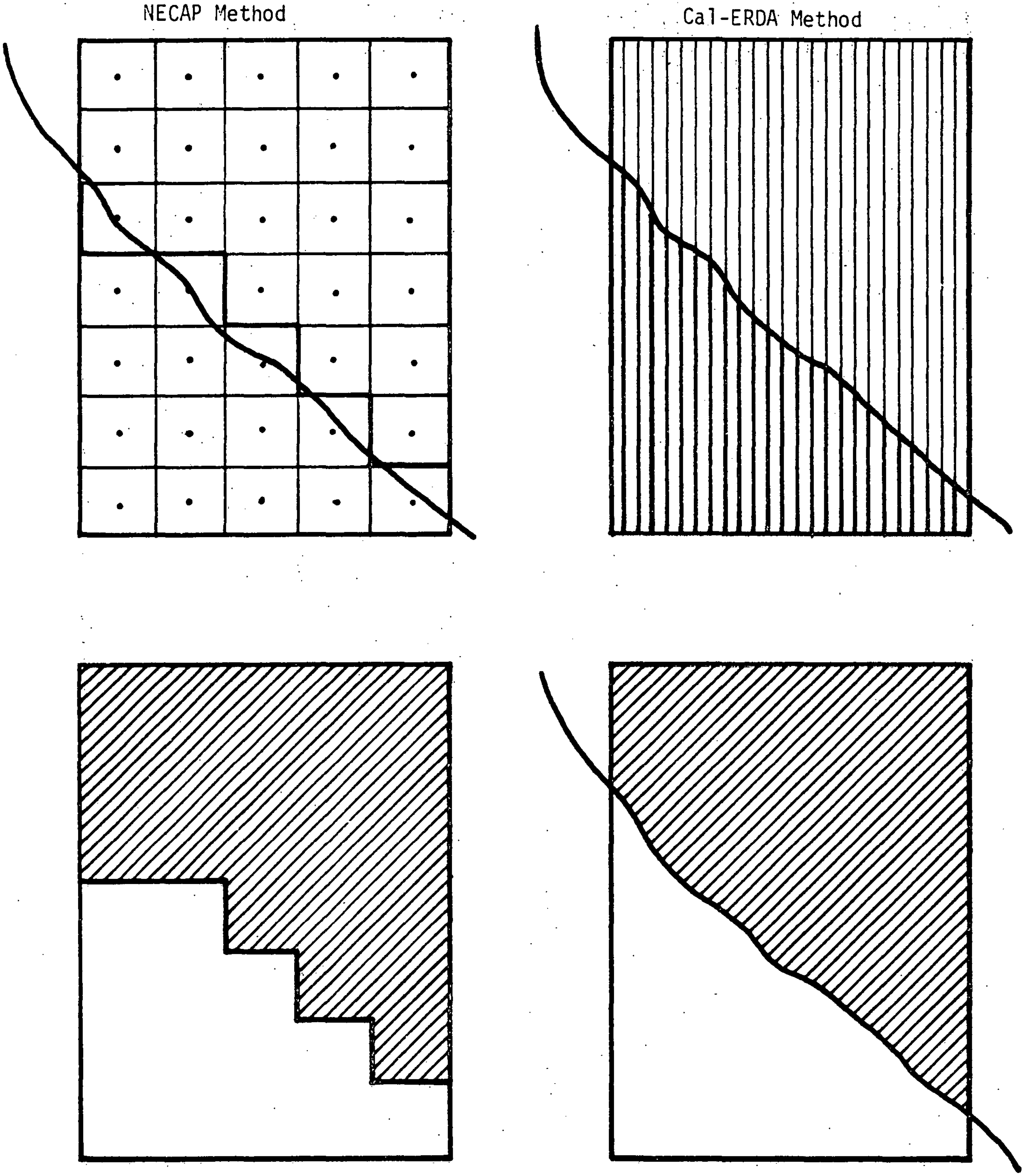


Fig. III.4. Shadow ratio methodology -
NECAP and Cal-ERDA.

24. Subroutine SUN1

Description

This subroutine calculates the solar data values that need to be evaluated only once per day.

Subprograms Calling This Subroutine

SUBROUTINE DAYCLC

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/DATE/	IDOY	None placed
/LOCALD/	TSTALA	None placed
/SUND/	None obtained	TDECLN, EQTIME, SOLCON, ATMEXT, SKYDFF, GUNDOG

Declarations

None

Input

Name	Description
IDOY	Day of the year (Julian date, 1-366)
TSTALA	Tangent of the latitude angle (dimensionless)

Output

Name	Description
TDECLN	Tangent of the solar declination for the day of the year (dimensionless)
EQTIME	Solar equation of time for the day of the year (hrs)
SOLCON	Calculated direct normal, extraterrestrial solar radiation (apparent solar constant)(Btu/hr-ft^2)
ATMEXT	Atmospheric extinction coefficient (air mass^{-1})
SKYDFF	Sky diffuse factor (dimensionless)
GUNDOG	The hour angle of sunrise and sunset for the day (solar altitude is zero)

Table III.10 lists the first five variables defined above for the 21st day of the month. These data are taken from ASHRAE algorithm SUN1 (Ref. 1).

TABLE III.10

FIRST FIVE VARIABLES FOR THE 21ST DAY OF THE MONTH

Date	Declination (degrees)	EQTIME (h)	SOLCON $(Btu/h\text{-}ft^2)$	ATMEXT (Air Mass^{-1})	SKYDFF
1/21	-20.0	-0.190	390	0.142	0.058
2/21	-10.8	-0.230	385	0.144	0.060
3/21	0.0	-0.123	376	0.156	0.071
4/21	11.6	0.020	360	0.180	0.097
5/21	20.0	0.060	350	0.196	0.121
6/21	23.45	-0.025	345	0.205	0.134
7/21	20.6	-0.103	344	0.207	0.136
8/21	12.3	-0.051	351	0.201	0.122
9/21	0.0	0.113	365	0.177	0.092
10/21	-10.5	0.255	378	0.160	0.073
11/21	-19.8	0.235	387	0.149	0.063
12/21	-23.45	0.033	391	0.142	0.057

Note: A value for the variable GUNDOG is not provided because it is a function of latitude and declination, not simply the date.

Calculation Procedure

1. To avoid interpolating from Table III.10, the values are expressed in Fourier series form and are calculated as a function of the day of the year from the following truncated series.

$$\begin{array}{ll} \text{TDECLN} & \\ \text{EQTIME} & \\ \text{SOLCON} & = A_0 + A_1 * \cos(\omega{*}d) + A_2 * \cos(2{*}\omega{*}d) \\ \text{ATMEXT} & \\ \text{SKYDFF} & + A_3 * \cos(3{*}\omega{*}d) + B_1 * \sin(\omega{*}d) \\ & + B_2 * \sin(2{*}\omega{*}d) + B_3 * \sin(3{*}\omega{*}d), \end{array} \qquad (1)$$

where

$\omega = 2\Pi/366 = 0.01721$, and
$d = \text{IDOY}$.

2. Defining C1 and S1 as

$C1 = \cos(\omega{*}d)$,

S1 = sin (ω*d),

and by trigonometric identity,

C2 = cos (2*ω*d) = C1 * C1 - S1 * S1,
S2 = sin (2*ω*d) = 2 * S1 * C1,
C3 = cos (3*ω*d) = C1 * C2 - S1 * S2, and
S3 = sin (3*ω*d) = C1 * S2 - S1 * C2.

Substituting these into Eq. (1) yields

$$\begin{array}{l} \text{TDECLN} \\ \text{EQTIME} \\ \text{SOLCON} = A_0 + A_1 * C_1 + A_2 * C^2 + A_3 * C3 \\ \text{ATMEXT} \\ \text{SKYDFF} \quad + B_1 * S1 + B_2 * S2 + B_3 * B3. \end{array}$$

The required Fourier series coefficients are given in Table III.11.

3. GUNDOG = ACOS (-TSTALA*TDCLN).

This is obtained from the general equation

sin (β) = sin (δ) * sin (L) + cos (δ)* cos (L) * cos (h)
= RAYCOS(3) (see WDTSUN subroutines),

where
β = solar altitude (radians)
δ = declination (radians)
L = latitude (radians, and
h = hour angle (radians)

GUNDOG is obtained by setting β = 0 and solving for h.

TABLE III.11

FOURIER COEFFICIENTS

	A_0	A_1	A_2	A_3	B_1	B_2	B_3
TDECLN	-0.00527	-0.4001	-0.003996	-0.00424	0.0672	0.0	0.0
EQTIME	0.0000696	0.00706	-0.533	-0.00157	-0.122	-0.156	-0.00556
SOLCON	368.44	24.52	-1.14	-1.09	0.58	-0.18	0.28
ATMEXT	0.1717	-0.0344	0.0032	0.0024	-0.0043	0.0	-0.0008
SKYDFF	0.905	-0.0410	0.0073	0.0015	-0.0034	0.0004	-0.0006

A description of and the relationships between these angles are given below and in Figs. III.5 and III.6.

δ = declination angle, the angle between the sun-earth centerline and the plane of the equator.

On June 21 (the summer solstice), the maximum declination, δ is equal to latitude of the tropic of Cancer. On December 21 (the winter solstice), δ is equal to latitude of tropic of Capricorn.

ASHRAE Verification

Table III.12 lists the important variables computed in SUN1 and their counterparts in ASHRAE (Ref. 1) and NECAP (Ref. 3).

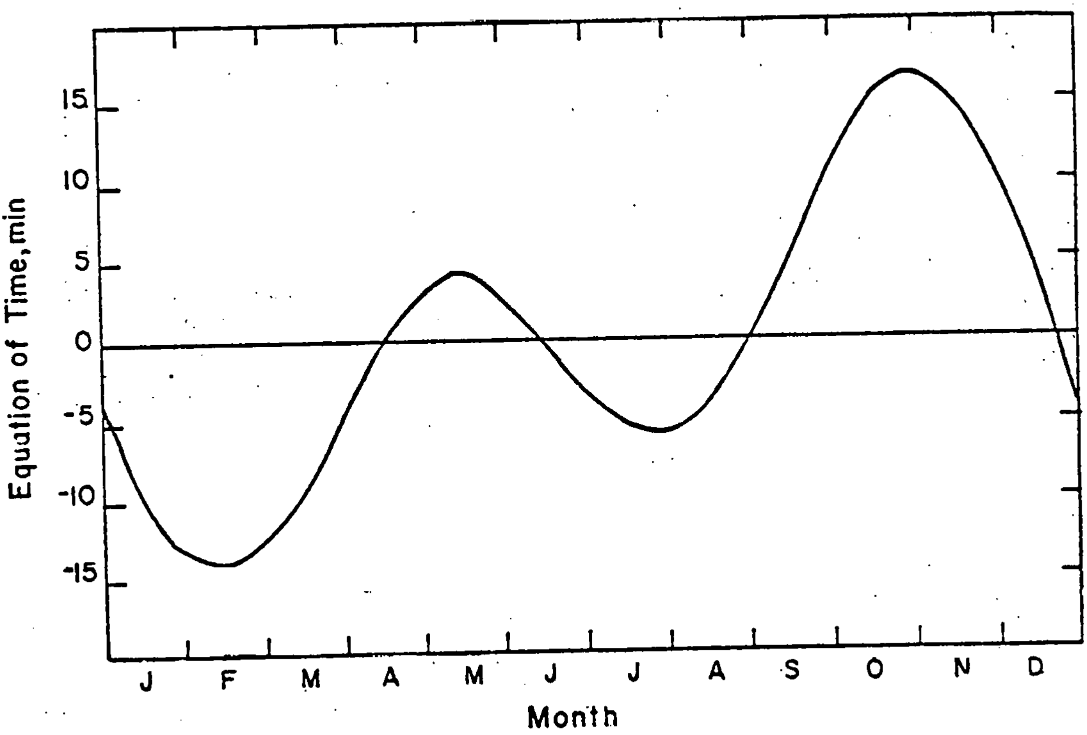


Fig. III.5. Solar equation of time.

EQTIME - Solar equation of time, takes into account the various perturbations in the earth's orbit and rate of rotation that affect the time the sun appears to cross the observer's meridian. (See Ref. 12.)

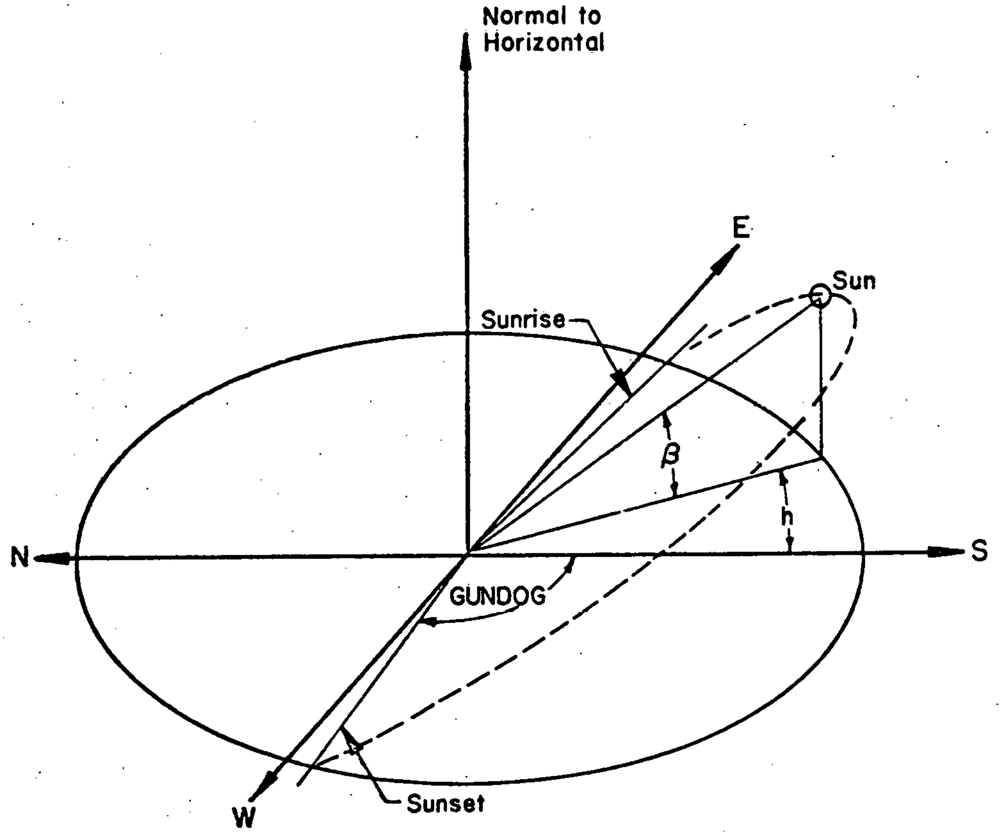


Fig. III.6. Diurnal solar geometry.

h = hour angle (radians) (0 at solar noon, projection of sun is true south).
δ = solar altitude angle (radians) (0 at sunrise and sunset).
GUNDOG = sunrise and sunset hour angle (β = 0) (radians).

TABLE III.12

VARIABLES IN SUN1, ASHRAE, AND NECAP

Cal-ERDA Name	NECAP Name	NECAP Location	ASHRAE Name	ASHRAE Location	Equivalent
TDECLN	DEABC(1)	SUN1	Tan δ	SUN	Yes
EQTIME	DEABC(2)	SUN1	ET	SUN	Yes
SOLCON	DEABC(3)	SUN1	A	SUN	Yes
ATMEXT	DEABC(4)	SUN1	B	SUN	Yes
SKYDFF	DEABC(5)	SUN1	C	SUN	Yes
GUNDOG	SUNRAS	SUN1	h'	SUN	Yes

The variables TDCLN, EQTIME, SOLCON, ATMEXT, and SKYDFF are listed in tabular form in the ASHRAE SUN subroutines for the 21st day of each month as shown in Table III.12. However, the Cal-ERDA and NECAP codes employ a set of Fourier series coefficients to curve fit these tabular data for computer use. Thus, these five variables are equivalent to the ASHRAE procedure. The variable GUNDOG is identical with SUNRAS in NECAP and h' in ASHRAE.

25. Subroutine SUN3.

Description

The SUN3 subroutine performs hourly calculations for solar data that are dependent on the orientation of a surface. It combines the calculations performed in SUN1 and WDTSUN with its own to produce the values for solar radiation incident on a given surface.

Subprograms Calling This Subroutine

SUBROUTINE CALEXT

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/PNTRS/	MX{WT, AA(MX+8), GAMMA, WA, SWA, CWA}	None placed
/SUND/	RAYCOS(1), RAYCOS(2), RAYCOS(3)	BG, ETA, RDIR, RDIF, RTOT
/WEATHD/	BSCC, RNDCC	None placed

Declarations

```
COMMON AA(1000)
INTEGER IA(1000)
EQUIVALENCE (IA(1), AA(1))
```

Input

Name	Description
WT	Surface tilt angle from horizontal (radians)
WA	Surface azimuth angle measured clockwise from Y-axis of building (radians) (See WDTSUN subroutine for explanation)
GAMMA	The cosine of the surface tilt angle, WT
SWT	The sine of the surface tilt angle, WT
CWA	The cosine of the surface azimuth angle, WA
SWA	The sine of the surface azimuth angle, WA
AA(MX+8)	The ground reflectivity (dimensionless)

RDNCC The intensity of the direct normal solar radiation corrected for the cloud cover (Btu/hr-ft^2)

BSCC Brightness of the sky corrected for the cloud cover (diffuse radiation) (Btu/hr-ft^2)

Output

Name Description

BG The solar radiation reflected from the ground (brightness of ground) (Btu/hr-ft^2)

ETA The cosine of the angle of incidence between the rays of the sun and the outward normal of the surface

RDIR Intensity of the direct normal solar radiation on a given surface (Btu/hr-ft^2)

RDIF Intensity of the diffuse component of solar radiation on a given surface (Btu/hr-ft^2)

RTOT Intensity of the total solar radiation incident on a surface (Btu/hr-ft^2)

Calculation Procedure

1. Obtain value for tilt angle WT from Common Blocks, WT = AA(MX+16).

2. Calculate the brightness of ground (reflected component).

   BG = AA(MX+8) * (BSCC + RDNCC * RAYCOS(3)), (1)

   where

   RAYCOS(3) = Cosine of angle of incidence for any horizontal surface.

3. Calculate cosine of angle of incidence, ETA, for surface. Check for certain surface orientations

   a. If surface is horizontal (WT = 0°), then set

      ETA = RAYCOS(3). (2)
      GAMMA = 1.

   b. If surface is vertical (WT = 90°), then set

      GAMMA = 0.
      SWT = 1.

   c. If surface is neither horizontal or vertical, obtain values from common block.

      GAMMA = AA(MX+18)
      SWT = AA(MX+17)

   d. Obtain value for surface azimuth angle WA from common block.

      WA = AA(MX+15)

   e. If surface azimuth angle WA is 0° (with respect to Y-axis of building), set

      SWA = 0.
      CWA = 1.

f. If WA is 90°, set

SWA = 1.
CWA = 0.

g. If WA is 180°, set

SWA = 0.
CWA = 1.

h. If WA is 270°, set

SWA = -1.
CWA = 0.

i. If the surface azimuth angle WA is not one of the four cardinal angles, obtain values from common block.

SWA = AA(MX+13)
CWA = AA(MX+14)

j. Compute ETA.

ETA = (RAYCOS(1) * SWA + RAYCOS(2) * CWA) * SWT + RAYCOS(3) * GAMMA (3)

4. Calculate the direct normal component of solar radiation, RDIR, incident on the surface.

a. If ETA $\leq$ 0 (direct beam not striking surface)

RDIR = 0.

b. If ETA < 0,

RDIR = RDNCC * ETA. (4)

5. Calculate the diffuse component of solar radiation, RDIF, incident on the surface.

a. If WT $\leq$ 45°,

RDIF = BSCC (brightness of sky). (5)

b. If WT > 35° (ground orientation)

RDIF = BG (brightness of ground). (6)

c. If 45° < WY $\leq$ 135°,

RDIF = Y * BSCC + 0.5 * BG, (7)

where

If ETA < -11.5° (direct beam strikes rear of surface),
Y = 0.45;

If ETA $\geq$ -11.5°,
Y = 0.55 + 0.437 * ETA + 0.313 * ETA * ETA (Ref. 13). (8)

6. Calculate total solar radiation striking surface,

RTOT = RDIR + RDIF.

ASHRAE Verification

Listed below are the important variable names employed by the Cal-ERDA Code and their counterparts, and locations in NECAP (Ref. 3) and ASHRAE (Ref. 1).

Cal-ERDA Name	NECAP Name	NECAP Location	ASHRAE Name	ASHRAE Location	Equivalent
BG	BG	SUN3	BG	SUN	Yes/No
ETA	ETA	SUN3	$\cos(\eta)$	SOLAD	Yes/No
RDIR	RDIR	SUN3	IdC	SOLAD	No
RDIF	RDIF	SUN3	IdHC	SOLAD	Yes/No
RTOT	RTOT	SUN3	ITC	SOLAD	Yes

BG - The equation to calculate the value of BG was given previously as (1). This equation is identical in form to that of ASHRAE; however, as explained in WDTSUN, the values of BSCC and RDNCC are not ASHRAE-derived. Also, in ASHRAE, the values of these two variables are cloudless day values only; whereas, in Cal-ERDA, they have been corrected (usually reduced) for cloud cover using the CCM subroutine.

ETA - The Cal-ERDA equation for this variable is given by Eq. (3) in the calculation procedure. The ASHRAE equation for this is given by

$$\cos(\eta) = \alpha * \cos(Z) + \beta * \cos(w) + \gamma * \cos(s).$$

Comparing the terms of the equations shows

a. ETA = $\cos(\eta)$.
b. GAMMA = α.
c. CWA * SWT = γ.
d. SWA * SWT = β.
e. RAYCOS(1) and RAYCOS(2) = cos(w) and cos(s) without building orientation information.

It can be seen that ETA is basically equivalent to $\cos(\eta)$ of ASHRAE. The exception is that RAYCOS(1) and RAYCOS(2) of Cal-ERDA are a combination of cos(w) and cos(s) of ASHRAE and are written in terms of an arbitrary building orientation.

RDIR - The Cal-ERDA equation used to calculate the direct beam radiation on a surface is given by Eq. (4) in the calculation procedure. It is

RDIR = RDNCC * ETA,

where

ETA = cosine of angle of incidence,
RDNCC = cloudy sky direct normal radiation
= RDN * CLDCOV (see WDTSUN), and
CLDCOV = cloud cover coefficient (see CCM).

The equation employed by ASHRAE is:

IdC = Id * IdHC/IdH,

where

Id = cloudless sky direct radiation on a surface
= IdN * $\cos(\eta)$ (IdN = direct normal radiation).

IdHC = cloudy sky direct radiation on a horizontal surface
= ITH * K * (1 - CC/10).

ITH = cloudless-sky total radiation on a horizontal surface.

K = empirical coefficient for solar position and sky conditions.

CC = cloud-cover coefficient calculated by ASHRAE CCF subroutine.

IdH = cloudless-sky direct radiation on a horizontal surface
 = IdN * cos(Z).

A comparison of the two equations shows that the methodology employed is not the same. The ASHRAE algorithm multiplies the cloudless-sky direct radiation on a surface by the ratio of cloudy sky to cloudless sky direct radiation on a horizontal surface. This assumes that the ratio is the same for a tilted surface.

The Cal-ERDA version simply multiplies the cloudy-sky direct normal radiation by the cosine of the angle of incidence. Note, however, that the Cal-ERDA version employs two ASHRAE equivalent variables, RDN and ETA, with a cloud-cover coefficient, CLDCOV, in a more efficient way than ASHRAE, which requires two more variables.

RDIF - The Cal-ERDA procedure for determining the diffuse radiation on a surface is equivalent to the ASHRAE methods for certain sections of the algorithm, but not for others. This can be seen by a direct comparison of the equations.

The Cal-ERDA method is given by Eqs. (5), (6), (7), and (8) in the calculation procedure, where

RDIF = cloudy-sky diffuse radiation,
 = BSCC (for WT $< 45°$)
 = BG (for WT $> 135°$)
 = BSCC * Y + 0.5 * BG (for $45° \leq$ WT $\leq 135°$),

and where

$Y = 0.55 + 0.437 * ETA + 0.313 * ETA^2$
 or if ETA $\leq -11.5°$, Y = 0.45.

The ASHRAE equations are:

IdC = diffuse radiation on a surface under consideration
 = Id * IdHC/IdH,

where

Id = cloudless sky diffuse radiation
 = BS (for WT = 0°)
 = BS * Y + 0.5 * BG (for WT = 90°),

and where

$Y = 0.55 + 0.437 * \cos(\eta) + 0.313 * \cos^2(\eta)$
 or if $\cos(\eta) \leq -11.5°$, Y = 0.45 ($\cos(\eta)$ = ETA),

IdHC = cloudy sky diffuse radiation on a horizontal surface
 = ITH * (CCF - K * (1 - CC/10)),

CCF = cloud cover factor (not the same as the cloud cover CC), and

IdH = cloudless sky diffuse radiation on a horizontal surface
 = BS.

It can be seen that the difference between the ASHRAE procedure and that of Cal-ERDA arises in the basic equations and the method in which they are used. The ASHRAE procedure does not provide separate calculations for surface tilt angles (WT) other than horizontal (0°) or vertical (90°). A footnote in the ASHRAE documentation (Ref. 1, p. 31) states that ASHRAE has no procedure for these surfaces. The Cal-ERDA method, however, accounts for any surface angle given by the user. The Cal-ERDA procedure is essentially equivalent to ASHRAE for vertical and horizontal surfaces (WT = 90°, 0°).

For any surface, ASHRAE uses the equation for IdC that employs the ratio of cloudy-sky diffuse (IdHC) to cloudless-sky diffuse radiation (IdH) on a horizontal surface multiplied by the cloudless-sky diffuse radiation for either a horizontal or vertical surface (Id). The Cal-ERDA method does not employ this ratio of IdHC/IdH, but instead uses an equivalent to the ASHRAE variable Id and multiplies it by the cloud-cover coefficient. The Cal-ERDA method also considers those surfaces that are oriented towards the ground (WT > 135°), which ASHRAE does not address.

RTOT - This procedure is the same in both ASHRAE and Cal-ERDA. However, the inputs that are summed to product RTOT (RDIR and RDIF) are not completely equivalent.

26. Subroutine TIMPRP.

SUBROUTINE TIMPRP (IN, OUT)

Description

This subroutine prepares time data for report printing.

Subprograms Calling This Subroutine

SUBROUTINE RPT1
SUBROUTINE RPT2

Subprograms Called by This Subroutine

FUNCTION SHIFT
FUNCTION ENCODE

Common Blocks

None

Declarations

```
DIMENSION IOUT (2), MO (12)
DATA MO/3HJAN, 3HFEB, 3HMAR, 3HAPR,3HMAY, 3HJUN, 3HJUL, 3HAUG,
3HSEP, 3HOCT, 3HNOV, 3HDEC/
```

Input

Name	Description
IN	Packed binary word containing month, day, and hour data

Output

Name	Description
IH	The hour of the day (1-12)
ID	The day of the month (1-31)
IM	The month of the year (JAN-DEC)
IOGLE	Label for one of the four quarters of the day (AM, PM, NOON, MDNT)

Calculation Procedure

1. Set up input array for titled month of the year.

```
DATA MO/3HJAN, 3HFEB, 3HMAR, 3HAPR, 3HMAY, 3HJUN, 3HJUL, 3HAUG, 3HSEP,
3HOCT, 3HNOV, 3HDEC/
```

2. Check to see if time data is available and, if not, return.

```
IF (IN .EQ. 0) GO TO 900
```

3. Set the input IN equal to I and pick off the required six binary bits for each output using the SHIFT function.

```
I = IN
IH = I .AND. 77B
ID = SHIFT (I, -6) .AND. 77B
IM = SHIFT (I, -12) .AND. 77B
```

4. Determine the correct time title for the hour of the day IH.

```
IF (IH - 12) 210, 220, 230
210 IOGLE = 4HAM
GO TO 250
220 IOGLE = 4HNOON
GO TO 250
230 IOGLE = 4HPM
IH = IH - 12
IF (IH .EQ. 12) IOGLE = 4HMDNT
250 CONTINUE
```

5. Set up the output IOUT in character format using the ENCODE function.

```
ENCODE (16, 1, IOUT) MO(IM), ID, IH, IOGLE 1 FORMAT (A3, I3, 1XA4, 2X)
900 CONTINUE
RETURN
```

Note that the functions SHIFT and ENCODE are specific to Control Data Corporation (CDC) equipment.

ASHRAE Verification

There is no equivalent algorithm in the ASHRAE documentation (Ref. 1).

27. Function TOTL.

FUNCTION TOTL (X,N)

Description

The function TOTL is a simple summing algorithm that adds the values of the array X(N) from X(1) to X(N).

Subprograms Calling This Function

SUBROUTINE RPT2S

Subprograms Called by This Function

None

Common Blocks

None

Declarations

DIMENSION X(N)

Input

Name	Description
X	An array of values to be summed (dimensionless)
N	The number of values in array X to be summed (dimensionless)

Output

Name	Description
TOTL	The sum of the values contained in the array X from X(1) to X(N) (dimensionless)

Calculation Procedure

1. Dimension the array X.

   DIMENSION X(N)

2. Initialize the sum.

   T = 0

3. Sum the values of the array.

   T = T + X(I)

4. Set the output.

   TOTL = T

ASHRAE Verification

Not applicable.

28. Subroutine TRANSL.

SUBROUTINE TRANSL (XO, YO, ZO, XYZ, N)

Description

The subroutine TRANSL translates the four vertices of a rectangle from the array XYZ by the values XO, YO, and ZO. It is called by the subroutine GEOPR1 for attachments to walls (IFLG = 1).

Subprograms Calling This Subroutine

SUBROUTINE GEOPR1

Subprograms Called by This Subroutine

None

Common Blocks

None

Declarations

```
DIMENSION XYZ (3,N)
```

Input

Name — Description

XO — The distance in the X-direction from the origin of an imaginary plane the rectangle must be translated (ft)

YO — The distance in the Y-direction from the origin of an imaginary plane the rectangle must be translated (ft)

ZO — The distance in the Z-direction from the origin of an imaginary plane the rectangle must be translated (ft)

XYZ — Vertex coordinates of the rectangle to be translated (ft)

N — The number of rectangle vertices to be translated (dimensionless)

Output

XYZ — An array of vertex coordinates translated from the origin of an imaginary plane by the values of XO, YO and ZO

Calculation Procedure

1. Dimension the array XYZ.

```
DIMENSION XYZ (3,N)
```

2. Using a DO loop, translate the required number of vertices by their respective values of XO, YO, and ZO and return.

```
DO 400 I = 1, N
XYZ (1,I) = XYZ (1,I) + XO
XYZ (2,I) = XYZ (2,I) + YO
XYZ (3,I) = XYZ (3,I) + ZO
400 CONTINUE
RETURN
```

ASHRAE Verification

There is no equivalent ASHRAE algorithm for this subroutine.

29. Subroutine UGPRP.

SUBROUTINE UGPRP (ZUGFUA, ZUGWUA)

Description

The LOADS subroutine UGPRP processes the data for underground floors and walls (UGFW).

Subprograms Calling This Subroutine

SUBROUTINE READSF

Subprograms Called by This Subroutine

SUBROUTINE GEOPR1

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/PNTRS/	MZ, MP, MU, MULEN	MU, MG, MP
/PTRBGN/	IBUG, IBGE, IBPR	None placed
/SURFD/	None obtained	AREA

Declarations

```
COMMON AA(1000)
INTEGER IA(1000)
EQUIVALENCE (IA(1), AA(1))
```

Input

Name	Description
ZUGFUA	Value of heat transfer function (UA) for underground floors. Input as zero for initialization (see output)
ZUGWUA	Value for heat transfer function (UA) for underground walls. Input as zero for initialization (see output)
IA(MZ+42)	Zone underground surface pointer
IBUG	Underground floors and walls (UGFW) data block beginning pointer
IA(MU)	Number of space underground surfaces
IA(MU+4)	UGFW geometry pointer
IBGE	Geometry data block beginning pointer
IA(MU+5)	UGFW properties pointer
IBPR	Properties data block beginning pointer
AA(MU+4)	UGFW surface area (ft^2) (temporary)
IA(MU+3)	UGFW type; 0 = underground floors, 1 = underground wall, 2 = floor
AA(MU+6)	UGFW multiplier
AA(MU+8)	Surface heat-transfer factor (U) (Btu/hr-ft^2)
MULEN	UGFW data entry length

Output

Name	Description
ZUGFUA	Value for heat-transfer function (UA) for underground floors (Btu/hr)
ZUGWUA	Value for heat transfer function (UA) for underground walls (Btu/hr)

Calculation Procedure

1. Locate the starting point for UGFW data block.

```
MU = IA(MZ+42) + IBUG
```

2. Set the number of UGFW surfaces.

```
NSUG = IA(MU)
```

3. Calculate the required data for each surface using a DO loop.

```
DO 1000 IU = 1, NSUG
```

4. Locate starting point for geometry and properties data blocks.

```
MG = IA(MU+4) + IBGE
MP = IA(MU+5) + IBPR
```

5. Call the subroutine GEOPR1 (0) to prepare geometry data. Argument of zero indicates data is for wall.

```
CALL GEOPR1 (0)
```

6. Determine the area of the surface being considered.

```
AREA = AA(MU+4)
```

7. Determine the UGFW type and proceed to the appropriate calculations; 0 = underground floor, 1 = underground wall, 2 = floor.

```
IF (IA(MU+3) -2) 300, 400, 500
300 ZUGFUA = ZUGFUA + AREA * AA(MU+6) * AA(MP+8)
GO TO 900
400 ZUGWUA = ZUGWUA + AREA * AA(MU+6) * AA(MP+8)
GO TO 900
500 CONTINUE
RETURN
END
```

Calculation Procedure

There is no equivalent ASHRAE algorithm for Cal-ERDA subroutine UGPRP.

30. Subroutine WDREAD.

Description

This subroutine reads the Cal-ERDA weather file for a given day and returns the current hourly data.

Subprograms Calling This Subroutine

SUBROUTINE WDTSUN
SUBROUTINE INITLZ

Subprograms Called by This Subroutine

SYSTEM ROUTINE REMARK
SYSTEM ROUTINE ABORT

Common Blocks	Variables Obtained From Common Blocks	Variables Placed in Common Blocks
/DATE/	IDOY, IHR	None placed
/FILES/	IWEATH	None placed
/WEATHD/	None obtained	TGNDR, CLRNES, DBT, WBT, DPT, PATM, WNDSPD, CLDAMT, ICLDTY, IWNDDR, DBTR

Declarations

```
DIMENSION IW(192), WW(192)
EQUIVALENCE (IW(1), WW(1))
```

Input

Name	Description
IDOY	The day of the year, 1-366 (dimensionless)
IHR	The hour of the day, 1-24 (dimensionless)

Output

Name	Description
IWDID(1)	
LDOY	The day of the year currently in memory, 1-366 (dimensionless)
TGNDR	Ground temperature (°R)
CLRNES	Clearness number, coefficient for the clarity of the atmosphere according to season and geographical location, 0.85 to 1.15
IW	A data array, containing the last 192 words of the file IWEATH
DBT	Dry-bulb temperature (°F)
WBT	Wet-bulb temperature (°F)
DPT	Dew-point temperature (°F)
PATM	Atmospheric pressure (in. Hg)
WNDSPD	Wind speed (knots)
CLDAMT	Amount of local cloud cover, 0-10 (dimensionless)
ICLDTY	Cloud type (dimensionless)
IWNDDR	The wind direction (radians)
DBTR	Dry-bulb temperature (°R)

Calculation Procedure

1. Compare the day currently in memory (LDOY) with the day desired (IDOY).

```
200 IF (IDOY-LDOY) 300, 600, 400
```

2. If the day of the year desired (IDOY) precedes the day in memory (LDOY, backspace to the day desired (IDOY). (LDOY > IDOY)

```
300 IDIF = LDOY - IDOY + 1
DO 320  I = 1, IDIF
BACKSPACE IWEATH
320 CONTINUE
```

3. If the day of the year desired (IDOY) follows the day currently in memory (LDOY), advance the file to the desired day (IDOY). (LDOY < IDOY)

```
400 READ (IWEATH) IWDID(1), LDOY, TGNDR, CLRNES, IW
IF (EOF (IWEATH).EQ.0) GO TO 200
CALL REMARK (26H*** WEATHER TAPE ERROR***)
CALL ABORT
```

This section of code also places the contents of the weather file as shown in statement No. 400 and checks for an end of file (EOF). If the file is not complete, an error message is printed and the program is aborted.

```
600 I = IHR * 8-8
DBT = WW(I+1)
WBT = WW(I+2)
DPT = WW(I+3)
PATM = WW(I+4)
WNDSPD = WW(I+5)
CLDAMT = WW(I+6)
ICLDTY = WW(I+7)
IWNDDR = IW(I+8)
DBTR = DBT + 460
```

ASHRAE Verification

This is a data reading subroutine and does not have an equivalent in the ASHRAE reference (Ref. 1).

31. Subroutine WDTSUN

Description

The WDTSUN subroutine performs the hourly calculations for solar position and intensity.

Subprograms Calling This Subroutine

SUBROUTINE DAYCLC

Subprograms Called by This Subroutine

SUBROUTINE WDREAD
SUBROUTINE CCM

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/DATE/	IHR	None placed
/LOCALD/	ITIMZ, STALON, CSTALA, SSTALA, SBAZIM, CBAZIM	None placed
/SUND/	GUNDOG, TDECLN, EQTIME, SOLCON, ATMEXT, SKYDFF	ISUNUP, HORANG, RND, RTOT, RDIR, BSUN, BG, RAYCOS(1), RAYCOS(2), RAYCOS(3)
/WEATHD/	CLRNES, CLDAMT, CLDCOV	RDNCC, BSCC, SKYA

Declarations

None

Input

Name	Description
CLDAMT	The total amount of cloud cover in the locale during the hour, 0-10 (dimensionless)
IHR	The hour of the day (1-24 hrs)

ITIMZ	The time zone index number
	Atlantic = 4
	Eastern = 5
	Central = 6
	Mountain = 7
	Pacific = 8

EQTIME	Daily value of the solar equation of time (hrs). This is detailed in the description of SUN1.
STALON	The station longitude (radians)
GUNDOG	The hour angle at sunrise and sunset (symmetric about solar noon) (radians). This is also detailed in SUN1.
TDECLN	The tangent of the daily solar declination angle, dimensionless. The declination angle is explained in detail in SUN1.
CSTALA	The cosine of the station latitude (dimensionless)
SSTALA	The sine of the station latitude (dimensionless)
SBAZIM	The sine of the building azimuth angle (dimensionless). For an explanation of building azimuth angle see Fig. III.7 in Calculation Procedure
CBAZIM	The cosine of the building azimuth angle (dimensionless)
SOLCON	Calculated direct normal, extraterrestrial solar radiation (apparent solar constant) (Btu/hr-ft^2), calculated in SUN1

CLRNES Clearness number, coefficient for the clarity of the atmosphere according to season and geographical location, 0.85 to 1.15, dimensionless. See Ref. 3.

ATMEXT Atmosphere extinction coefficient, air mass^{-1}; calculated in SUN1

SKYDFF Sky diffuse factor (dimensionless); calculated in SUN1

CLDCOV The cloud cover modifier calculated in CCM as the local variables CC (dimensionless)

Output

Name Description

RAYCOS(1)
RAYCOS(2)
RAYCOS(3) Directional cosines of the sun (dimensionless). RAYCOS(1) and RAYCOS(2) provide the hourly azimuthal position (east to west) of the sun in relation to the building orientation. RAYCOS(3) provides the hourly altitude (vertical) position of the sun and is a function of the latitude and daily declination (dimensionless)

RDNCC Direct normal intensity of the solar radiation modified for the local cloud conditions (Btu/hr-ft^2)

BSCC The diffuse component of the solar radiation (brightness of sky) modified for the local cloud conditions (Btu/hr-ft^2)

SKYA Sky view factor based on amount of clouds and used in the subroutine CALEXT. It is calculated here to eliminate the necessity of rereading the weather file in CALEXT (dimensionless).

Calculation Procedure

1. Obtain the required weather information.

   CALL WDREAD

   The subroutine WDREAD inputs the necessary weather data from the Cal-ERDA weather file.

2. Calculate the sky view factor SKYA for use by the subroutine CALEXT.

   SKYA = (10. - CLDAMT) * 2. (1)

3. Calculate the hour angle (HORANG) of the sun.

   HORANG = 0.2618 * (FLOAT (IHR-12 + ITIMZ) + EQTIME) - STALON (2)

4. Check to see if the sun is up.

   If the absolute value of HORANG is greater than the absolute value of GUNDOG, set ISUNUP equal to zero.

   If (ABS(HORANG).GT.ABS (GUNDOG)) GO TO 1800 (3)
   where

   HORANG = present hour angle (radians),
   GUNDOG = hour angle of sunrise and sunset (radians),
   1800 = statement number of ISUNUP = 0,
   ISUNUP = 0. (sun is down),
   ISUNUP = 1. (sun is up).

5. Determine directional cosines for the hour.

   a. Combine portions of equation elements.

```
TDECL2 = TDECLN * TDECLN
SD = TDECLN * (1. - .1667 * TDECL2)
CD = 1. - .5 * TDECL2
CHCD = COS (HORANG) * CD
SHCD = SIN (HORANG) * CD
CLSD = SD * CSTALA
CHCDSC = CHCD * SSTALA - CLSD
```

   b. Use the values obtained above in conjunction with the building azimuth angle to determine the directional cosines. See Fig. III.7 for explanation of building azimuth angle.

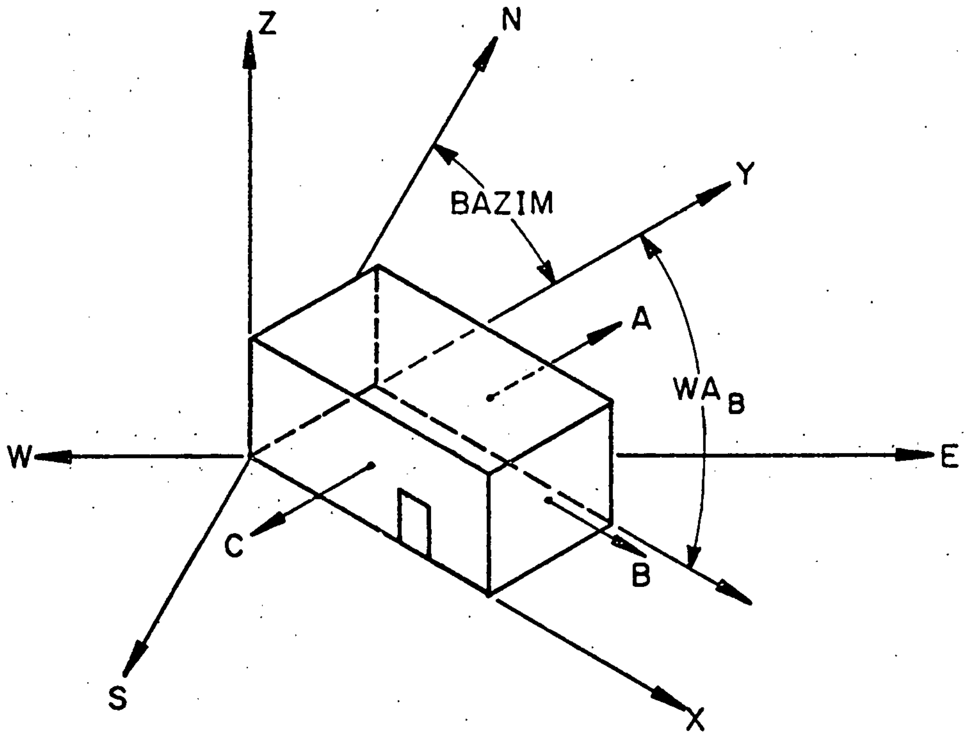


Fig. III.7 Building orientation.

BAZIM = Building azimuth angle, the difference between the Y-axis of the building and the north axis as shown in Fig. III.7.

WA = Wall azimuth angle, the difference between the Y-axis of the building and the outward normal of the wall (surface) in question.

For example, in Fig. III.7, the building azimuth angle, BAZIM, is equal to 50°.

BAZIM = 50° †

and the wall azimuth angle of wall A is 0°.

WA_A = 0°.†

The wall azimuth of walls B and C are 90° and 180°, respectively.

WA_B = 90° †

WA_C = 180° †

RAYCOS(1) = CHCDSC * SBAZIM - SHCD * CBAZIM (4)

RAYCOS(2) = CHCDSC * CBAZIM - SHCD * SBAZIM (5)

RAYCOS(3) = SSTALA * SD + CSTALA * CHCD (6)

Note that RAYCOS(3) is independent of the building orientation.

6. At this point, another check is made to verify that sunrise has occurred. If the value of RAYCOS(3) is greater than 0.001, sunrise has occurred and the radiation component calculations will continue. If not, the sunrise has not occurred and ISUNUP is set to zero.

   ```
   IF (RAYCOS(3)).GT.0.001) GO TO 1200          (7)
   RAYCOS(3) = 0.
   GO TO 1800
   ```

   where

   1200 = statement number for first radiation component calculations, and
   1800 = statement number for ISUNUP = 0.

7. Calculate the cloudless-sky direct normal beam radiation, RDN.

   RDN = SOLCON * CLRNES * EXP(-ATMEXT/RAYCOS(3)) (8)

8. Calculate the cloudless-sky diffuse radiation component (brightness of sky), BSUN, as a function of RDN.

   BSUN = RDN * SKYDFF/(CLRNES * CLRNES) (9)

9. Modify the values of RDN and BSUN (Eqs. 8 and 9) to account for the current local cloud conditions.

   a. Call the subroutine CCM that will return a cloud-cover coefficient, CLDCOV, each hour (dimensionless).

      CALL CCM (10)

   b. Modify the direct normal beam radiation component, RDN, with CLDCOV to calculate the new variable RDNCC.

      RDNCC = RDN * CLDCOV (11)

   c. Modify the diffuse radiation component, BSUN, with CLDCOV to form BSCC.

      BSCC = BSUN * CLDCOV (12)

† This value is converted to radians in the program.

ASHRAE Verification

Listed below are the important variable names employed by the Cal-ERDA code and their counterparts and location in NECAP (Ref. 1) and ASHRAE (Ref. 3).

Cal-ERDA Name	NECAP Name	NECAP Location	ASHRAE Name	ASHRAE Location	Equivalent
HORANG	H	Main Program	h	SUN	Yes
RAYCOS(1)	RAYCOS(1)	SUN 2	cos(w) & cos(s)	SUN	Yes/No
RAYCOS(2)	RAYCOS(2)	SUN 2	cos(s) & cos(w)	SUN	Yes/No
RAYCOS(3)	RAYCOS(3)	SUN 2	Cos(z)	SUN	Yes
RDN	RDN	SUN 2	IDN	SUN	Yes
BSUN	BS	SUN 2	BS	SUN	Yes
CLDCOV	see description of subroutine CCM				No
RDNCC	RDN	Main Program	IDHC	SOLAD	Yes/No
BSCC	BS	Main Program	IdHC	SOLAD	Yes/No

HORANG - The equation to calculate the current hour angle of the sun is given in Eq.(2) of the calculation procedure. It is identical to the ASHRAE equation except the units are in radians instead of degrees.

RAYCOS(1) - The directional cosines for solar position are shown in Eqs. (4),
RAYCOS(2) (5), and (6) of the calculation procedure. RAYCOS(1) and RAYCOS(2)
RAYCOS(3) in Cal-ERDA differ only slightly from the ASHRAE method. This difference arises because of the incorporation of building orientation data (CBAZIM and SBAZIM) into RAYCOS(1) and (2). RAYCOS(3), which is dependent only on the geographical location of the buildings, is exactly the same as the ASHRAE version.

RDN - This is detailed by Eq. (8) in the calculation procedure and is identical to its ASHRAE counterpart IDN.

BSUN - This variable is given by Eq. (9) and is also identical to the ASHRAE equivalent, BS.

CLDCOV - This variable is discussed in more detail in the description of the subroutine CCM. It is not equivalent to the ASHRAE version. It is mentioned here because of its effect on Items 5 and 6 above in forming the output variables RDNCC and BSCC below.

RDNCC - This is the variable name for the cloudy-sky direct normal beam radiation and is given by Eq. (11) in the calculation procedure. The ASHRAE variable that corresponds the closest is IdHC and is the cloudy-sky direct radiation on a horizontal surface. It is calculated by

IdHC = ITH * K * (1 - CC/10.),

where

ITH = Total cloudless-sky radiation on a horizontal surface (ITH = RDN + BSUN of Cal-ERDA).

K = Empirical coefficient for solar position and sky conditions (see Ref. 1, p. 30).

CC = Cloud cover calculated by ASHRAE subroutine CCF (see the description of Cal-ERDA subroutine CCM).

It is obvious that although the Cal-ERDA algorithm begins with an ASHRAE equivalent variable, RDN, the similarity in method stops there. The ASHRAE method prefers to work with values on the horizontal and a set of coefficients and ratios. The Cal-ERDA method simplifies this by taking the cloudless-sky value for direct radiation and simply multiplying by the cloud-cover coefficient.

BSCC - This is the variable name for the cloudy sky diffuse component of the solar radiation. It is calculated in exactly the same way as RDNCC and is shown in Eq. (12) of the calculation procedure. Again, the closest ASHRAE variable is for horizontal surface and is called IdHC. This is calculated by the equation

IdHC = ITH * (CCF - K * (1 - CC/10.)),

where

CCF = Cloud-cover factor calculated by ASHRAE subroutine CCF. This is a function of the cloud cover, CC.

ITH = Defined in Item 8 above.

K = Defined in Item 8 above.

CC = Defined in Item 8 above.

The difference between the Cal-ERDA variable BSCC and the ASHRAE variable IdHC is quite evident. The Cal-ERDA method provides the highest value of available diffuse radiation (brightness of sky) when the sky is cloudless and, therefore,

BSCC = BSUN.

This approach will provide different results for conditions of light-overcast skies when the diffuse value will tend to increase. This occurs because the majority of the radiation entering the outer surface of the cloud layer is transformed into diffuse radiation.

The ASHRAE method, however, considers the value of IdHC to be a function of the total radiation, ITH, and the local cloud conditions. Note that the value of diffuse radiation on a particular surface is not the same and is discussed in the description of subroutine SUN3.

32. Subroutine ZONLOC.

SUBROUTINE ZONLOC (XYZ, KOSE)

Description

This subroutine provides zone location data in global coordinates.

Subprograms Calling This Subroutine

SUBROUTINE GEOPR1

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/ZWALOC/	XZO, YZO, ZZO, CAZ, SAZ	None placed

Declarations

DIMENSION XYZ (3, KOSE)

Input

Name	Description
XYZ	Coordinates for vertices
KOSE	Number of vertices
XZO	X-coordinate of zone with respect to building coordinate system
YZO	Y-coordinate of zone with respect to building coordinate system
ZZO	Z-coordinate of zone with respect to building coordinate system
CAZ	Cos (AZO), cosine of azimuth of space
SAZ	Sin (AZO), sine of azimuth of space

Output

Name	Description
XYZ(1,I)	X-coordinate (global) for vertex I
XYZ(2,I)	Y-coordinate (global) for vertex I
XYZ(3,I)	Z-coordinate (global) for vertex I

Calculation Procedure

1. Dimension an output array for three numbers per vertex.

   DIMENSION XYZ(3,KOSE)

2. Loop through the calculations for each vertex specified.

   DO 400 I = 1, KOSE

3. Convert the coordinates of the zone vertices to global coordinates.

```
X = XYZ(1,I)
Y = XYZ(2,I)
XYZ(1,I) = XZO + X * CAZ + Y * SAZ
XYZ(2,I) = YZO + X * SAZ + Y * CAZ
XYZ(3,I) = ZZO + XYZ(3,I)
```

4. Continue loop for each vertex.

```
400 CONTINUE
RETURN
END
```

ASHRAE Verification

There is no ASHRAE equivalent algorithm for ZONLOC.

33. System Subroutine REMARK.

```
REMARK (43H    ERROR     LOADS STANDARD FILE MISSING)
REMARK (39H    ERROR     BAD LOADS STANDARD FILE)
REMARK (26H    WEATHER TAPE ERROR)
```

Description

This is an operating system routine that prints the statement contained in the calling argument. This particular routine is specific to the LBL operating system BKY. It is used in lieu of a PRINT command.

Subprograms Calling This System Routine

```
SUBROUTINE READSF
SUBROUTINE WDREAD
```

The remainder of the subroutine description format is not applicable to this type of routine.

34. System Subroutine SETFLS

```
SETFLS (L)
SETFLS (LOCF (AA (L)))
```

Description

This is an operating system routine that sets the field length to the argument contained in the call statement. This particular routine is specific to the LBL operating system BKY.

Subprograms Calling This System Routine

```
SUBROUTINE READSF
SUBROUTINE SETBAC
```

The remainder of the subroutine description format is not applicable to this type of routine.

IV. SYSTEMS PROGRAM

A. Objective and Description

The SYSTEMS Program simulates the operation and response of the equipment and systems that control the temperature and/or humidity and distribute heating and/or cooling to the spaces being conditioned. These particular systems are referred to as "secondary" or "terminal" systems because they distribute and control energy flow rather than convert energy. The program that simulates the latter category of equipment and the energy conversion system is called PLANT and is described in Chap. V. The hourly space loads calculated by the LOADS Program are not necessarily the loads that are seen by the heating and cooling "plant." Because of ventilation air requirements, equipment operating schedules, and temperature fluctuations required for control actuation, the building's hourly heating and/or cooling requirement will be different from the summation of the hourly space transmission and internal loads. The purpose of the SYSTEMS simulation program is therefore twofold:

(1) Based upon peak (or design day) heating and cooling requirements, provide information for sizing "secondary" energy distribution system components; and

(2) Simulate each distribution system as it responds to space thermal requirement and determine the requirement it is placing upon the central heating cooling plant.

The SYSTEMS program simulates heat and moisture exchange processes using ASHRAE procedures where applicable (Ref. 2). At the current stage of development, the user must select from a listing or menu of "standard" or "familiar" system types. Additional components or subsystems may be added with user modification of the SYSTEMS Program. The user can select options or alternative component and control features available for each system type. The available systems are described, along with their optional features, in Sec. C of this chapter.

B. Program Structure

As shown on the Cal-ERDA configuration chart, the SYSTEMS Program requires input from the preprocessor and the LOADS output file. A brief flow chart is shown in IV.B.2 and a description follows.

1. SYSTEMS Program Calculation Procedure. Subroutine SETUP is called to initialize and assign data block storage locations of schedule, zone,

system, and plant information. This subroutine rewinds files, sets pointers, checks for sufficient input data, and prints several data input error messages. The main program generates the System Verification Reports 1 and 2. BLOCKIO is called twice to enable load and plant binary input/output. The program then enters the hourly calculation loop by calling subroutine HOURIN.

Subroutine HOURIN sets the length of a day at 24 hrs., reads the building loads information, and calls WDREAD for the next hour of weather data to be used in the hourly calculations.

The program then enters the plant loop, sums loads to be passed to the PLANT program, and updates yearly maximums. The program can accommodate up to two plants with 10 zones each. The program enters the systems loop within the plant loop, computes a system pointer, calls the particular system subroutine, performs the simulation, and stores the resulting cooling and heating loads and flow rates. The monthly and yearly sums are updated, and a check is made for the yearly maximum loads. At the end of each hour, subroutine REPORT stores the plant output information in an output buffer.

The program increments the hour and checks for the end of the day. If the end of the day has not been reached, control is returned to HOURIN for the next hour of calculations. If the end of the day has been reached, a check is made for the end of the run.

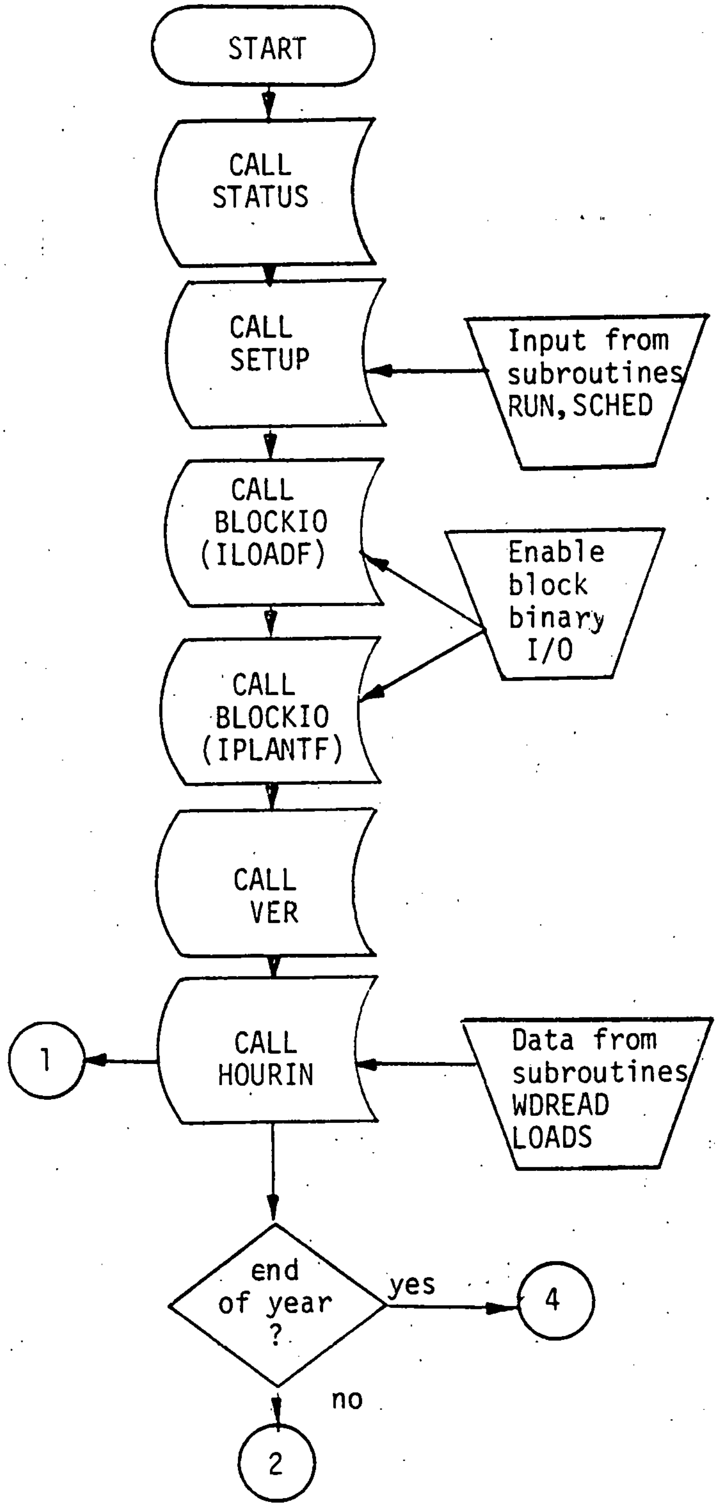
If the end of a run has been reached, subroutine REPORT stores the plant output information. The program then checks to see if further runs are to be made. If there are no further runs, the program stops. If additional runs are required, subroutine RUN stores the run period limits and resets date information. Subroutines SCHED and ZERO are called to reset schedule information and zero out old totals.

If the run is not complete, the hour of the day is reset to 1, the day number is incremented, and a check is made for the end of the month.

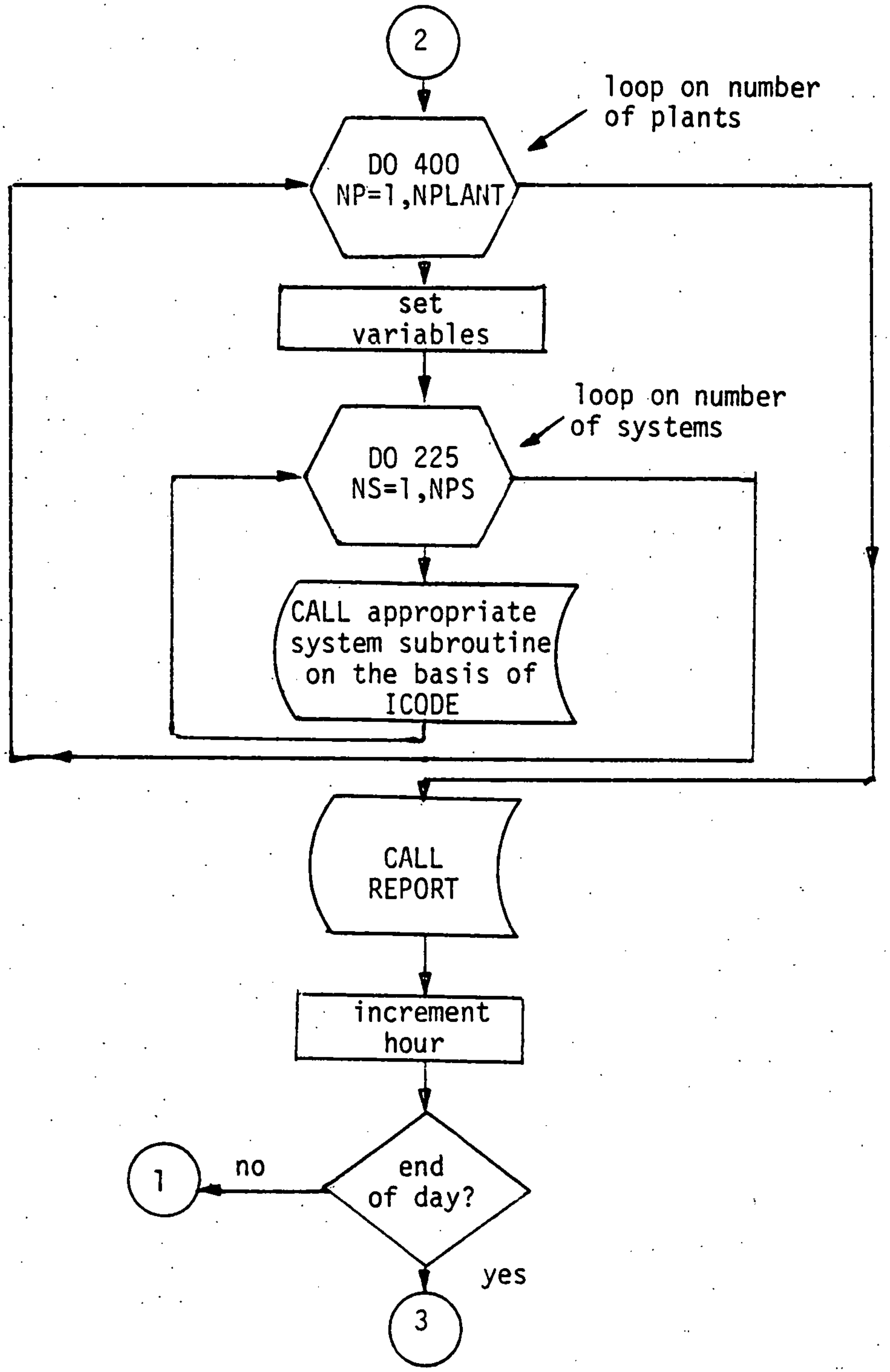
If the calculations have reached the end of the month, the day of month number is reset to 1, the month number is incremented, monthly totals are written to a record, and subroutine ZERO is called to zero out old totals. If the calculations have also reached the end of a year, the month and year numbers are reset and a check is made for a leap year.

If the calculations have not reached the end of a month, the day of the week and day of the year numbers are incremented. Subroutines SCHED and DST are called; then control returns to HOURIN for further hourly calculations.

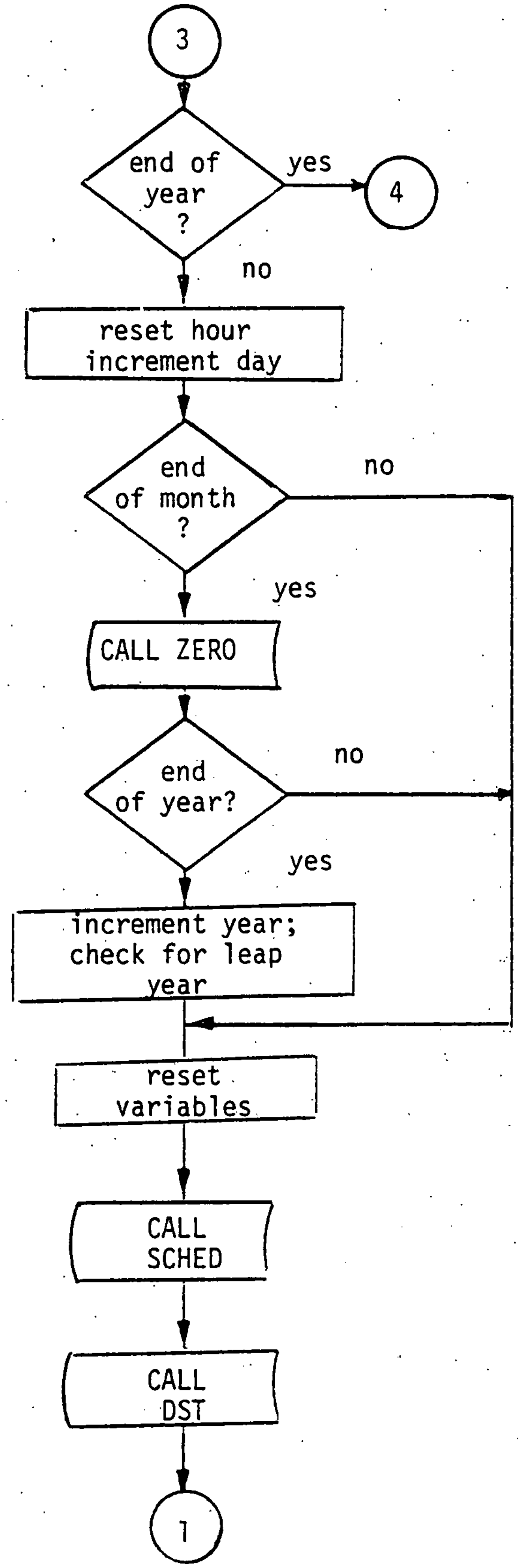
2. SYSTEMS Flow Chart. July 21, 1977



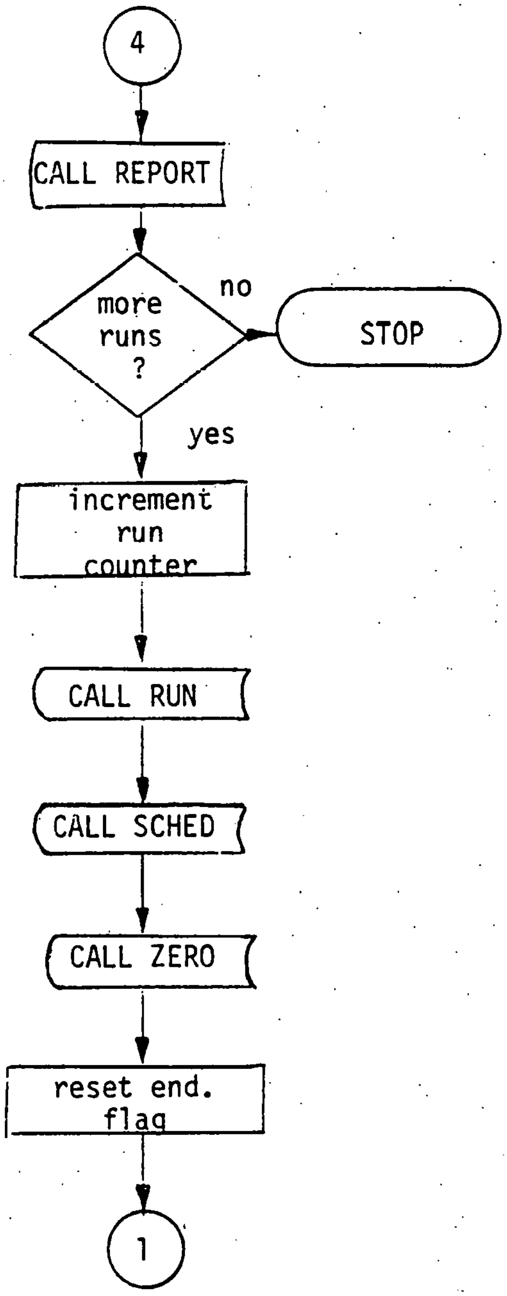
SYSTEMS Flow Chart - Continued



SYSTEMS Flow Chart - Continued



SYSTEMS Flow Chart - Continued



C. System Descriptions

The following pages contain the system schematics that appeared in the draft Cal-ERDA Users Manual. At this date, updated schematics and formulation notes have not been received for some systems. Sometimes the formulation notes provided do not match the schematics in NECAP or Cal-ERDA. Also,sometimes the code provided for a system does not match the schematics or the old formulation notes.

1. System No. 1 - Single-Zone For System

a. General Description. The single-zone fan system consists of an air handling unit, heating and cooling coils, face and bypass dampers, reheat coils, a return fan, and an economizer. The face and bypass damper algorithm has not yet been implemented. The system is designed to serve one zone that controls the supply air on cold deck dry-bulb temperature. If the system serves more than one zone, the supply air is reheated in the remaining zones. See Fig. IV.1.

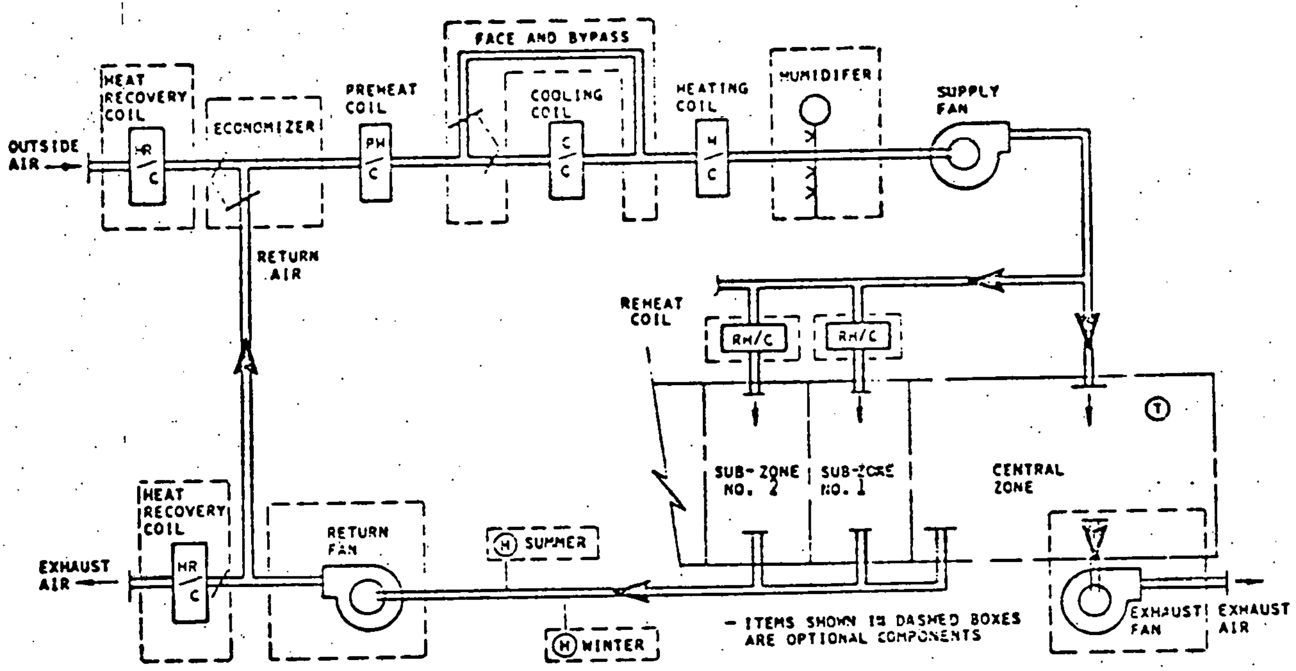


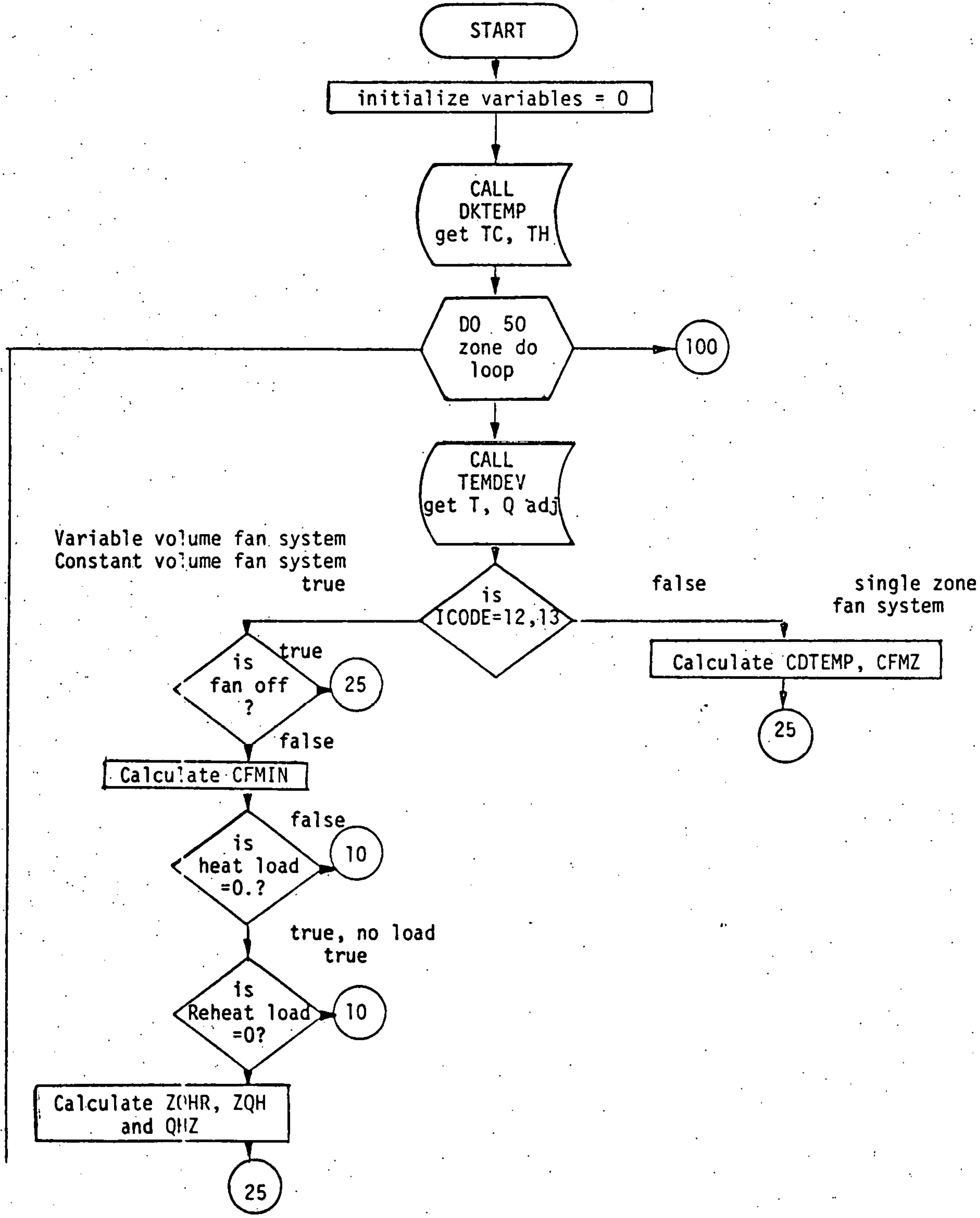
Fig. IV.1. System No. 1 - Single-zone fan system with optional sub-zone reheat.

b. VARVOL Calculational Sequence. Subroutine VARVOL is called by the SYSTEMS Program as the driver subroutine for the single-zone fan system, the variable-volume fan system with optional reheat, and the constant-volume reheat fan system. Parameters and variables used in the subroutine are initialized or set equal to zero. Subroutine DKTEMP is called to set the hot- and cold-deck temperatures.

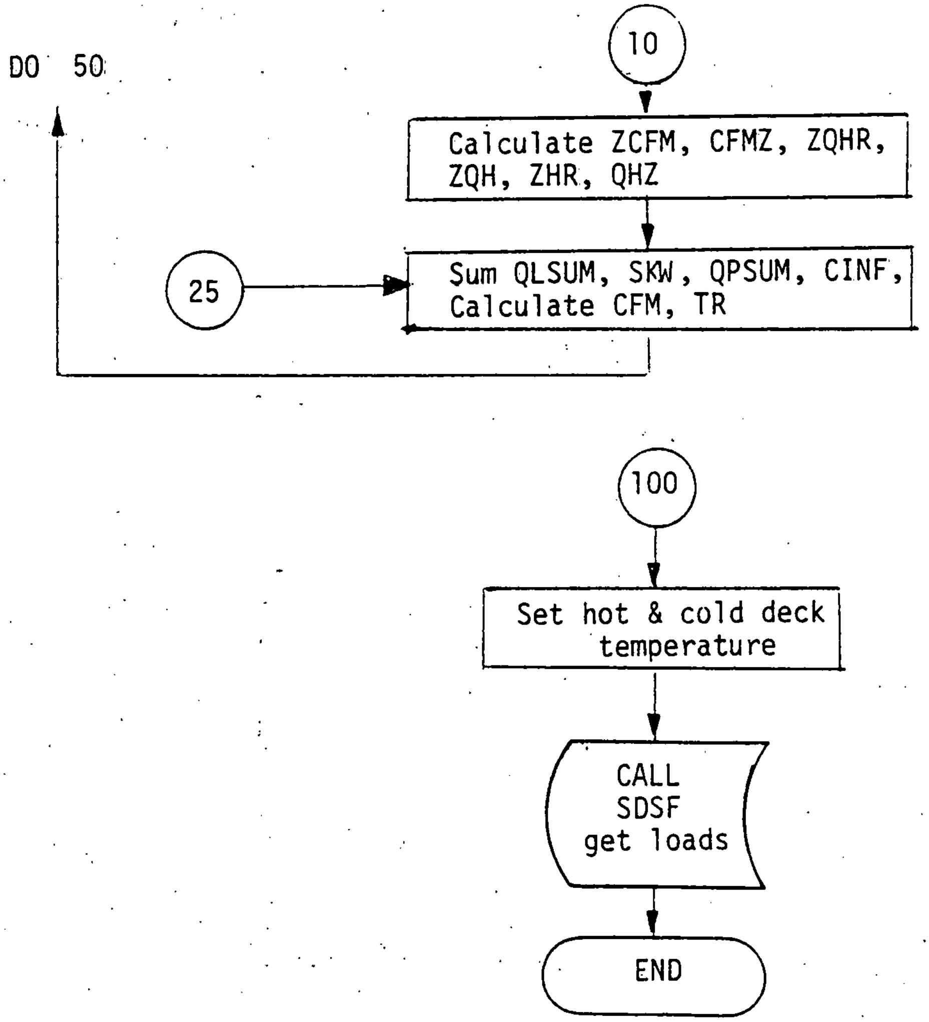
The routine enters a loop on all the zones serviced by this system. TEMDEV is called to get the adjusted load and temperature for each zone. A check is then made for system type. If system type is a single-zone fan system, the cold-deck temperature and the zone air flow are calculated, and control passes to the load summing part of the loop. If system type is either variable-volume or constant-volume reheat, a check is made for fan operation. If the fan is off, control is passed to the load-summing part of the loop. If the fan is on, the CFMIN variable is calculated. CFMIN can vary from zero to one for the variable-volume system, but must be set equal to one for the constant-volume reheat system. A check is made for sensible loads. If there is a heat load, control passes to the zone air flow and load-calculation portion of the loop. If there is a cooling load, the zone air flow is calculated, and a check is made for any necessary reheating. If reheating is required, the zone reheat load is calculated and added to the zone heat load. If no reheat is required, control passes to the load-summing part of the loop. Results of this DO loop are the calculations of zone air flows, heating and reheating loads, sensible cooling loads, and the sums of latent, plenum, electrical, and infiltration loads.

After the zone DO loop is complete, the hot- and cold-deck temperatures are reset and subroutine SDSF is called. The primary air system loads are calculated and then added to the zone loads to get total system loads.

c. VARVOL Functional Flow Chart.



VARVOL Functional Flow Chart - Continued


10
DO 50
Calculate ZCFM, CFMZ, ZQHR, ZQH, ZHR, QHZ
25
Sum QLSUM, SKW, QPSUM, CINF, Calculate CFM, TR
100
Set hot & cold deck temperature
CALL SDSF get loads
END

2. System No. 2 - Multizone Fan System

a. General Description. The multizone fan system consists of a blow-through supply fan, a return fan, heating and cooling coils in parallel, a humidifier in the hot-air duct, and an economizer. The hot- and cold-air streams are mixed at the air handling unit to control space temperature in each zone. The preheat coil and humidity control in each zone have not yet been implemented. See Fig. IV.2.

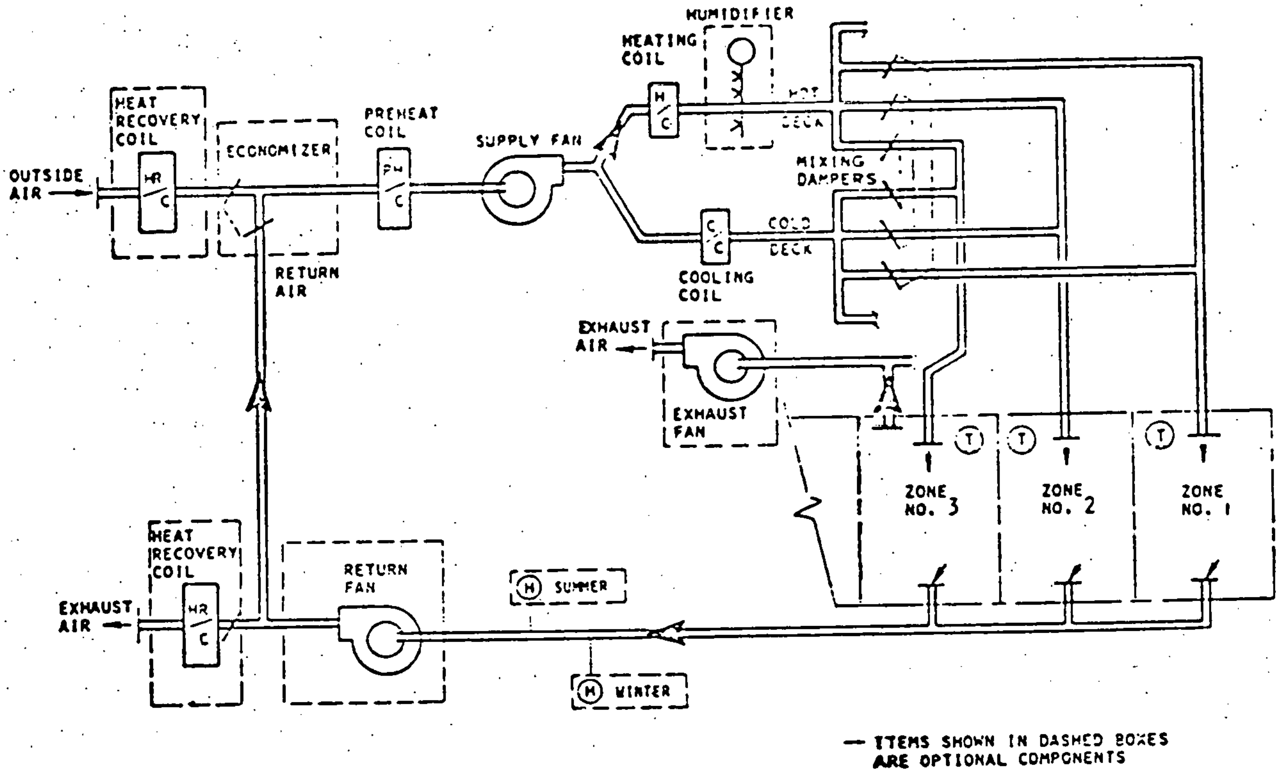


Fig. IV.2. System No. 2 - Multizone fan system.

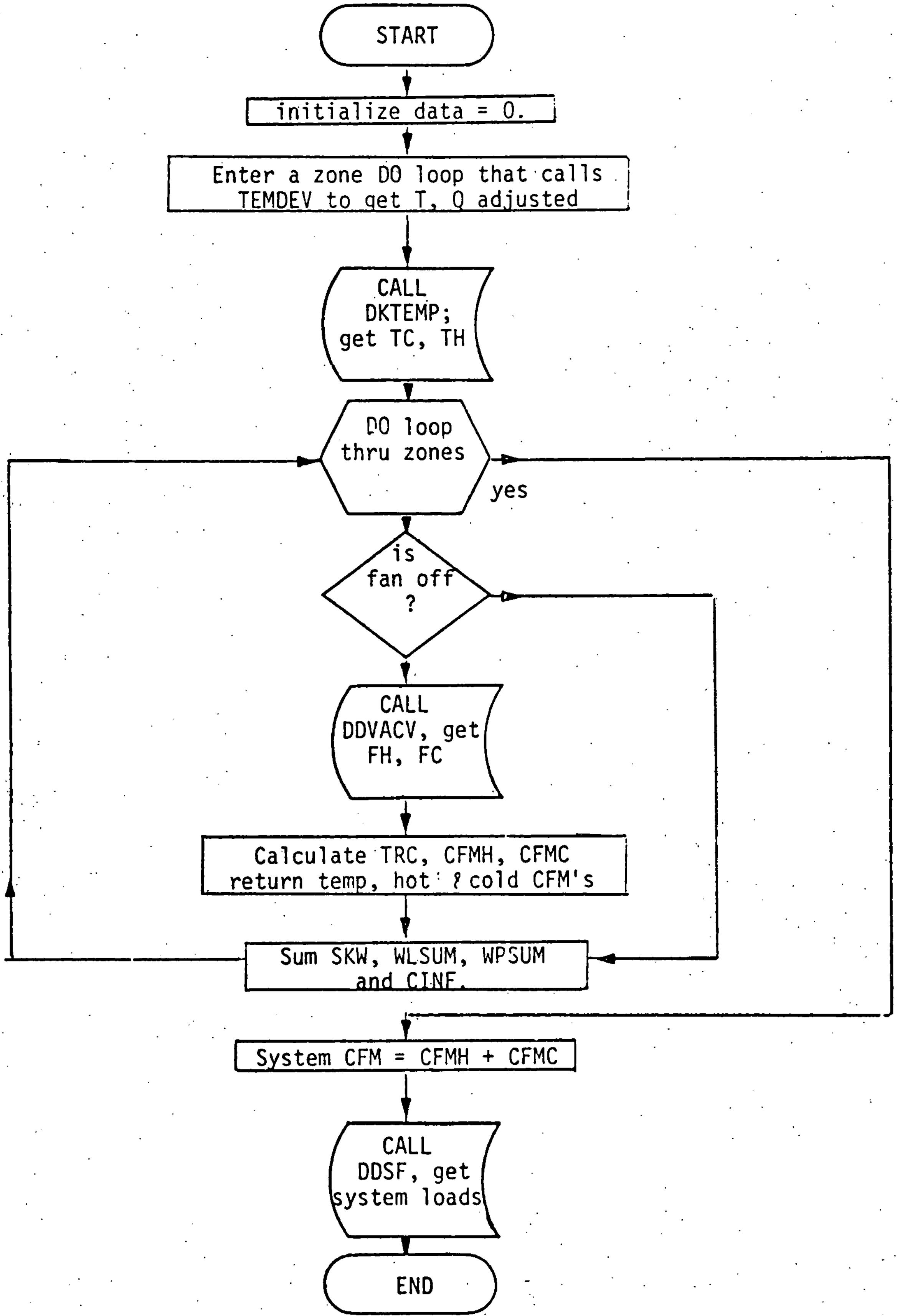
b. DOUBLE Calculational Sequence. Subroutine DOUBLE is called by the SYSTEMS Program as the driver subroutine for the multizone fan system and the dual-duct fan system. Parameters and variables used in the simulation are initialized or set equal to zero. The routine enters a DO loop on all the zones serviced by the system and calls TEMDEV to get the adjusted temperature and loads in each zone. Subroutine DKTEMP is then called to get the hot- and cold-deck temperatures.

The routine enters a second zone DO loop and checks for operation of the fan. If the fan is off, control passes to the summation of electrical loads. If the fan is operating, subroutine DDVACV is called to calculate the hot- and cold-duct air flow ratios. Results from this loop are the summation of the hot- and cold-duct air flows, the normalized return air temperatures, and electrical loads.

After the zone DO loop is completed, the total system air flow is calculated. Subroutine DDSF is called to get the primary system heating and cooling loads.

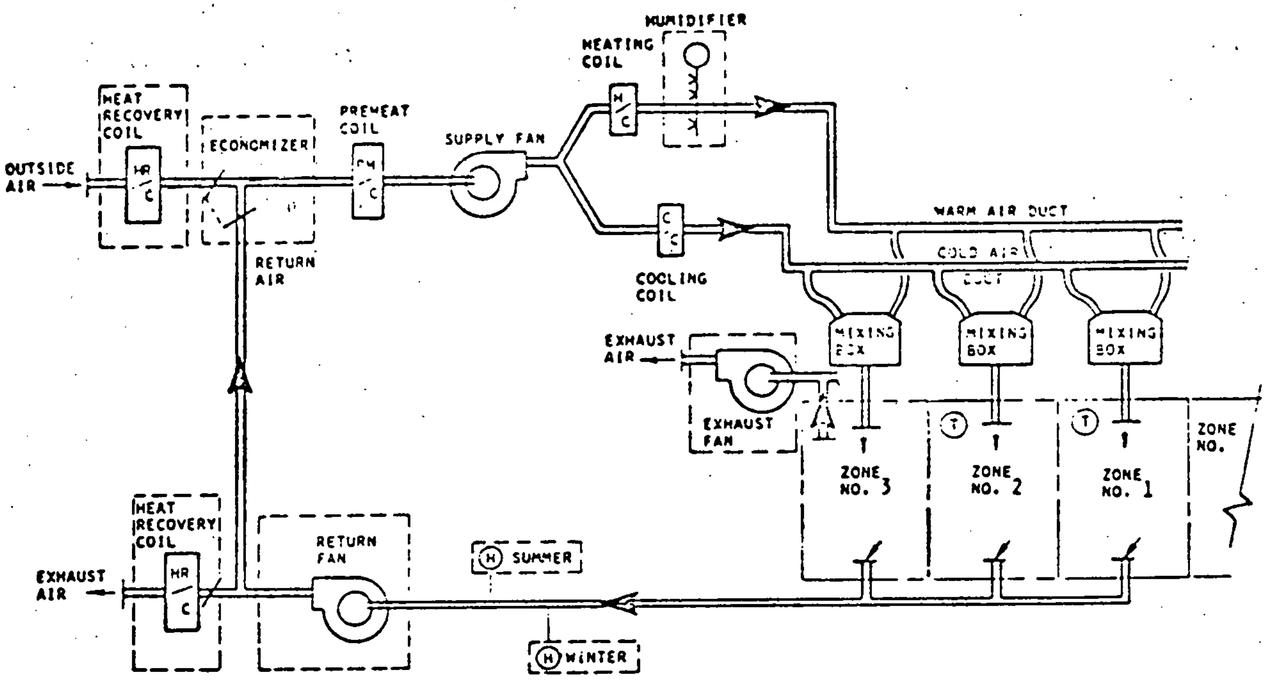
Now subroutine DOUBLE does not contain simulation of reheat capability. However, that simulation is being developed and may be completed in time to be included in the October 1, 1977, version of SYSTEMS.

c. DOUBLE Functional Flow Chart.


START
initialize data = 0.
Enter a zone DO loop that calls TEMDEV to get T, Q adjusted
CALL DKTEMP; get TC, TH
DO loop thru zones
yes
is fan off ?
CALL DDVACV, get FH, FC
Calculate TRC, CFMH, CFMC return temp, hot & cold CFM's
Sum SKW, WLSUM, WPSUM and CINF.
System CFM = CFMH + CFMC
CALL DDSF, get system loads
END

3. System No. 3 - Double-Duct Fan System

a. General Description. The double-duct fan system consists of a blow-through supply fan, a return fan, heating and cooling coils in parallel, a humidifier in the hot-air duct, and an economizer. The hot- and cold-air streams are mixed in mixing boxes located near the zones they service to control the space temperature. The preheat coil and humidity control in each zone have not yet been implemented. See Fig. IV.3.



—ITEMS SHOWN IN DASHED BOXES ARE OPTIONAL COMPONENTS

Fig. IV.3. System No. 3 - Dual-duct fan system.

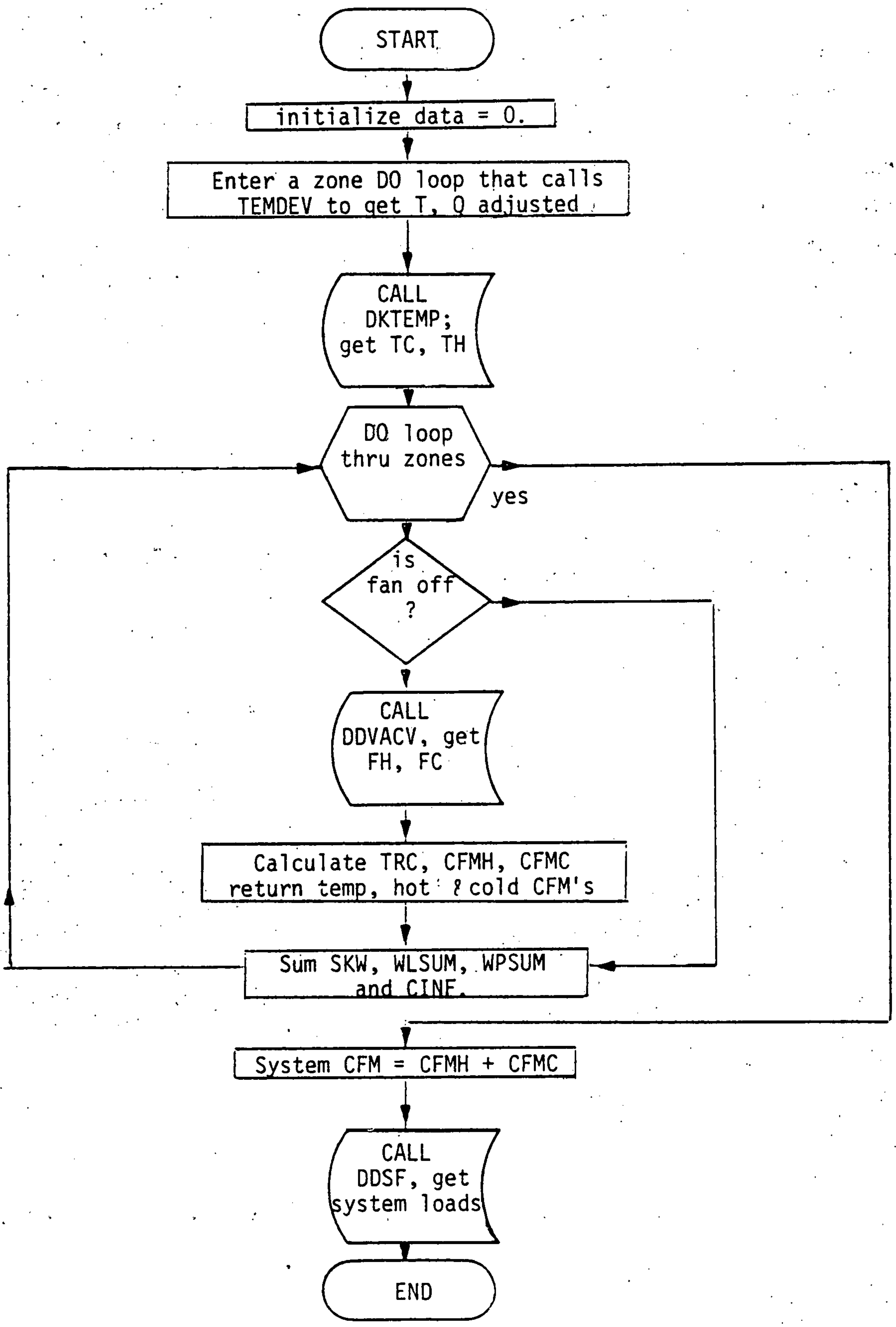
b. DOUBLE Calculational Sequence. Subroutine DOUBLE is called by the SYSTEMS Program as the driver subroutine for the multizone fan system and the dual-duct fan system. Parameters and variables used in the simulation are initialized or set equal to zero. The routine enters a DO loop on all the zones serviced by the system and calls TEMDEV to get the adjusted temperature and loads in each zone. Subroutine DKTEMP is then called to get the hot- and cold-deck temperatures.

The routine enters a second zone DO loop and checks for operation of the fan. If the fan is off, control passes to the summation of electrical loads. If the fan is operating, subroutine DDVACV is called to calculate the hot- and cold-duct air flow ratios. Results from this loop are the summation of the hot- and cold-duct air flows, the normalized return air temperatures, and electrical loads.

After the zone DO loop is completed, the total system air flow is calculated. Subroutine DDSF is called to get the primary system heating and cooling loads.

Now subroutine DOUBLE does not contain simulation of reheat capability. However, that simulation is being developed and may be completed in time to be included in the October 1, 1977, version of SYSTEMS.

c. DOUBLE Functional Flow Chart.


START
initialize data = 0.
Enter a zone DO loop that calls TEMDEV to get T, Q adjusted
CALL DKTEMP; get TC, TH
DO loop thru zones
yes
is fan off ?
CALL DDVACV, get FH, FC
Calculate TRC, CFMH, CFMC return temp, hot & cold CFM's
Sum SKW, WLSUM, WPSUM and CINF.
System CFM = CFMH + CFMC
CALL DDSF, get system loads
END

4. System No. 4 - Single-Zone Fan System with Subzone Induction Boxes

a. General Description. The single-zone fan with subzone induction boxes and optional reheat system consists of a draw-through supply fan, a return fan, preheating and heating coils, cooling coils, face and bypass dampers, an economizer, a humidifier, and induction boxes and reheat coils in each subzone. This system is designed primarily to serve one large central zone that requires cooling throughout the year. The subzones have induction boxes that can induce a maximum of 50% room air. Reheating of the mixed air is supplied if needed. The preheating and heating coils, the face and bypass dampers, and the zone humidity control have not yet been implemented. See Fig. IV.4.

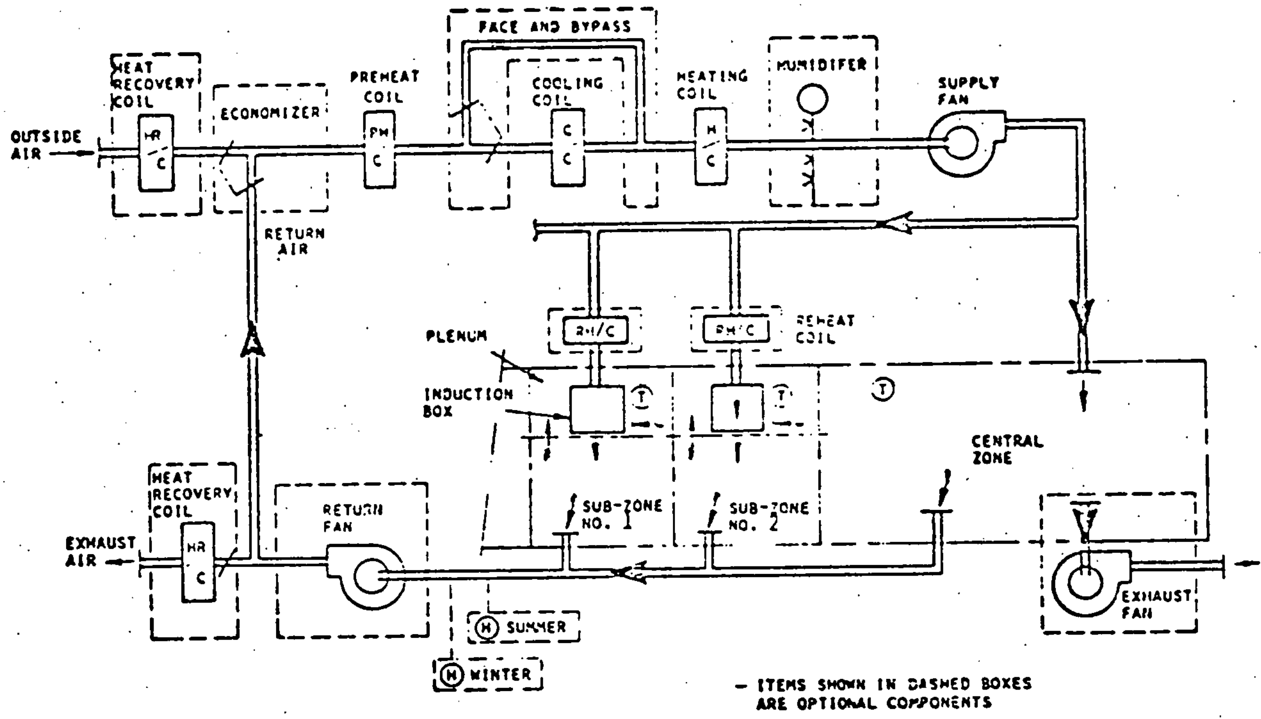


Fig. IV.4. System No. 4 - Single-zone fan system with subzone induction mixing boxes.

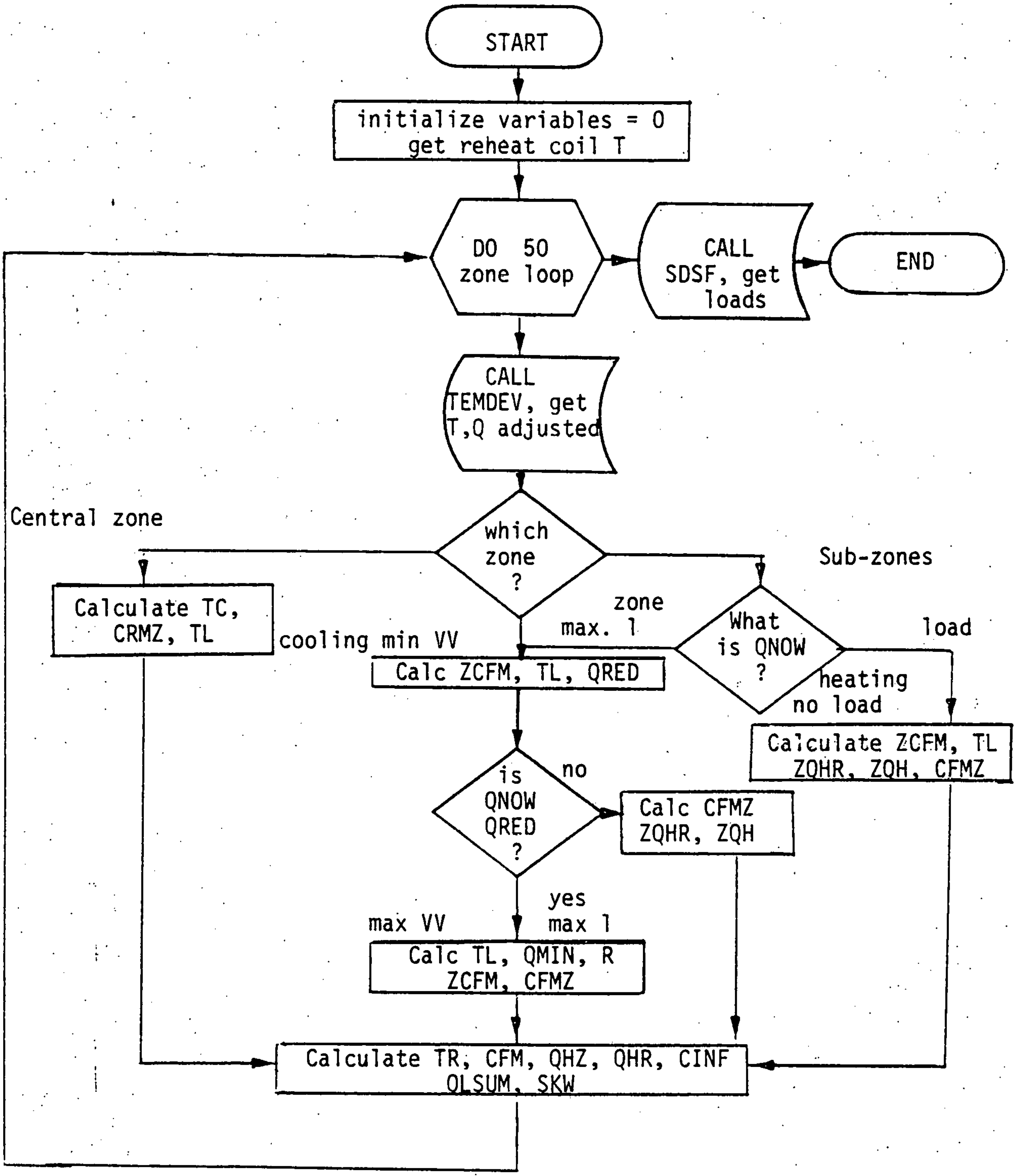
b. SZVV Calculational Sequence. Subroutine SZVV is called by the SYSTEMS Program as the driver subroutine for the single-zone fan system with induction boxes and reheat coils. Parameters and variables used in the subroutine are initialized or set equal to zero. The reheat coil ΔT is compared with zero. The reheat coils are burned off if the ΔT is equal to zero.

The routine enters a DO loop of all the zones serviced by this system. Subroutine TEMDEV is called to get adjusted loads and temperatures for each zone. A check is made for zone number. If the zone number is equal to one, the central-zone air-flow rate, cold-duct temperature, and the temperature at the lights are calculated. Contol then passes to the load-summing part of the loop. If the zone is a subzone, a check is made to determine the type of load. If the subzone has no load or a heating load, the zone air flow, reheat coil load, zone heat load, and temperature at the lights are calculated. Control then passes to the load-summing part of the loop. If the subzones require further cooling, the load is split between the induced and primary air streams. The zone air-flow rate, the temperature at the lights, and a reduced load are calculated for the case of minimum primary air and maximum induced air. The reduced load is then compared with the adjusted space load. If the zone adjusted load is less than the calculated reduced load, the zone air-flow rate, heat load, and reheat loads are calculated, and control passes to the load summing part of the loop. If the zone adjusted load is greater than the calculated reduced load, the zone air-flow rate, induced air ratio, temperature at the lights, and a new primary load are calculated for the case of maximum primary air and maximum induced air flows. Results of this DO loop are the calculation of the return air-stream temperature and the summation of system primary air flow, zone heat loads, zone reheat loads, infiltration, and latent and electrical loads.

After the zone DO loop is completed, subroutine SDSF is called to get the total system heating and cooling loads.

Now it has been determined that the present version of subroutine SZVV will produce incorrect results, and a revision is in progress. It is hoped that the corrected SZVV will be ready for inclusion in the October 1, 1977, version of SYSTEMS.

c. SZVV Functional Flow Chart.



5. System No. 5 - Unit Heater

a. General Description. The unit heater system consists of a draw-through fan and heating coil, whose function is to heat a space. The unit heater is primarily designed for one space, but can be used for several spaces. The heating coil is controlled by the first space, and the system assumes total recirculation. See Fig. IV.5.

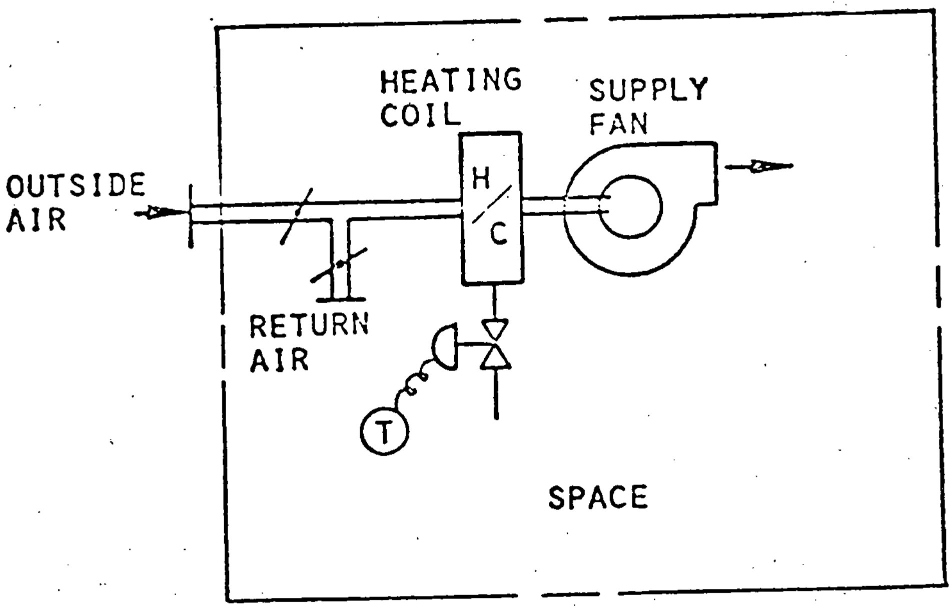


Fig. IV.5. Unit heater.

b. UNITHV Calculational Sequence. Subroutine UNITHV is called by the SYSTEMS Program to simulate a unit heater or a unit ventilator system. Parameters and variables used in the simulation are set at zero or are initialized from the data array. The routine enters a DO loop that sums the latent, plenum, electrical, and infiltration loads, and calculates a normalized return air temperature for all the zones served by this system. A check is made to determine whether or not the fan is operating. If the fan is operating, the zone flow rate, percentage of minimum outside air, and the hot-deck temperatures are set. The change in zone specific humidity and an adjusted return air temperature are calculated. If the fan is not operating, control is diverted to the determination of the system loads.

The routine checks whether the percentage minimum outside air is zero to determine whether the system is a unit heater or unit ventilator.

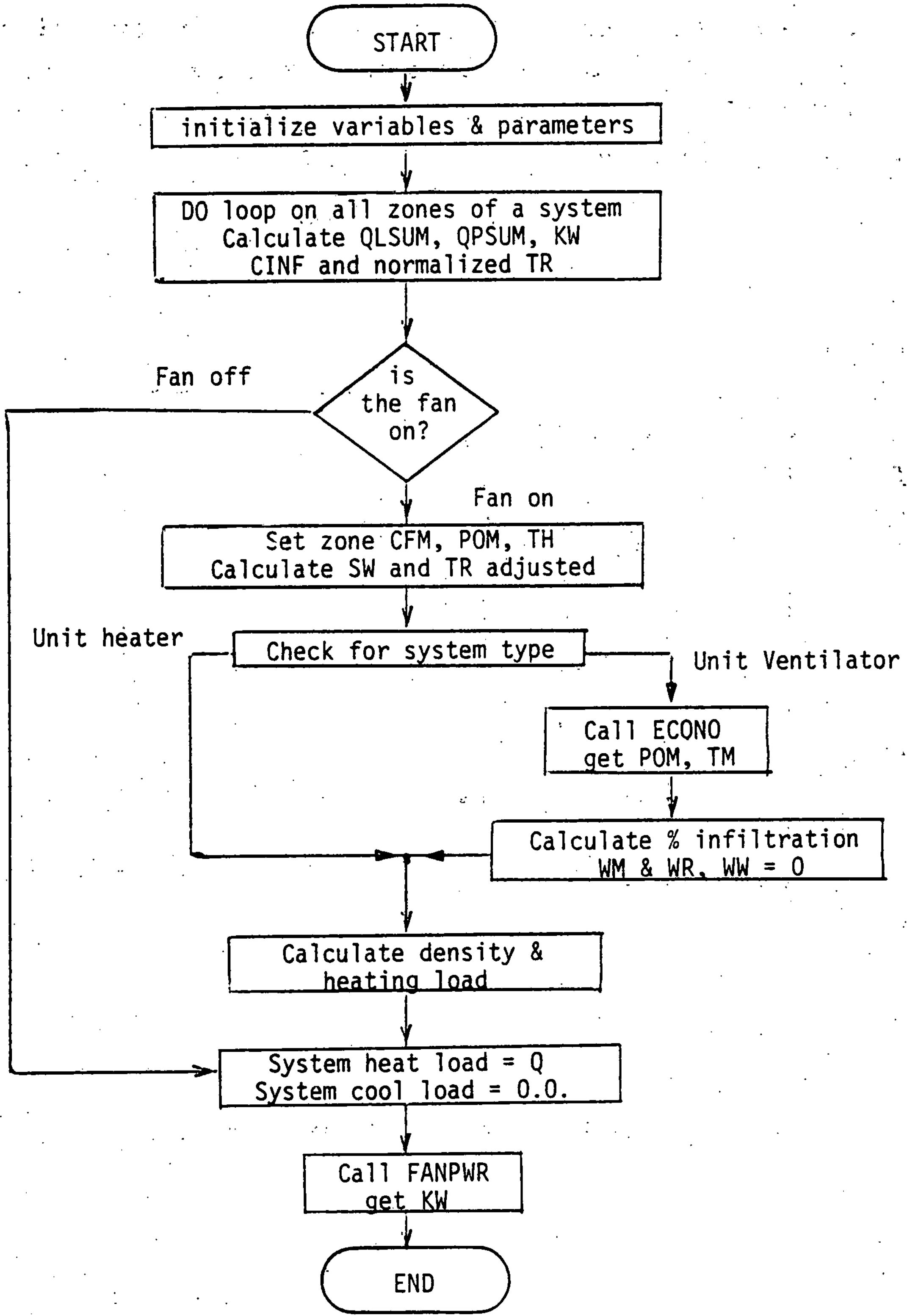
The simulation of a unit ventilator is set for heating mode only. The subroutine ECONO is called and returns the mixed air temperature based on the given percentage of outside air. The percentage of infiltration and the mixed and return air specific humidities are calculated. The specific humidity removal from the system is set at zero.

The simulation of the unit heater and the continuation of the unit ventilator simulation are the same. The air density is calculated based on the specific volume of outside air. The system load is calculated based on the difference in enthalpy between the mixed air and hot-deck air.

The system heating load is then determined by choosing the minimum algebraic value, given the system load and zero load. The system cooling load is set at zero. Control for fan not operating returns for this part of the subroutine.

The subroutine FANPWR is called and returns the electrical energy consumption of the fan. Control returns to the SYSTEMS Program.

c. UNITHV Functional Flow Chart.



6. System No. 6 - Unit Ventilator

a. General Description - The unit ventilator system consists of a draw-through fan, heating coil, and dampers, whose function is to heat and ventilate a space by conditioning outside air. The heating coil is controlled by the first space, and the dampers are set at a fixed percentage of outside air. The unit ventilator can be used to simulate more than one space. See Fig. IV.6.

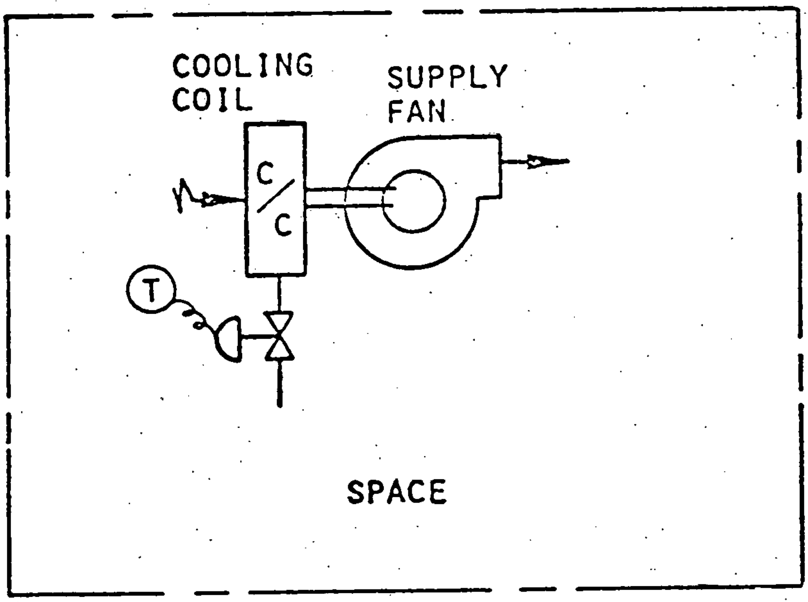


Fig. IV.6. System No. 6 - Unit ventilator.

b. UNITHV Calculational Sequence. Subroutine UNITHV is called by the SYSTEMS Program to simulate a unit heater or a unit ventilator system. Parameters and variables used in the simulation are set at zero or are initialized from the data array. The routine enters a DO loop that sums the latent, plenum, electrical, and infiltration loads, and calculates a normalized return air temperature for all the zones served by this system. A check is made to determine whether or not the fan is operating. If the fan is operating, the zone flow rate, percentage of minimum outside air, and the hot-deck temperatures are set. The change in zone specific humidity and an adjusted return air temperature are calculated. If the fan is not operating, control is diverted to the determination of the system loads.

The routine checks whether the percentage minimum outside air is zero to determine whether the system is a unit heater or unit ventilator.

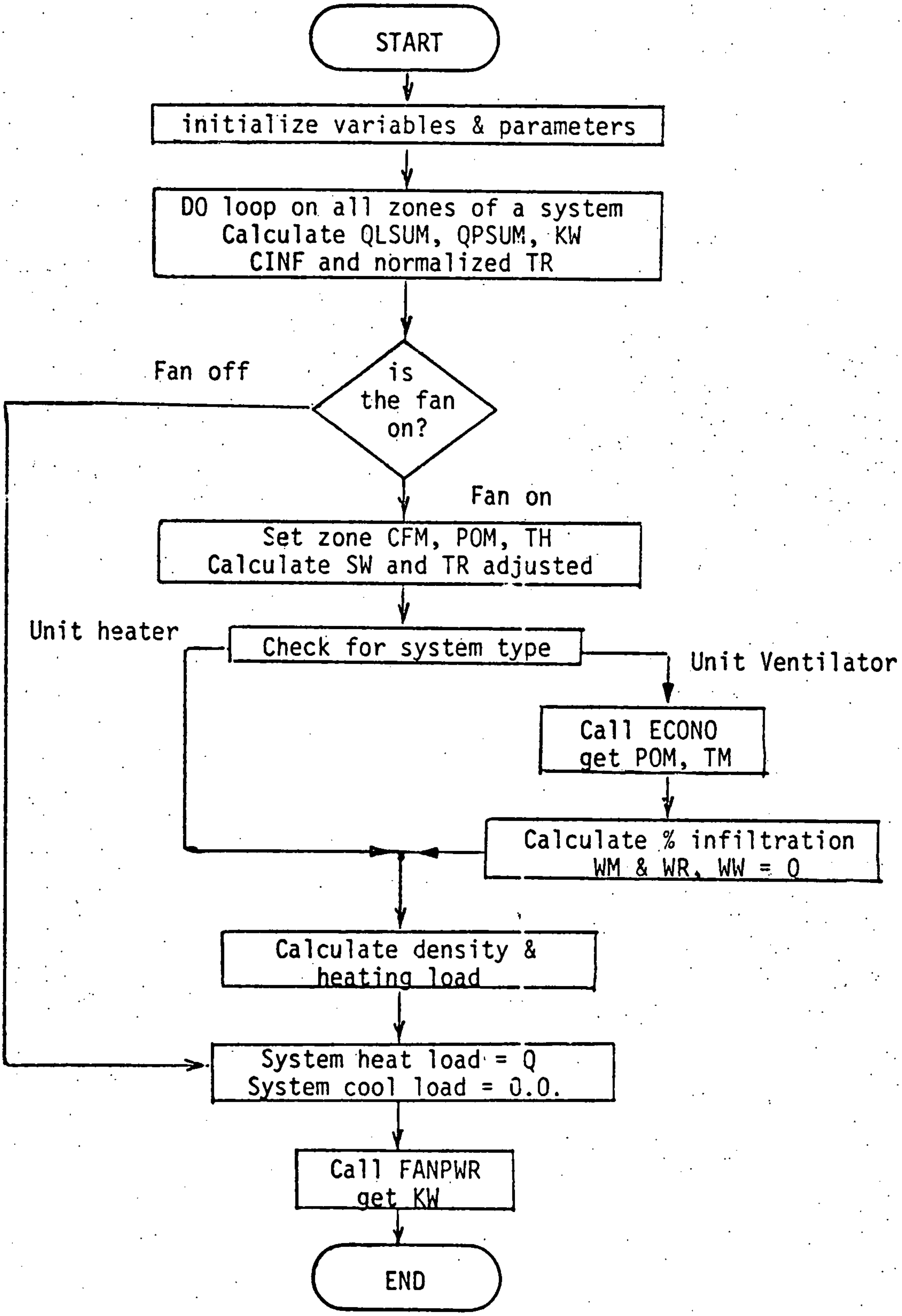
The simulation of a unit ventilator is set for heating mode only. The subroutine ECONO is called and returns the mixed air temperature based on the given percentage of outside air. The percentage of infiltration and the mixed and return air specific humidities are calculated. The specific humidity removal from the system is set at zero.

The simulation of the unit heater and the continuation of the unit ventilator simulation are the same. The air density is calculated based on the specific volume of outside air. The system load is calculated based on the difference in enthalpy between the mixed air and hot-deck air.

The system heating load is then determined by choosing the minimum algebraic value, given the system load and zero load. The system cooling load is set at zero. Control for fan not operating returns for this part of the subroutine.

The subroutine FANPWR is called and returns the electrical energy consumption of the fan. Control returns to the SYSTEMS Program.

c. UNITHV Functional Flow Chart.



7. System No. 7 - Floor-Panel Heating System

a. General Description. The floor-panel heating system consists of a radiant-panel system embedded in a slab and a central air-conditioning unit. Single-duct, double-duct, and variable-volume air handler simulations could be used with the radiant-panel system. The heat-transfer calculations to simulate the floor-panel heating system have not yet been implemented, but it is hoped that they will be ready for inclusion in the October 1, 1977, version of SYSTEMS. See Fig. IV.7.

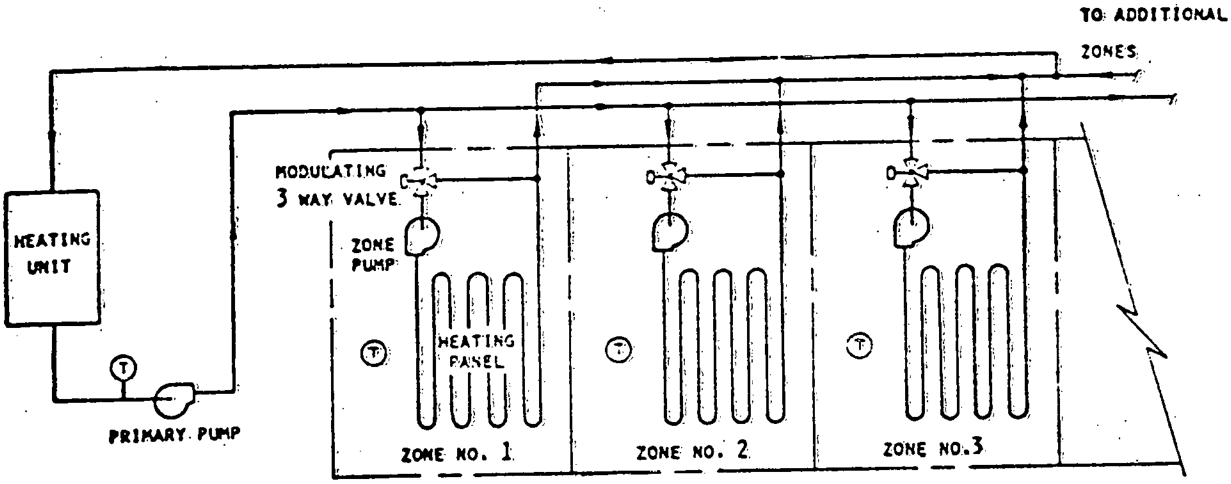


Fig. IV.7. System No. 7 - Floor-panel heating system.

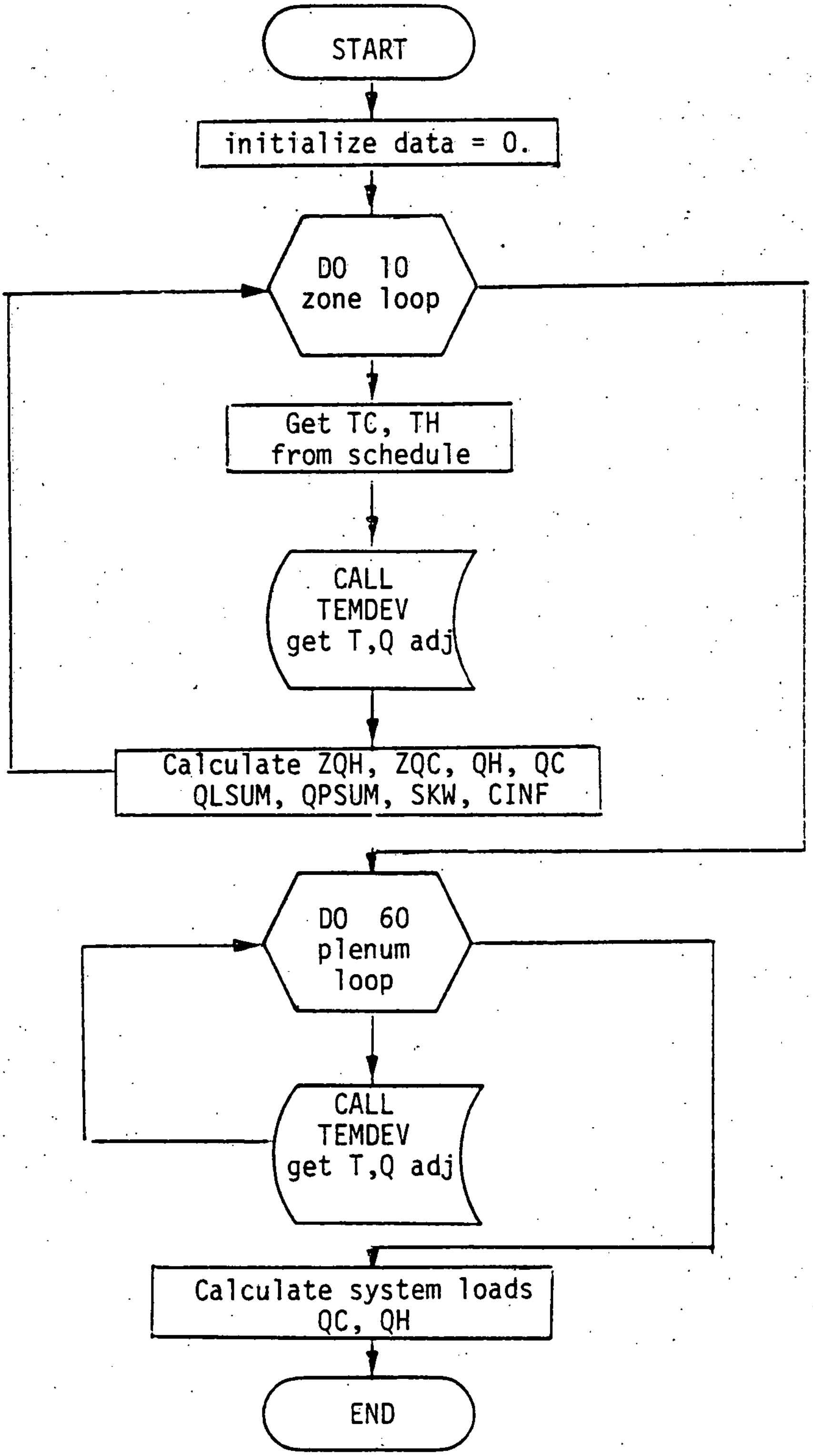
b. PANEL Calculational Sequence. Subroutine PANEL is called by the SYSTEMS Program to simulate a floor-panel heating system. Parameters and variables to be used are initialized or set equal to zero.

The subroutine enters a DO loop on all the zones serviced by this system. The upper and lower limits of the throttling range are obtained from the appropriate schedules. Subroutine TEMDEV is called and returns the adjusted temperatures and sensible loads in each zone. The heating and cooling loads are checked for the correct algebraic sign and are summed. Results of this loop are the sums of plenum, latent, electrical, infiltration, and building sensible loads.

After completion of the zone loop, the subroutine enters a plenum loop. Subroutine TEMDEV is called to get adjusted temperatures and loads in each plenum.

Results of the subroutine are the sums of latent, plenum, electrical, infiltration, and adjusted building loads for all zones and plenums.

c. PANEL Functional Flow Chart.



8. System No. 8 - Two-Pipe Fan-Coil System

a. General Description. The two-pipe fan-coil system consists of a flow-through fan and one water coil that provide heating or cooling for each zone. One water distribution circuit, two-pipe, serves all the fan-coil units in a building. A changeover mechanism, based on ambient dry-bulb temperature, is used to determine the temperature of the circulated water. The calculation of energy required for heating/cooling changeover has not been implemented. A fixed quantity of outside ventilation air is provided as an option. See Fig. IV.8.

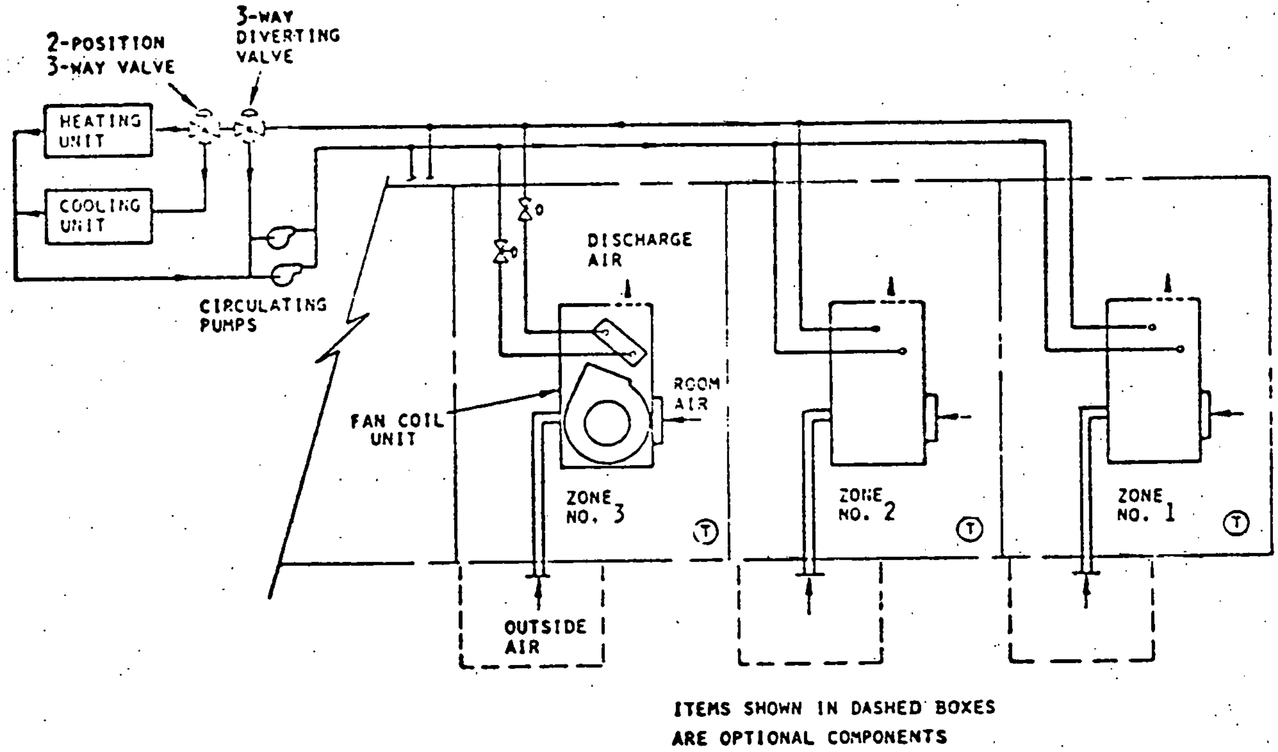


Fig. IV.8. System No. 8 - Two-pipe fan-coil system.

b. FCOIL Calculational Sequence. Subroutine FCOIL is called by the SYSTEMS Program as the driver subroutine for the two-pipe fan-coil and four-pipe fan-coil system simulations. Parameters and variables are initialized or set equal to zero. A check is made for system type. If the system is a four-pipe fan-coil unit, control passes to the zone DO loop. If the system is a two-pipe fan-coil unit, the dry-bulb temperature is compared to the changeover temperature to determine whether the coil is heating or cooling.

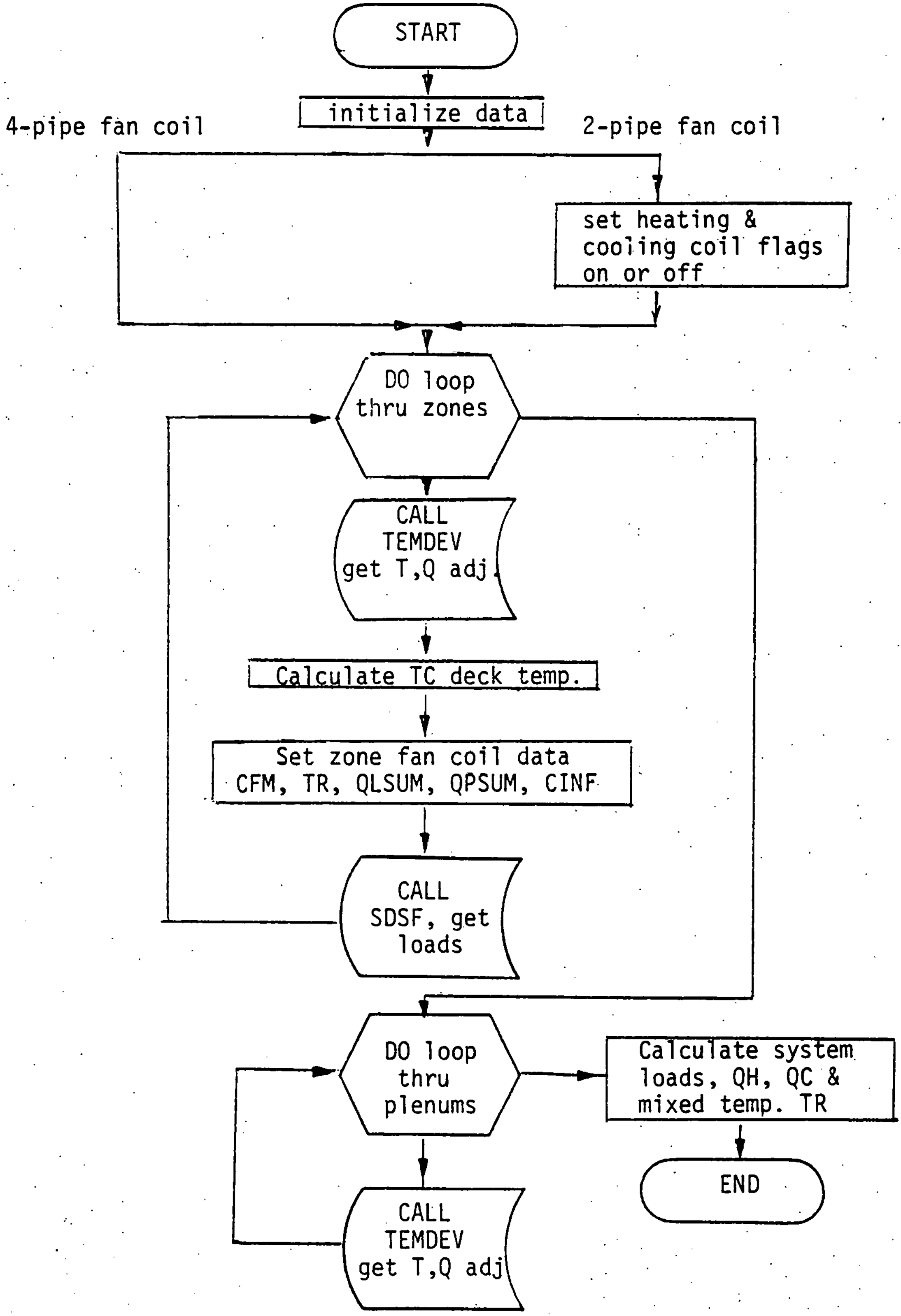
The subroutine enters a DO loop on all the zones served by fan-coil units. Subroutine TEMDEV is called to get adjusted temperatures and loads in each zone. The necessary cold-deck temperature is calculated and zone air flow, return air temperature, and latent and plenum loads are defined for each zone. Subroutine SDSF is called to get heating and/or cooling loads. Results of this loop include electrical, latent, infiltration, and heating and/or cooling loads.

After completion of the zone DO loop, the subroutine enters a plenum loop. Subroutine TEMDEV is called to get adjusted temperatures and loads in each plenum.

Results of this subroutine are the electrical, latent, infiltration, heating and/or cooling loads, and the mixed-stream return-air temperature for all zones.

The condition mode is currently determined by a comparison of dry-bulb and changeover temperatures. This decision will be replaced by the entire changeover energy-calculation algorithm.

c. FCOIL Functional Flow Chart.



9. System No. 9 - Four-Pipe Fan-Coil System

a. General Description. The four-pipe fan-coil system consists of a blow-through fan, a heating coil, and a cooling coil that provide air conditioning for each zone. Two water-distribution systems, the hot- and chilled-water circuits, serve all the fan-coil units in a building. A net heat gain in a space yields calculation of a cooling load if the cooling system is on, and a net heat loss yields the calculated heating load. A fixed quantity of outside ventilation air is provided as an option. See Fig. IV.9.

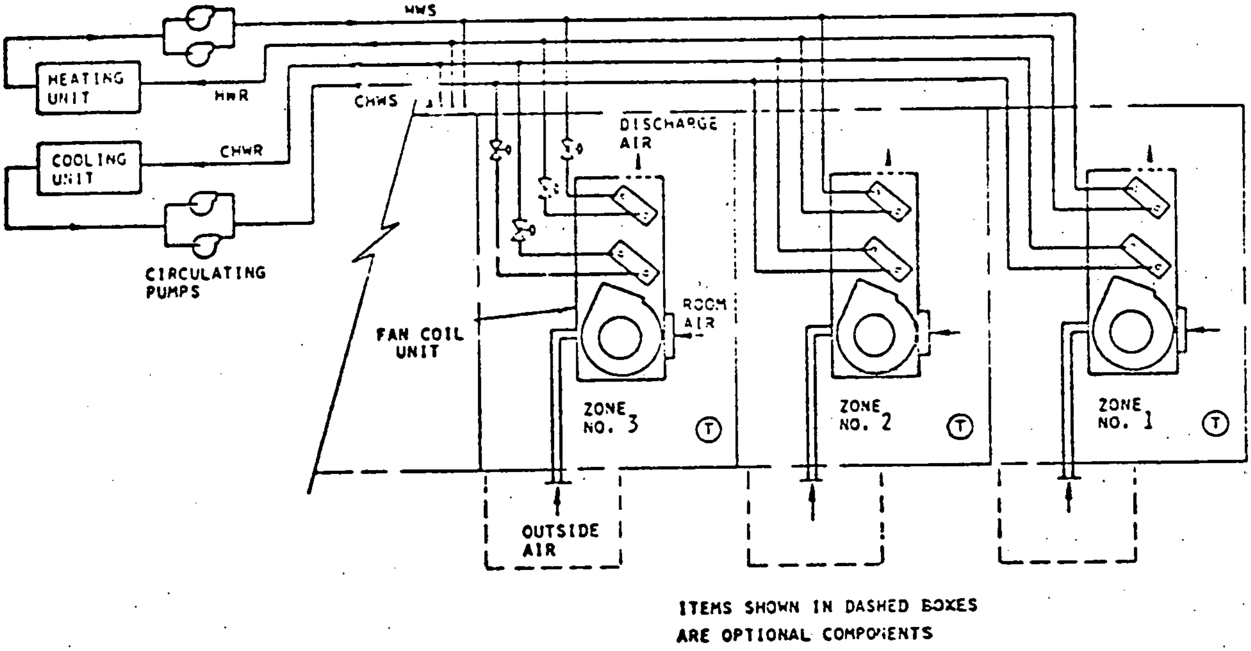


Fig. IV.9. System No. 9 - Four-pipe fan-coil system.

b. FCOIL Calculational Sequence. Subroutine FCOIL is called by the SYSTEMS Program as the driver subroutine for the two-pipe fan-coil and four-pipe fan-coil system simulations. Parameters and variables are initialized or set equal to zero. A check is made for system type. If the system is a four-pipe fan-coil unit, control passes to the zone DO loop. If the system is a two-pipe fan-coil unit, the dry-bulb temperature is compared to the changeover temperature to determine whether the coil is heating or cooling.

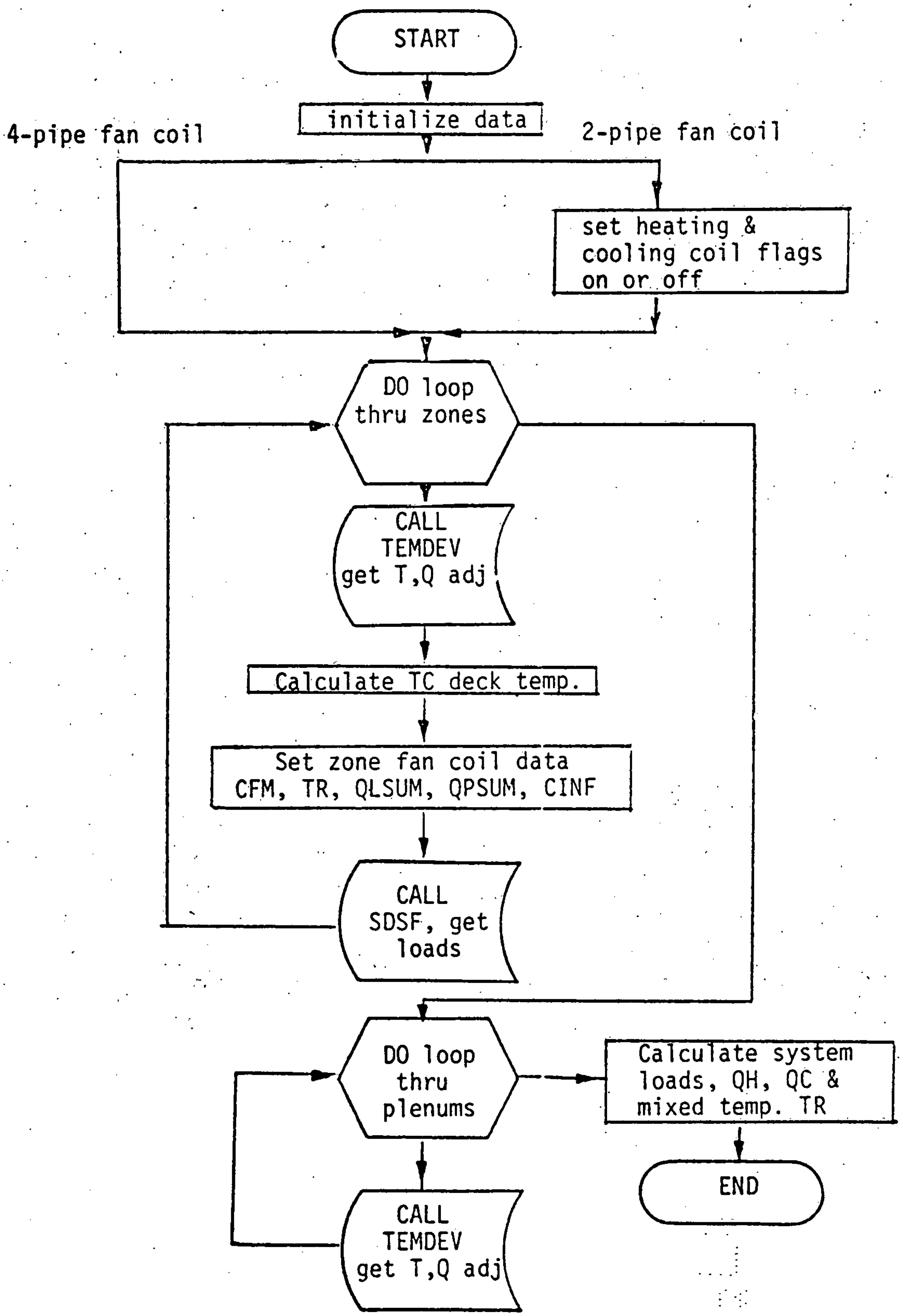
The subroutine enters a DO loop on all the zones served by fan-coil units. Subroutine TEMDEV is called to get adjusted temperatures and loads in each zone. The necessary coil-deck temperature is calculated and zone air flow, return air temperature, and latent and plenum loads are defined for each zone. Subroutine SDSF is called to get heating and/or cooling loads. Results of this loop include electrical, latent, infiltration, and heating and/or cooling loads.

After completion of the zone DO loop, the subroutine enters a plenum loop. Subroutine TEMDEV is called to get adjusted temperatures and loads in each plenum.

Results of this subroutine are the electrical, latent, infiltration, heating and/or cooling loads, and the mixed-stream return-air temperature for all zones.

The condition mode is currently determined by a comparison of dry-bulb and changeover temperatures. This decision will be replaced by the entire changeover energy-calculation algorithm.

c. FCOIL Functional Flow Chart.



10. System No. 10 - Two-Pipe Induction Unit Fan System

a. General Description. The two-pipe induction system uses forced air and a single (two-pipe) distribution system for circulated water to condition a space. The central primary air system includes heating and cooling coils, face and bypass dampers, a humidifier, and an economizer. The heating coil and bypass dampers have not been implemented.

The induction unit consists of a water coil for hot or chilled circulating water and a mixing chamber for primary and induced air. Primary air flow through the nozzles induces flow of room air across the induction coil. A changeover mechanism, based on ambient dry-bulb temperatures, is used to determine the temperature of the circulated water. The calculation of the energy required for heating/cooling changeover has not been implemented. See Fig. IV.10.

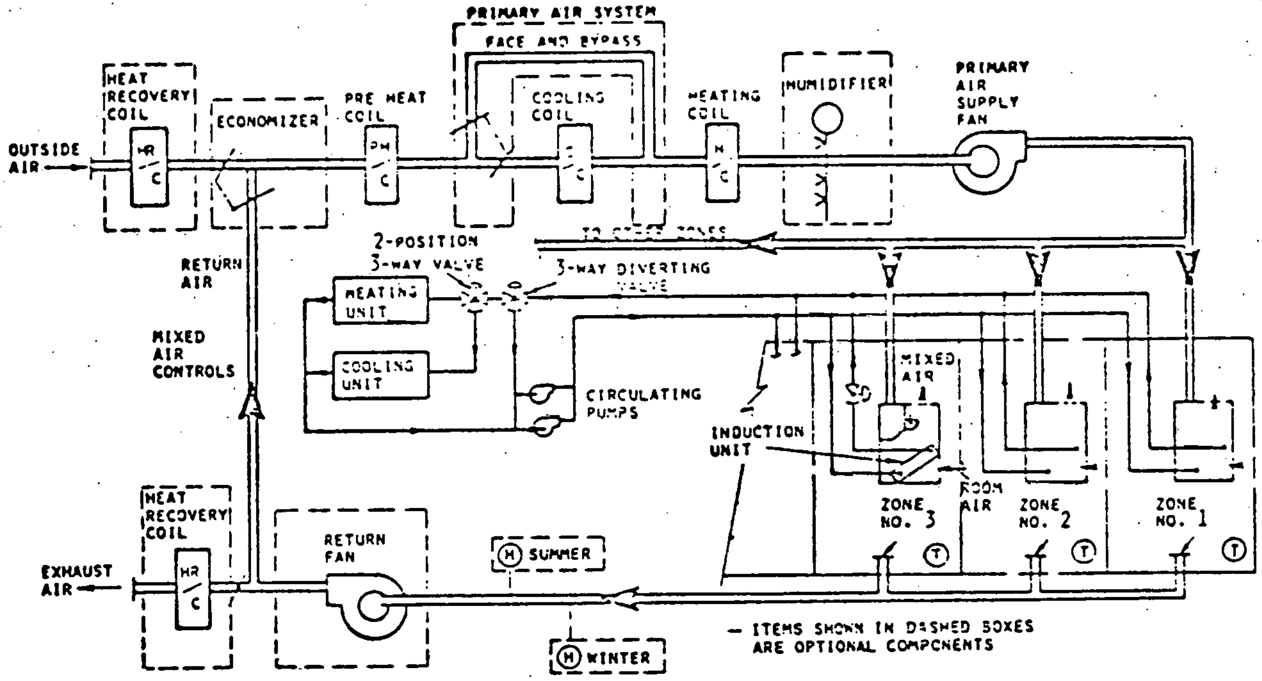


Fig. IV.10. System No. 10 - Two-pipe induction unit fan system.

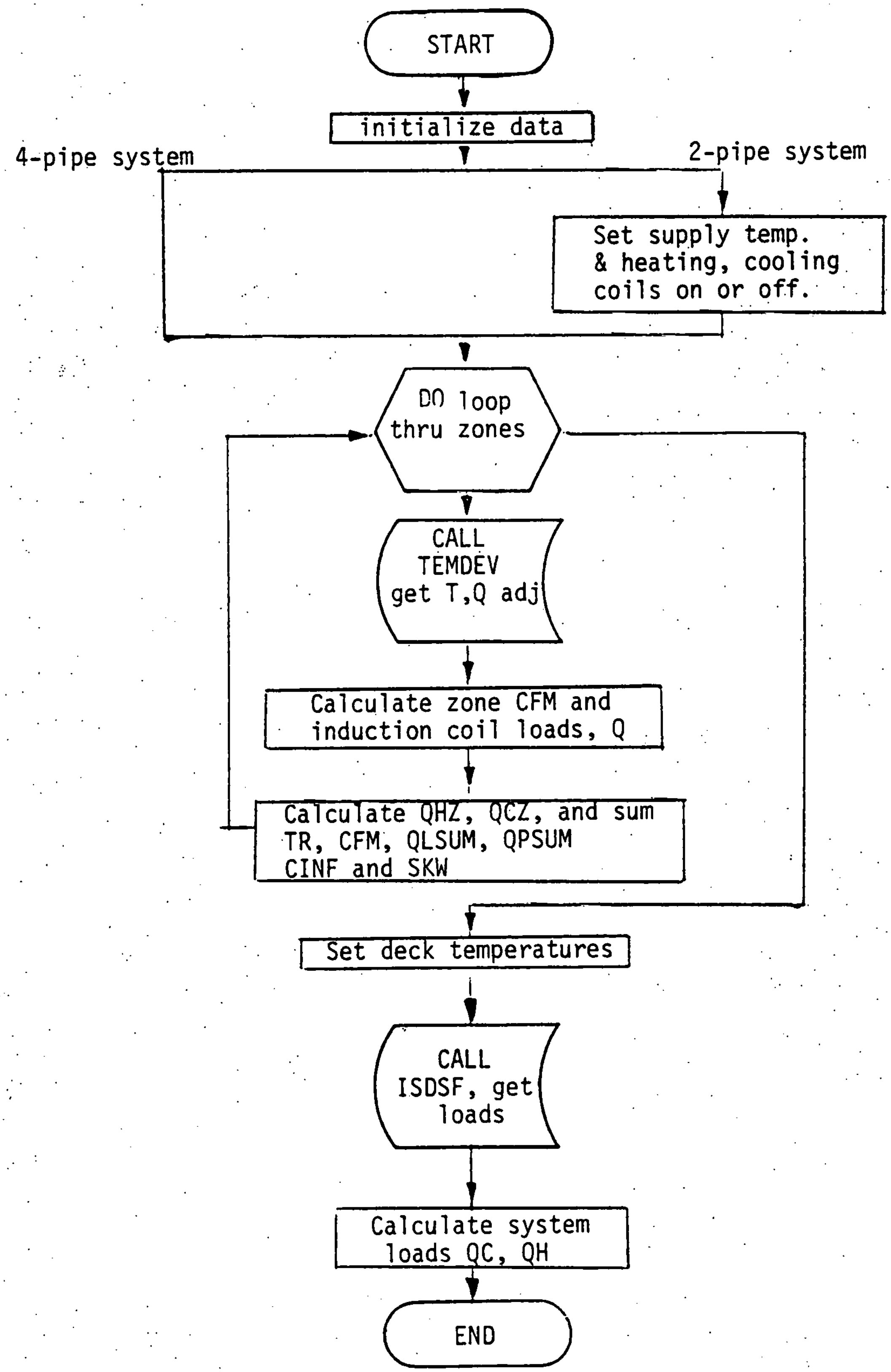
b. INDUC Calculational Sequence. Subroutine INDUC is called by the SYSTEMS Program as the driver subroutine for the two-pipe and four-pipe induction system simulations. Parameters and variables are initialized or set equal to zero, and the system type is checked. If the system is a four-pipe induction unit, control passes to the zone DO loop. If the system is a two-pipe induction unit, the ambient dry-bulb temperature is compared to the changeover temperature to determine the conditioning mode and calculate the primary air supply temperature.

The subroutine enters a DO loop on all zones served by induction units. Subroutine TEMDEV is called and returns the adjusted zone temperatures and loads. The primary air-flow rate zone induction-coil loads are calculated. Results of the zone loop are the sums of system air-flow rate, and plenum, infiltration, electrical, latent, and induction coil loads.

After completion of the zone loop, the hot- and cold-deck temperatures are defined. Subroutine ISDSF is called and returns the primary air heating and cooling loads.

The condition mode is currently determined by a comparison of dry-bulb and changeover temperatures. This decision will be replaced by the entire changeover energy calculation algorithm.

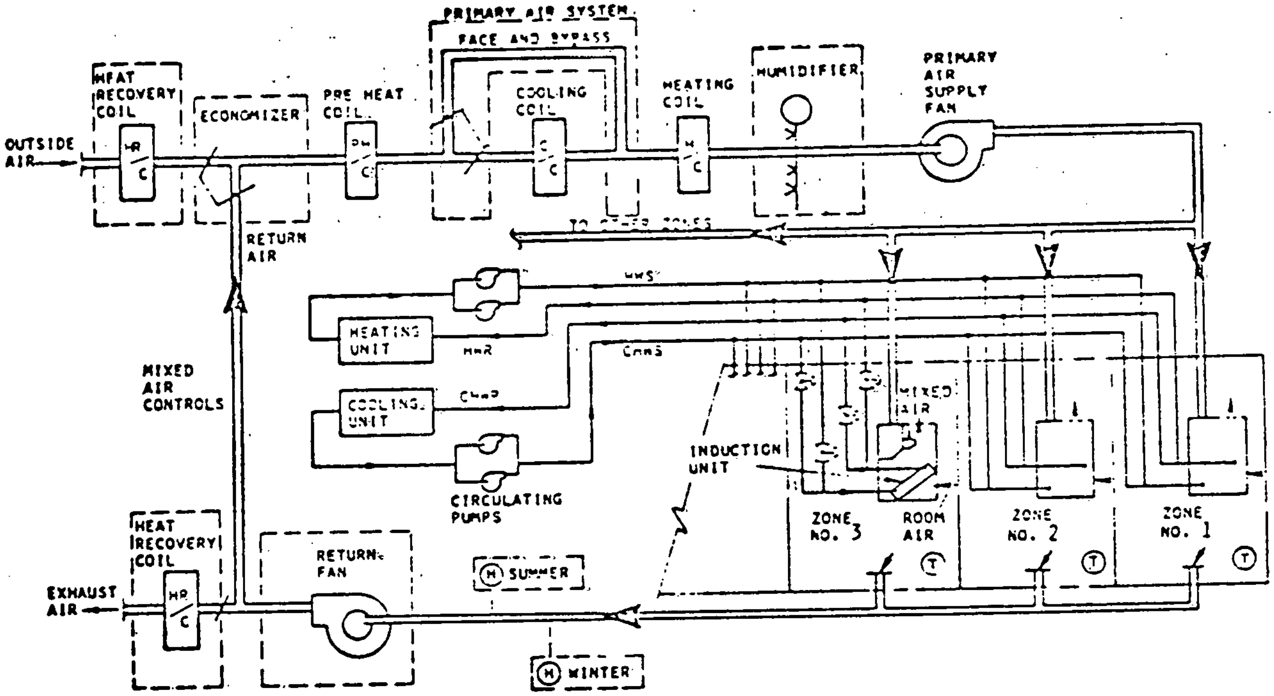
c. INDUC Functional Flow Chart.


START
initialize data
4-pipe system
2-pipe system
Set supply temp. & heating, cooling coils on or off.
DO loop thru zones
CALL TEMDEV get T,Q adj
Calculate zone CFM and induction coil loads, Q
Calculate QHZ, QCZ, and sum TR, CFM, QLSUM, QPSUM CINF and SKW
Set deck temperatures
CALL ISDSF, get loads
Calculate system loads QC, QH
END

11. System No. 11 -- Four-Pipe Induction Unit Fan System

a. General Description. The four-pipe induction system uses forced air and two distribution systems (hot and chilled) for circulated water to condition a space. The central primary air system includes heating and cooling coils, face and bypass dampers, a humidifier, and an economizer. The heating coil and bypass dampers have not yet been implemented.

The induction unit consists of a single water coil for hot or chilled circulating water and a mixing chamber of primary and induced air. Primary air flow through the nozzles induces room air to flow across the induction coil. Three-way valves at the inlet and outlet of the induction coil admit either hot or chilled water without mixing the streams. See Fig. IV.11.



—ITEMS SHOWN IN DASHED BOXES ARE OPTIONAL COMPONENTS

Fig. IV.11. System No. 11 - Four-pipe induction unit fan system.

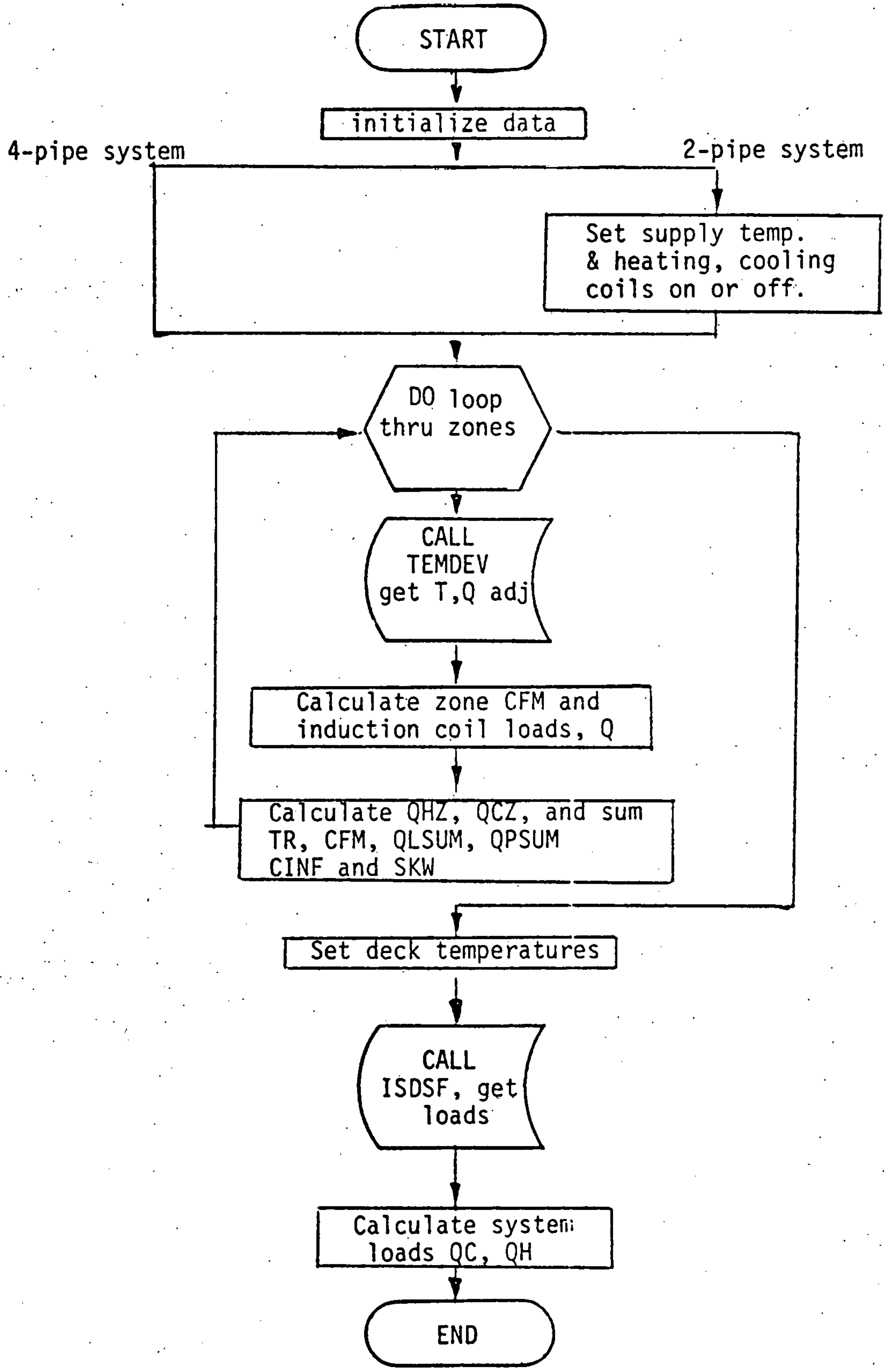
b. INDUC Calculational Sequence. Subroutine INDUC is called by the SYSTEMS Program as the driver subroutine for the two-pipe and four-pipe induction system simulations. Parameters and variables are initialized or set equal to zero, and the system type is checked. If the system is a four-pipe induction unit, control passes to the zone DO loop. If the system is a two-pipe induction unit, the ambient dry-bulb temperature is compared to the changeover temperature to determine the conditioning mode and calculate the primary air supply temperature.

The subroutine enters a DO loop on all zones served by induction units. Subroutine TEMDEV is called and returns the adjusted zone temperatures and loads. The primary air-flow rate and zone induction-coil loads are calculated. Results of the zone loop are the sums of system air-flow rate, and plenum, infiltration, electrical, latent, and induction coil loads.

After completion of the zone loop, the hot- and cold-deck temperatures are defined. Subroutine ISDSF is called and returns the primary air heating and cooling loads.

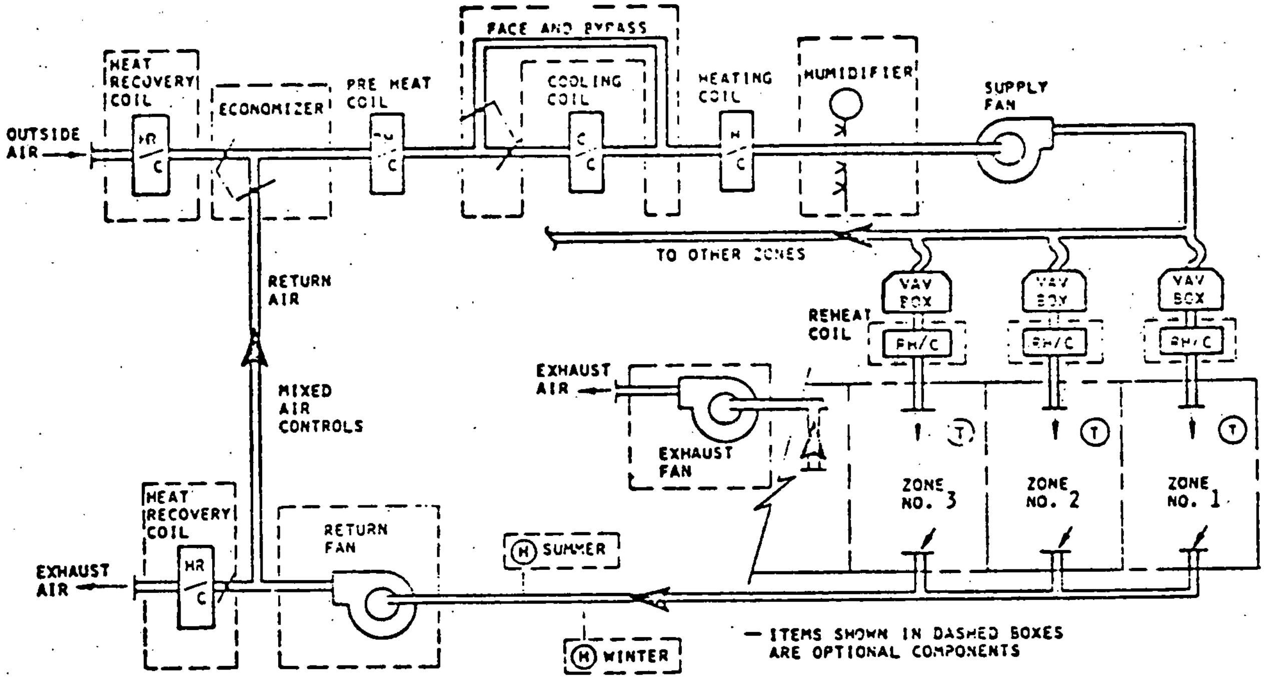
The condition mode is currently determined by a comparison of dry-bulb and changeover temperatures. This decision will be replaced by the entire changeover energy calculation algorithm.

c. INDUC Functional Flow Chart.


START
initialize data
4-pipe system
2-pipe system
Set supply temp. & heating, cooling coils on or off.
DO loop thru zones
CALL TEMDEV get T,Q adj
Calculate zone CFM and induction coil loads, Q
Calculate QHZ, QCZ, and sum TR, CFM, QLSUM, QPSUM CINF and SKW
Set deck temperatures
CALL ISDSF, get loads
Calculate system loads QC, QH
END

12. System No. 12 - Variable-Volume Fan System

a. General Description. The variable air volume fan system with optional reheat consists of a supply fan, a return fan, heating and cooling coils in series, face and bypass dampers, a humidifier, an economizer, variable-air volume (VAV) terminal boxes, and reheat coils in each zone. The heating coil and the face and bypass dampers have not been implemented. The VAV terminal boxes are controlled by thermostats in each zone and vary the air-flow rate into each zone. Optional reheat is available to reheat primary air supplied to zones with cooling loads smaller than the minimum provided by the VAV terminals. Both the supply and return fans have three options for volume control, variable motor speed, inlet vanes, or discharge damper control. See Fig. IV.12.



— ITEMS SHOWN IN DASHED BOXES ARE OPTIONAL COMPONENTS

Fig. IV.12. System No. 12 - Variable volume fan system.

b. VARVOL Calculational Sequence. Subroutine VARVOL is called by the SYSTEMS Program as the driver subroutine for the single-zone fan system, the variable-volume fan system with optional reheat, and the constant-volume reheat fan system. Parameters and variables used in the subroutine are initialized or set equal to zero. Subroutine DKTEMP is called to set the hot- and cold-deck temperatures.

The routine enters a loop on all the zones serviced by this system. TEMDEV is called to get the adjusted load and temperature for each zone. A check is then made for system type. If system type is a single-zone fan system, the cold-deck temperature and the zone air flow are calculated and control passes to the load summing part of the loop. If system type is either variable-volume or constant-volume reheat, a check is made for fan operation. If the fan is off, control is passed to the load-summing part of the loop. If the fan is on, the CFMIN variable is calculated. CFMIN can vary from zero to one for the variable-volume system, but must be set equal to one for the constant-volume reheat system. A check is made for sensible loads. If there is a heat load, control passes to the zone air flow and load-calculation portion of the loop. If there is a cooling load, the zone air flow is calculated and a check is made for any necessary reheating. If reheating is required, the zone reheat load is calculated and added to the zone heat load. If no reheat is required, control passes to the load summing part of the loop. Results of this DO loop are the calculations of zone air flows, heating and reheating loads, sensible cooling loads, and the sums of latent, plenum, electrical, and infiltration loads.

After the zone DO loop is completed, the hot- and cold-deck temperatures are reset and subroutine SDSF is called. The primary air system loads are calculated and added to the zone loads to get total system loads.

13. System No. 13 - Constant-Volume Reheat Fan System

a. General Description. The constant-volume fan system consists of constant volume supply and return fans, heating and cooling coils in series, face and bypass dampers, a humidifier, an economizer, and reheat coils in each zone. The heating coil and face and bypass dampers have not been implemented. The reheat coils reheat primary air in zones that require heating. See Fig. IV.13.

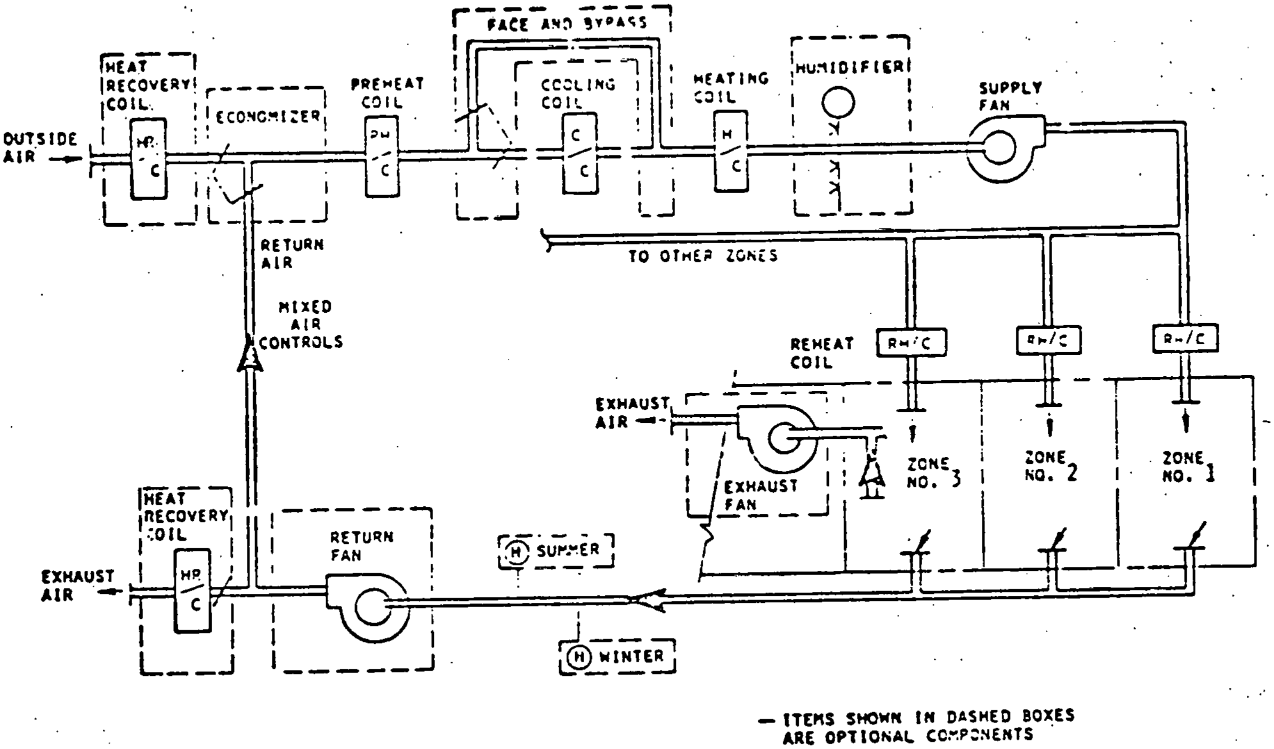


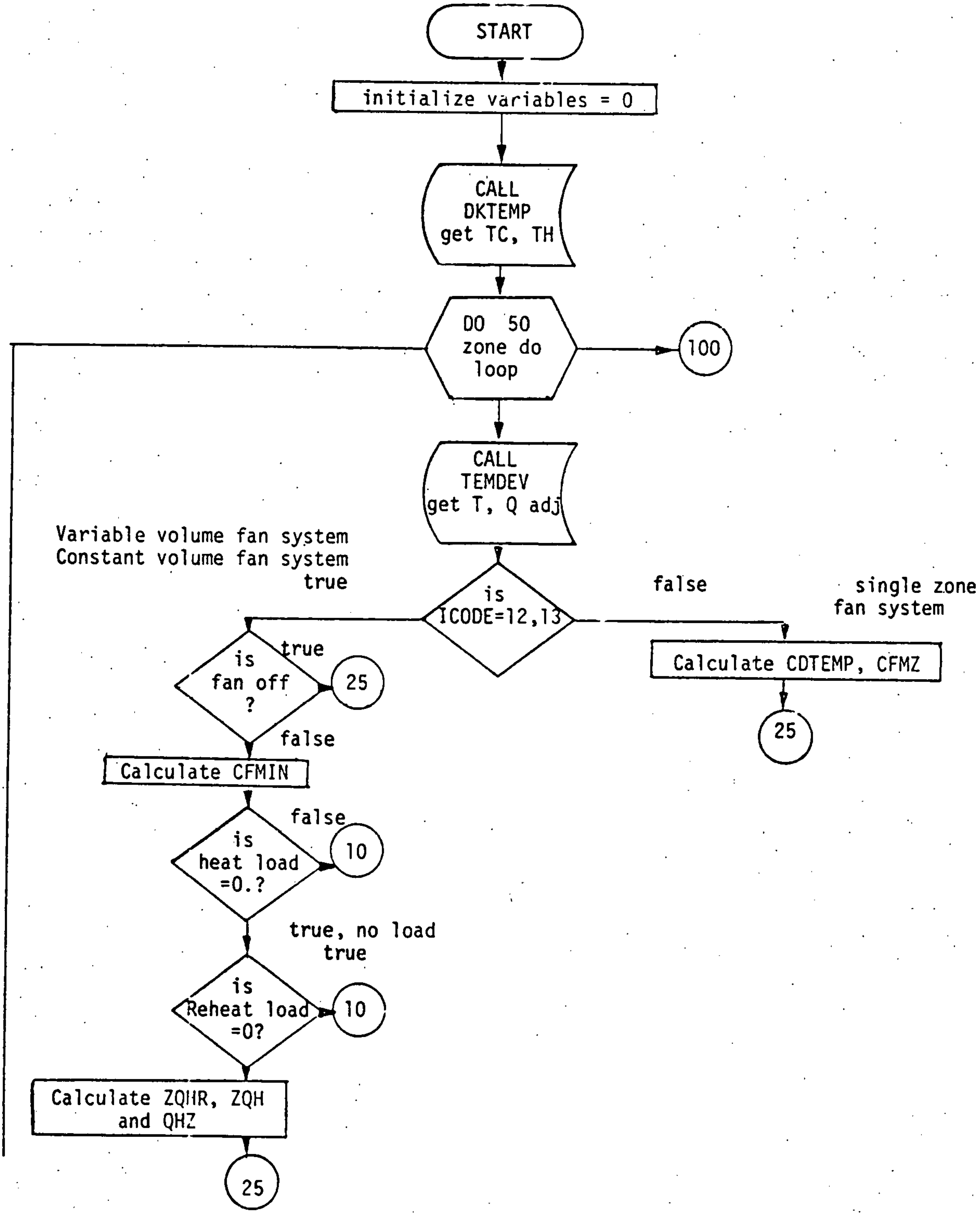
Fig. IV.13. System No. 13 - Constant-volume reheat fan system.

b. VARVOL Calculational Sequence. Subroutine VARVOL is called by the SYSTEMS Program as the driver subroutine for the single-zone fan system, the variable-volume fan system with optional reheat, and the constant-volume reheat fan system. Parameters and variables used in the subroutine are initialized or set equal to zero. Subroutine DKTEMP is called to set the hot- and cold-deck temperatures.

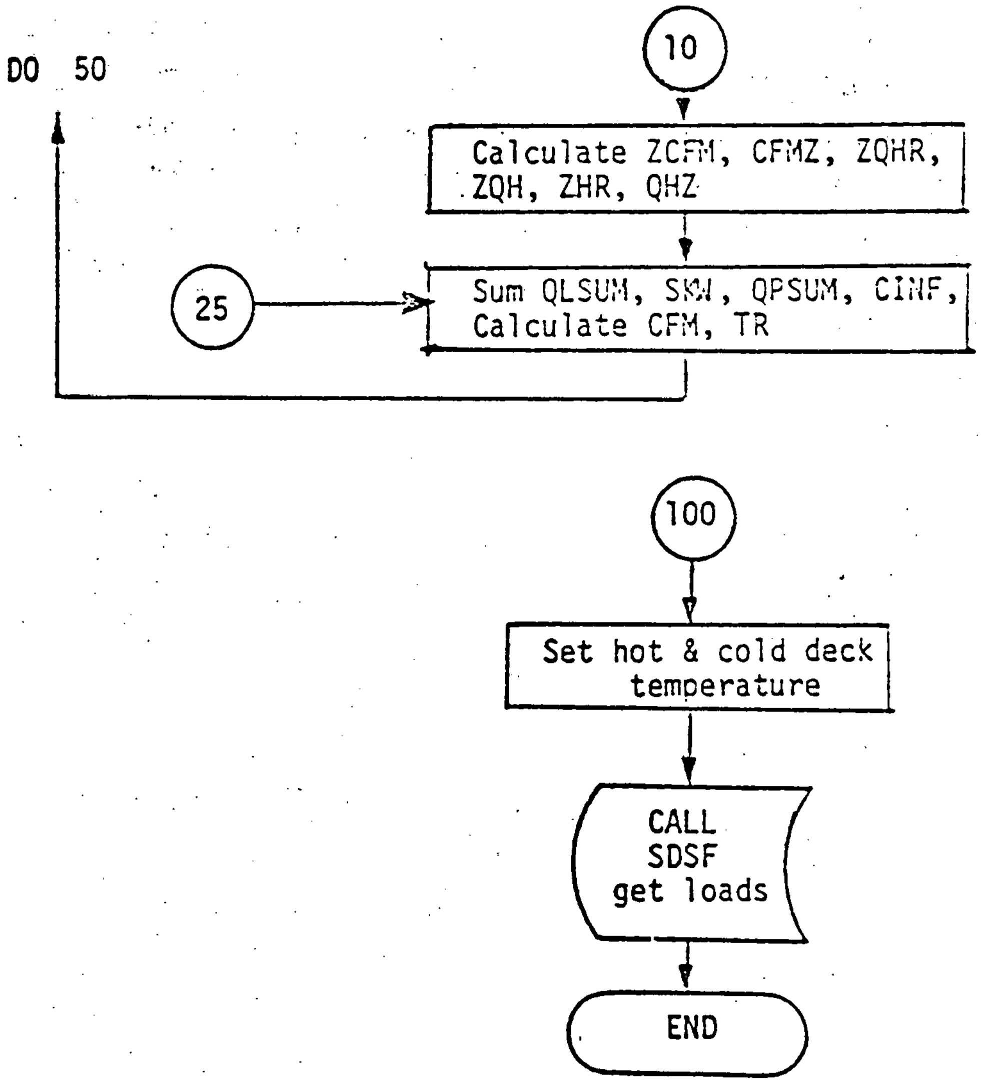
The routine enters a loop on all the zones serviced by this system. TEMDEV is called to get the adjusted load and temperature for each zone. A check is then made for system type. If system type is a single-zone fan system, the cold-deck temperature and the zone air flow are calculated and control passes to the load summing part of the loop. If system type is either variable-volume or constant-volume reheat, a check is made for fan operation. If the fan is off, control is passed to the load-summing part of the loop. If the fan is on, the CFMIN variable is calculated. CFMIN can vary from zero to one for the variable-volume system, but must be set equal to one for the constant-volume reheat system. A check is made for sensible loads. If there is a heat load, control passes to the zone air flow and load-calculation portion of the loop. If there is a cooling load, the zone air flow is calculated and a check is made for any necessary reheating. If reheating is required, the zone reheat load is calculated and added to the zone heat load. If no reheat is required, control passes to the load summing part of the loop. Results of this DO loop are the calculations of zone air flows, heating and reheating loads, sensible cooling loads, and the sums of latent, plenum, electrical, and infiltration loads.

After the zone DO loop is completed, the hot- and cold-deck temperatures are reset and subroutine SDSF is called. The primary air system loads are calculated and added to the zone loads to get total system loads.

c. VARVOL Functional Flow Chart.



VARVOL Functional Flow Chart - Continued


DO 50
10
Calculate ZCFM, CFMZ, ZQHR, ZQH, ZHR, QHZ
25
Sum QLSUM, SKW, QPSUM, CINF, Calculate CFM, TR
100
Set hot & cold deck temperature
CALL SDSF get loads
END

14. System No. 14 - Unitary Hydronic Heat Pump

a. General Description. The unitary hydronic heat pump consists of a compressor, an expander, evaporator coils, and condenser coils. In the heating mode, heat is pumped from the water circuit to the zone requiring heating. In the cooling mode, heat is pumped from the zone to the circulating water. The temperature of the circulated water is allowed to vary between the hot- and cold-deck temperatures. If the water temperature drops below the necessary hot-deck temperature, the boiler supplies needed heat input. If the water temperature rises above the cold-deck temperature, the excess heat is rejected through a cooling tower. The formulation of the calculational procedure is not complete now, and the coding has not been implemented. See Fig. IV.14.

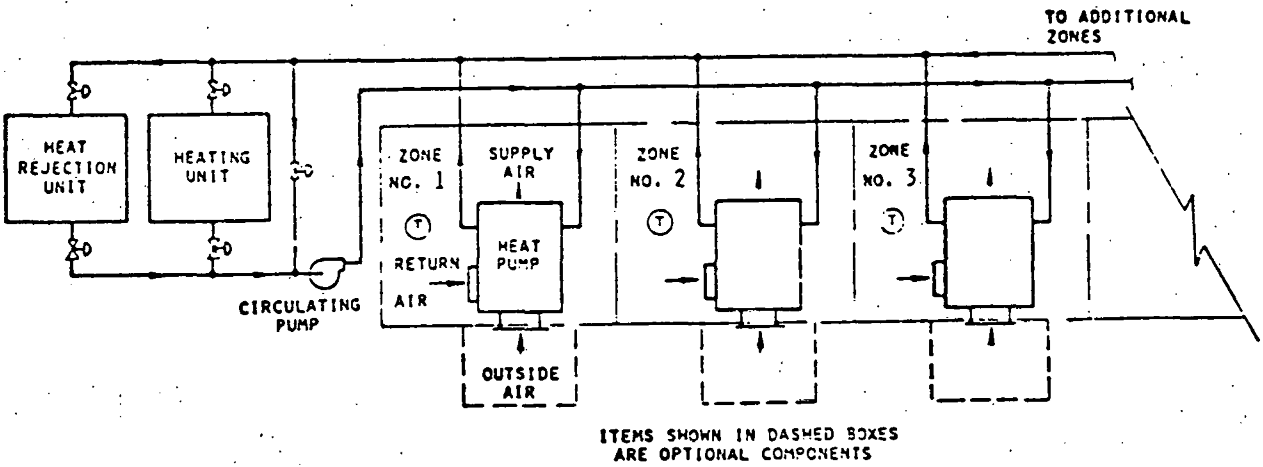


Fig. IV.14. System No. 14 - Unitary heat pump system.

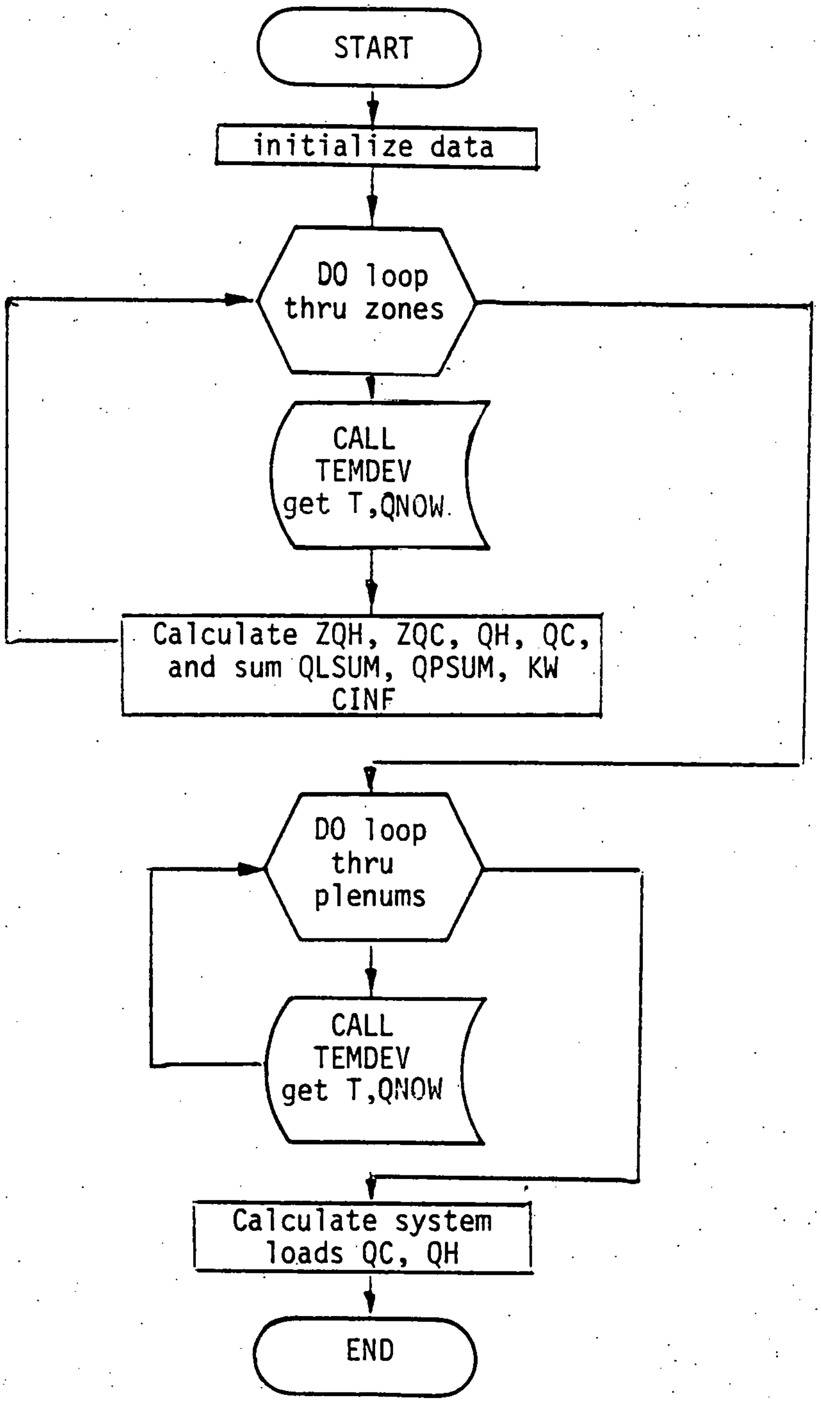
b. HTPUMP Calculational Sequence. Subroutine HTPUMP is called by the SYSTEMS Program as the driver subroutine for the unitary hydronic heat pump system simulation. Parameters and variables are initialized or set equal to zero.

The subroutine enters a DO loop on all the zones served by heat pumps. Subroutine TEMDEV is called to get the adjusted zone temperatures and loads. Results of this loop are the zone heating and cooling loads and sums of the latent, plenum, infiltration, electrical, and building heating and cooling loads.

After completion of the zone loop, the subroutine enters a plenum loop. Subroutine TEMDEV is called to get the adjusted plenum temperatures and loads. Results of this subroutine are the sums of building heating and cooling loads multiplied by the on/off heating and cooling system flags.

Subroutine HTPUMP is incomplete, but it is hoped that it will be complete for the October 1, 1977, version of the SYSTEMS Program.

c. HTPUMP Functional Flow Chart.



15. System No. 15 - Internal Variable-Volume/External Fan-Coil System

a. General Description. The internal variable-volume/external fan-coil system is composed of a centralized variable-volume system and individual fan-coil units in each zone. The variable-volume system consists of variable-volume supply and return fans, heating and cooling coils in series, face and bypass dampers, a humidifier, an economizer, and variable-air-volume terminal boxes in each zone. The heating coil and face and bypass dampers have not yet been implemented. The external fan-coil units consist of a blow-through fan, a water coil, and dampers to admit outside air.

The external fan-coil unit supplies a fraction of the zone load based on a design that allows the hot and chilled water temperatures to be reset. The internal variable-volume system supplies only cooling. See Fig. IV.15.

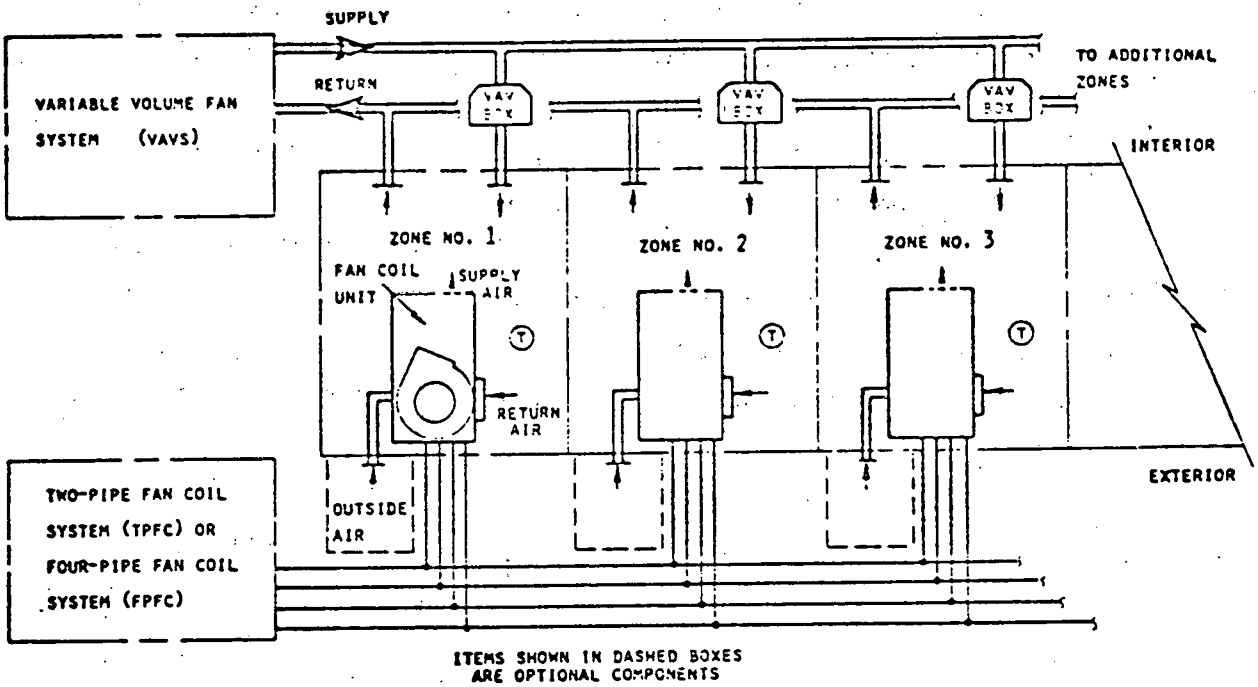

SUPPLY
RETURN
VARIABLE VOLUME FAN SYSTEM (VAVS)
VAV BOX
VAV BOX
VAV BOX
TO ADDITIONAL ZONES
INTERIOR
ZONE NO. 1
ZONE NO. 2
ZONE NO. 3
FAN COIL UNIT
SUPPLY AIR
T
T
T
RETURN AIR
OUTSIDE AIR
EXTERIOR
TWO-PIPE FAN COIL SYSTEM (TPFC) OR FOUR-PIPE FAN COIL SYSTEM (FPFC)
ITEMS SHOWN IN DASHED BOXES ARE OPTIONAL COMPONENTS

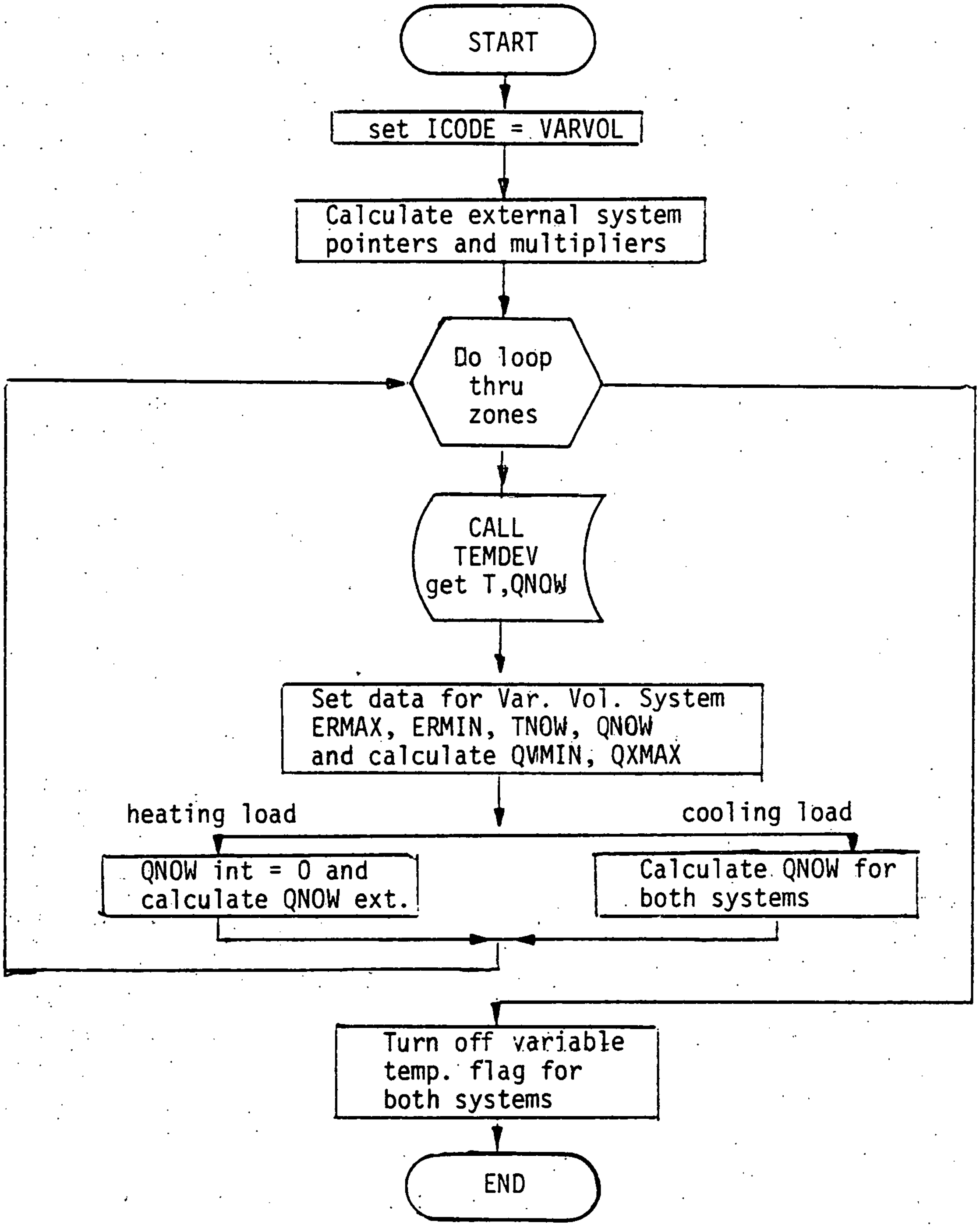
Fig. IV.15. System No. 15 - Internal variable-volume/external fan-coil system.

b. IVEFI Calculational Sequence. Subroutine IVEFI is called by the SYSTEMS Program as the load-splitting subroutine for the internal variable-volume/external fan-coil and the internal variable-volume/external induction system simulations. The system identification code is reset to the VARVOL driver subroutine. The pointers and multipliers are defined or calculated for both systems.

The subroutine enters a DO loop for all the zones served by these systems. Subroutine TEMDEV is called to get the zone adjusted temperatures and loads. The minimum and maximum energy extraction rates are calculated for the variable-volume system. The building load is then divided for the systems to operate at minimum load on the variable-volume system and maximum load on the external fan-coil or induction system. A check is then made for algebraic sign of the zone load. If the zone requires heating, the internal system load is set to zero, and the external system load is calculated. If the zone requires cooling, the load is split between the internal and external systems. Results of this loop are the distribution of the building heating and cooling loads between the internal and external systems.

After completion of the zone DO loop, the variable temperature flags are turned off for both systems. Control returns to the systems loop of the SYSTEMS Program. Subroutine VARVOL is called to simulate the operation of the internal variable-volume system. After this simulation, control returns to SYSTEMS. The index of the systems loop is incremented and either Subroutine FCOIL or INDUC is called to simulate operation of the external systems.

c.. IVEFI Functional Flow Chart.


START
set ICODE = VARVOL
Calculate external system pointers and multipliers
Do loop thru zones
CALL TEMDEV get T,QNOW
Set data for Var. Vol. System ERMAX, ERMIN, TNOW, QNOW and calculate QVMIN, QXMAX
heating load
cooling load
QNOW int = 0 and calculate QNOW ext.
Calculate QNOW for both systems
Turn off variable temp. flag for both systems
END

16. System No. 16 - Internal Variable-Volume/External Induction System

a. General Description. The internal variable-volume/external induction system is composed of a centralized variable-volume system and individual induction units in each zone. The variable-volume system consists of variable-volume supply and return fans, heating and cooling coils in series, face and bypass dampers, a humidifier, an economizer, and variable-air-volume (VAV) terminal boxes in each zone. The heating coil and the face and bypass dampers have not been implemented. The external induction units consist of a water coil, nozzles for the primary air flow, and a mixing chamber for primary and induced air.

The external induction unit supplies a fraction of the zone load based on a design that allows for the hot and chilled water temperatures to be reset. The internal variable-volume system supplies cooling only. See Fig. IV.16.

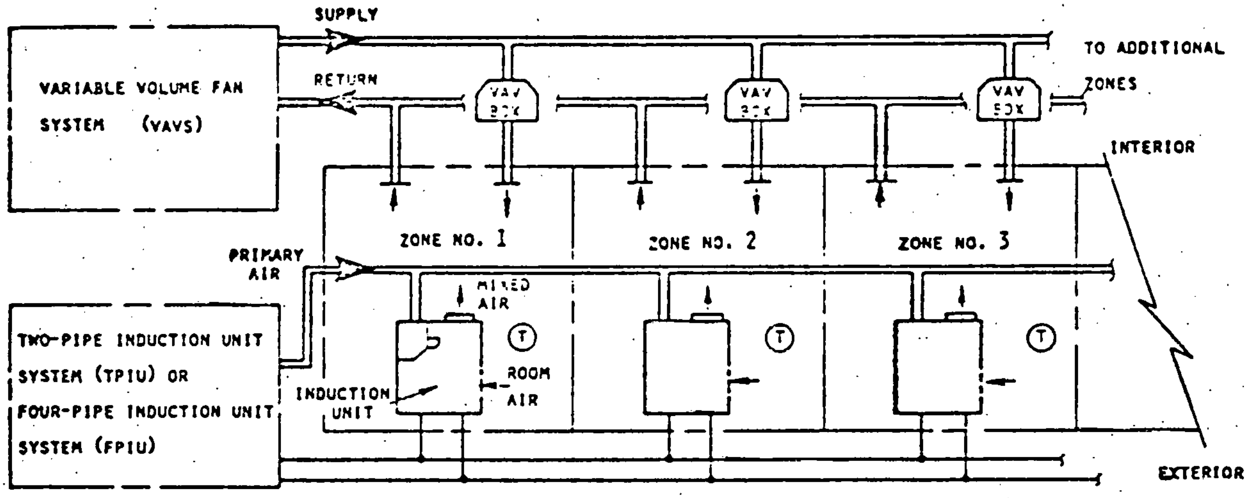


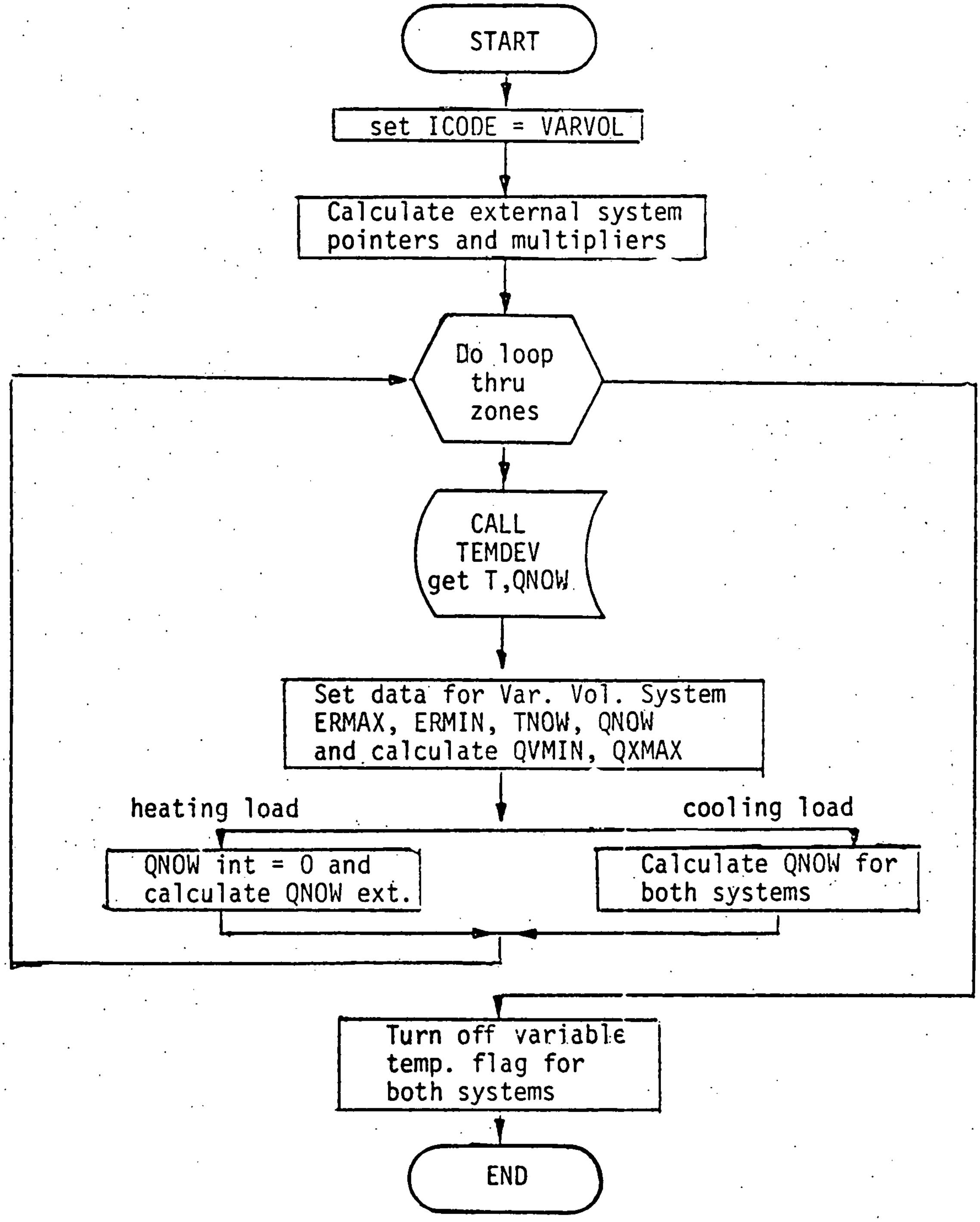
Fig. IV.16. System No. 16 - Internal variable-volume/external induction system.

b. IVEFI Calculational Sequence. Subroutine IVEFI is called by the SYSTEMS Program as the load-splitting subroutine for the internal variable-volume/external fan-coil and the internal variable-volume/external induction system simulations. The system identification code is reset to the VARVOL driver subroutine. The pointers and multipliers are defined or calculated for both systems.

The subroutine enters a DO loop for all the zones served by these systems. Subroutine TEMDEV is called to get the zone adjusted temperatures and loads. The minimum and maximum energy extraction rates are calculated for the variable-volume system. The building load is then divided for the systems to operate at minimum load on the variable-volume system and maximum load on the external fan-coil or induction system. A check is then made for algebraic sign of the zone load. If the zone requires heating, the internal system load is set to zero, and the external system load is calculated. If the zone requires cooling, the load is split between the internal and external systems. Results of this loop are the distribution of the building heating and cooling loads between the internal and external systems.

After completion of the zone DO loop, the variable temperature flags are turned off for both systems. Control returns to the systems loop of the SYSTEMS Program. Subroutine VARVOL is called to simulate the operation of the internal variable-volume system. After this simulation, control returns to SYSTEMS. The index of the systems loop is incremented and either Subroutine FCOIL or INDUC is called to simulate operation of the external systems.

c. IVEFI Functional Flow Chart.


START
set ICODE = VARVOL
Calculate external system pointers and multipliers
Do loop thru zones
CALL TEMDEV get T,QNOW
Set data for Var. Vol. System ERMAX, ERMIN, TNOW, QNOW and calculate QVMIN, QXMAX
heating load
cooling load
QNOW int = 0 and calculate QNOW ext.
Calculate QNOW for both systems
Turn off variable temp. flag for both systems
END

D. Program and Subroutine Description

The list of subroutines and descriptions is based on the May 5, 1977, version of the SYSTEMS Program. Subroutine flow charts are dated with the version of the listing that the flow chart is based on. Subroutines that changed as of the flow chart date will be updated to the SYSTEMS 1.1 version date of May 5, 1977.

1. List of Subroutines.

May 5, 1977

DDSF Calculates double-duct heating, cooling, and electrical loads

DDVACV Routine to calculate CFM in cold and hot ducts of a double-duct system

DESIGN Calculates design CFM, per cent outside air, TC, TH, etc., from input choices

DKTEMP Calculates hot- and cold-deck temperatures

DOUBLE Base subroutine for systems using two ducts

DST Determines daylight savings time flag

DUMP Dumps information about systems, zeros, plant numbers, and schedule days for debugging purposes

ECONO Calculates percentage of minimum outside air for three options of economizer cycle

FANPWR Calculates energy consumption of fan

FCOIL Calculates the system heating and cooling loads for a two-pipe or four-pipe fan-coil system

H Calculates enthalpy of moist air

HOLDAY Determines holidays during the year

HOURIN Gets hourly weather data and building loads data

HTPUMP Calculates the system heating and cooling loads for a heat pump system

INDUC Calculates the system heating and cooling loads for a two-pipe or a four-pipe induction system

IVEFI Splits system loads between an internal variable-air-volume system and an external fan-coil or induction system

PANEL Calculates the system heating and cooling loads for a floor-panel heating system

PPWVMS Calculates the partial pressure of water vapor in moisture-saturated air

REPORT Prints reports at end of month for QC, QH, and KW

RUN Initializes everything to start a new run; called before each run period

SCHED Updates pointer to beginning hours; called once daily

SDSF Calculates single-duct system heating and cooling loads

SETUP Reads and creates data structure blocks; will be part of SDL

SUM Accumulates the sum of heating, cooling, electrical, plenum, latent, and infiltration loads over the zones serviced by a system

SYSTEMS Main program

SZVV Calculates the zone and system temperatures, flow rates, and loads for a single-duct fan system with subzone induction mixing boxes

TEMDEV Calculates adjusted temperatures and loads in spaces that allow space temperature to float in a throttling range

UNITHV Calculates heating loads for unit heater and/or unit ventilator for several zones

V Calculates the specific volume of moist air

VARVOL Base subroutine for single-duct systems with variable-volume and reheat capability

VREP1* Writes reports, plants and systems information

VREP2* Writes report, internal heat transfer information

WDREAD Reads weather data file for given day and returns current hourly weather data

ZERO Zeros out sums of running totals for new time periods

* Will not appear in later versions of SYSTEMS.

2. Nomenclature - Definition of Variables.

ITS	First word of schedule block
IZ	First word of zone block
IS	First word of system block
IP	First word of plant block
NTS	Number of schedules
NZONE	Number of zones
NSYS	Number of systems
NPLANT	Number of plants
ISCN	First schedule name
IZN	First zone name
ISN	First system name
ISPN	First space name
IPS	First plant name
IZI	Number of NCZD items from standard file - ZP2
ISS	Number of NSS items from standard file NSP
ISZ	Number of NZD items from standard file - ZP1
IA & AA	Main data structure
INAME (1,-1)	First word of name
INAME (2,-1)	Second word of name
IM (12)	Days in each month
IHR	Current hour
IDAY	Current day
IMO	Current month
IYR	Current year
LDAY	End day
LMO	End month
LYR	End year
IRUN	Number of run-period
CRATF (1-3, construction type)	G_0, G_1, and G_2
CRATF (4-5, construction type)	P_0, P_1

ZP1	Pointer-to-current zone NZD
ZP2	Pointer-to-current zone NCZD
NSP	Pointer-to-current system NSS
ILEAPF	0 - not a leap year 1 - is a leap year
IDSTF	0 - not daylight savings time 1 - is daylight savings time
IEFLAG	End-of-run-period flag
IDOY	Current day of year
IDOW	Current day of week 1-7 Sun-Sat
SCHR	Schedule hour = IHR-IDSTF
ISCDAY	Schedule day = 1-8 Sun-Sat, Holiday
ISTDF	Standard file
ILOADF	Loads file
IWEATH	Weather file
IREPF	Report output file
IPLANTF	Plant output file
ISDNF	Design file
IMON (14)	Names of months, 13-blank, 14-"TOTAL"
BUF & IBUF	Input buffer for loads file data
ISZ	Pointer-to-initial system NZD
NSZ	Number of zones in system
P1, P2	Miscellaneous used pointers
TH	Hot-deck temperature
TC	Cold-deck temperature
QH	Total QH
QC	Total QC
CFMH	Hot CFM
CFMC	Cold CFM
CFMAX	Maximum design CFM
CFM	CFMH + CFMC
QLSUM	Sum of latent Q's
QPSUM	Sum of plenum Q's

QESUM	Sum of equipment Q's
ICON	Cooling
IHON	Heating on
ION	System 0 - off 1 - on
CONS (1)-1.08	Conversion Q → CFM
(2)-4790	lb/hr → Btu/hr
(3)-.372	
(4) (5) 0.,0.	

3. Variable Lists

a. Plant Specific Data.

Indexed Variable Location	Description	Equivalent Variable Name	Default Values
	Pointer to beginning of plant data block	NPL	0
IA(NPL)	Number of systems served by plant	NPLSYS	1
IA(NPL+1)	Pointer to pointer to first system	IPLSYS	1
AA(NPL+2)	Hourly plant total QC	QCPL	1
AA(NPL+3)	Hourly plant total QH	QHPL	1
AA(NPL+4)	Hourly plant total kW	PKW	1
AA(NPL+5)	Sum of QC over run period	QCPLYR	1
AA(NPL+6)	Sum of QH over the year	QHPLYR	1
AA(NPL+7)	Sum of kW over the year	PKWYR	1
AA(NPL+8)	Maximum QC in run period	QCPLM	1
AA(NPL+9)	Maximum QH in run period	QHPLM	1
AA(NPL+10)	Maximum kW in run period	PKWM	1

b. System Specific Data.

Indexed Variable Location	Description	Equivalent Variable Name	Default Values
	Pointer to first item of system data	NSP	0
IA(NSP)	Number of zones on this system	NZONES	1
IA(NSP+1)	The type of the system	ITYPE	1
IA(NSP+2)	Pointer to first zone data block (NZD)	ISZONES	1
AA(NSP+3)	This hour QC	QC	1
AA(NSP+4)	This hour QH	QH	1
AA(NSP+5)	This hour kW	SKW	1
AA(NSP+6)	Sum of QC for month	QCMO	1
AA(NSP+7)	Sum of QH for month	QHMO	1
AA(NSP+8)	Sum of kW for month	SKWMO	1
AA(NSP+9)	Maximum QC in month	QCMOM	1
AA(NSP+10)	Maximum QH in month	QHMOM	1
AA(NSP+11)	Maximum kW in month	SKWMOM	1
AA(NSP+12)	Sum of QC for run period	QCYR	1
AA(NSP+13)	Sum of QH for run period	QHYR	1
AA(NSP+14)	Sum of kW for run period	SKWYR	1
AA(NSP+15)	Maximum QC in run period	QCYRM	1
AA(NSP+16)	Maximum QH in run period	QHYRM	1
AA(NSP+17)	Maximum kW in run period	SKWYRM	1
IA(NSP+18)	Day and hour of QC maximum in month	IQCDAYHR	1
IA(NSP+19)	Day and hour of QH maximum in month	IQHDAYHR	1
AA(NSP+20)	Design system fan CFM	SYSCFM	1

Indexed Variable Location	Description	Equivalent Variable Name	Default Values
AA(NSP+21)	Design hot-deck temperature	HDTEMP	1
AA(NSP+22)	Design cold-deck temperature	CDTEMP	1
AA(NSP+23)	Minimum CFM ratio system wide	SYSCFMIN	1
AA(NSP+24)	Outside air minimum ratio system wide	OUTMIN	1
AA(NSP+25)	Supply fan static pressure	SUPSTAT	1
AA(NSP+25)	Temperature change across supply fan	SUPPELTA	0 -1
AA(NSP+26)	Supply fan efficiency	SUPEFF	1
AA(NSP+26)	Supply + return full load kW	FULLKW	0 -1
AA(NSP+27)	Return fan static pressure	RETSTAT	1
AA(NSP+27)	Return fan temperature change	REIDELTA	0 -1
AA(NSP+28)	Return fan efficiency	RETEFF	1
AA(NSP+29)	Maximum return humidity	HUMAX	1
AA(NSP+30)	Minimum return humidity	HUMIN	1
AA(NSP+31)	Cool coil leaving humidity	HUMCOIL	1
IA(NSP+32)	Type of economizer	IECONO	1
IA(NSP+33)	Outside air schedule	IOUT	1
IA(NSP+34)	Heating schedule	IHON	1
IA(NSP+35)	Cooling schedule	ICON	1
IA(NSP+36)	Fan schedule	IFON	1
AA(NSP+37)	Reheat coil temperature difference	REHEAT	1
IA(NSP+38)	Type of fan control	IFANCTRL	1
IA(NSP+39)	Return fan control	IRETCTRL	1
IA(NSP+40)	Heating reset schedule	IHRESET	1
IA(NSP+41)	Cooling reset schedule	ICRESET	1
IA(NSP+42)	Baseboard reset schedule	IBRESET	1
IA(NSP+43)	Heating temperature control method	IHCTRL	1
IA(NSP+44)	Cooling temperature control method	ICCTRL	1
AA(NSP+45)	Induced air ratio	RINDUC	1

Indexed Variable Location	Description	Equivalent Variable Name	Default Values
AA(NSP+46)	Changeover temperature	COVERT	1
AA(NSP+47)	Water volume	WATVOL	1
IA(NSP+48)	Variable temperature on/off	IVTEMP	1
AA(NSP+49)	Safety factor	SFACTOR	1
IA(NSP+50)	Plenum No. 1 pointer	IPLEN1	1
IA(NSP+51)	Plenum No. 2 pointer	IPLEN2	1
IA(NSP+52)	Plenum No. 3 pointer	IPLEN3	1
AA(NSP+53)	Plenum summed multiplier	PLENMULT	1
AA(NSP+54)	Last hour mix temperature	PASTMIX	1
IA(NSP+55)	Last hour mode heating/ cooling	IMODEL	1
IA(NSP+56)	Return air path	IPATH	1

c. System Specific Zone Data.

Indexed Variable Location	Description	Equivalent Variable Name	Default Values
	Pointer to system specific zone data	ZP1	0
IA(ZP1)	Value of ZP2	IZP2	1
AA(ZP1+1)	This hour temperature of zone	TNOW	1
AA(ZP1+2)	Last hour temperature of zone	TPAST	1
AA(ZP1+3)	Maximum heating	ERMIN	1
AA(ZP1+4)	This hour heat extraction	QNOW	1
AA(ZP1+5)	Minimum outside air ventilation	VENTMIN	1
AA(ZP1+6)	Zone design CFM	CFMAX	1
AA(ZP1+7)	Maximum cooling	ERMAX	1
AA(ZP1+8)	This hour's temperature setting	TSET	1
AA(ZP1+9)	Last hour infiltration	VIPAST	1
IA(ZP1+10)	Cooling temperature schedule	ICSCHED	1
AA(ZP1+11)	Throttling range	THRANGE	1
IA(ZP1+12)	Heating schedule	IHSCHED	
AA(ZP1+13)	Heating minimum	ERMINM	1
AA(ZP1+14)	Cooling minimum	ERMAXM	1

d. Constant Zone Data.

Indexed Variable Location	Description	Equivalent Variable Name	Default Values
	Pointer to zone specific zone data	ZP2	0
IA(ZP2-1)	Zone space number	NZSPACE	0 -1
IA(ZP2)	Buffer address in loads buffer of space	IBUFFP	1
AA(ZP2+1)	Sensible load	QS	1
AA(ZP2+2)	Latent load	QL	1
AA(ZP2+3)	Electrical load	ZKW	1
AA(ZP2+4)	Plenum load	QP	1
AA(ZP2+5)	Equipment load	QE	1
AA(ZP2+6)	Internal transfer load	QI	1
AA(ZP2+7)	CFM of outside air infiltration	CFMINF	1
IA(ZP2+8)	Components of load	ICOMP	1
AA(ZP2+8)	Maximum people in space	PEOPLE	0 -1
AA(ZP2+9)	Conductance of space	CONDUC	1
AA(ZP2+10)	QS maximum loads or SDL	QMAX	1
AA(ZP2+11)	QS minimum loads or SDL	QMIN	1
IA(ZP2+12)	Construction type	ICONST	1
AA(ZP2+13)	Area of space	AREA	1
AA(ZP2+14)	Load calculation temperature	TLOADS	1
AA(ZP2+15)	Last hour QS	QSPAST	1
AA(ZP2+16)	Volume of space	VOLUME	1
AA(ZP2+17)	Throttling range	THROTT	1
AA(ZP2+18)	Minimum ventilation	VENTMIN	1
IA(ZP2+19)	Zone cooling schedule	ICOOL	1
AA(ZP2+20)	Design CFM	DCFM	1
AA(ZP2+21)	Minimum air changes	AIRCH	1
AA(ZP2+22)	CFM per ft^2	CFMSQFT	1

Indexed Variable Location	Description	Equivalent Variable Name	Default Values
AA(ZP2+23)	Outside air changes	OUTCH	1
AA(ZP2+24)	Outside air person	OUTPER	1
IA(ZP2+25)	Zone heating schedule	IHEAT	1
AA(ZP2+26)	Zone multiplier	ZMULTR	1
AA(ZP2+27)	Design heating temperature	DHTEMP	1
AA(ZP2+28)	Design cooling temperature	DCTEMP	1

e. Schedule Data.

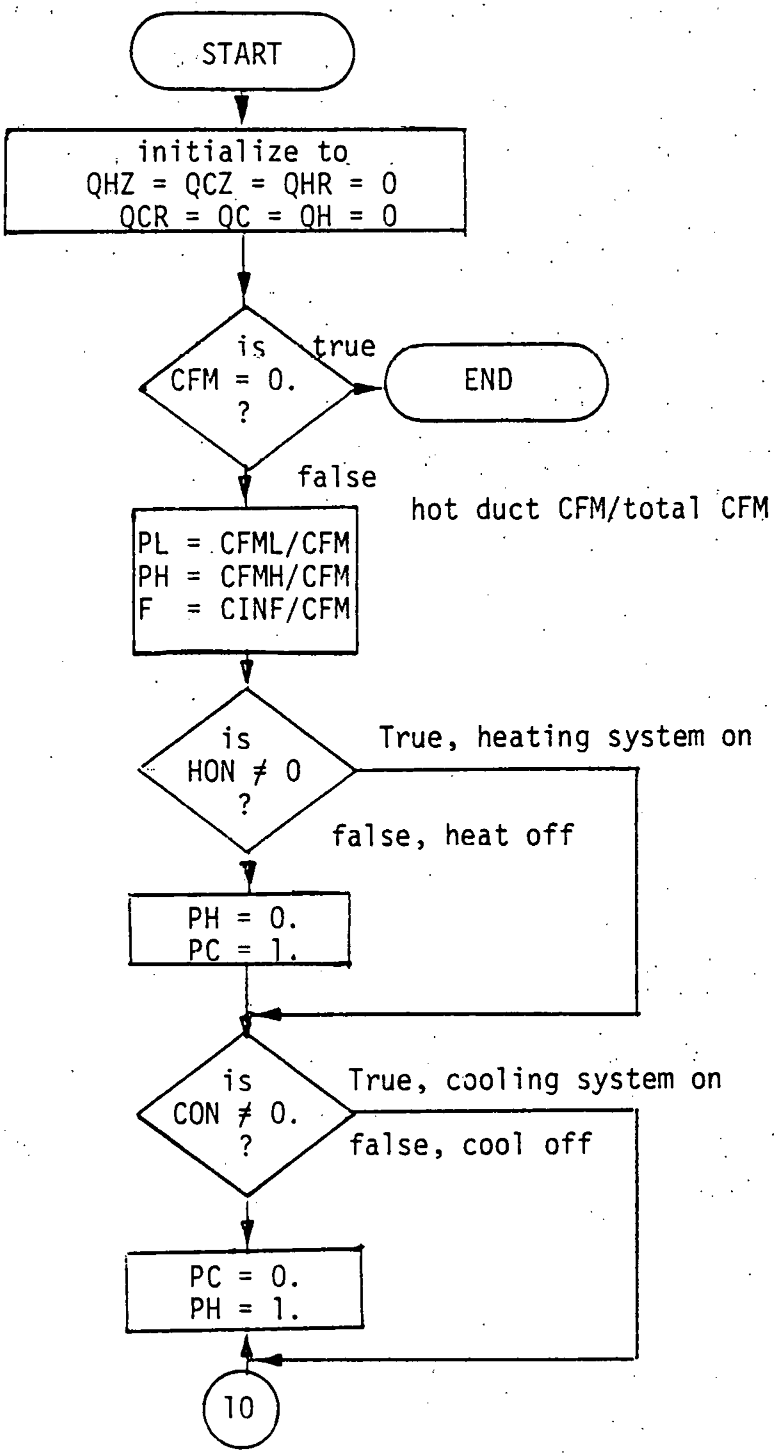
Indexed Variable Location	Description	Equivalent Variable Name	Default Values
	Pointer to beginning of schedule table	NSC	0
IA(NSC)	Number of weeks in this schedule	NWEEKS	1
IA(NSC+1)	Number of current week	NCWEEK	1
IA(NSC+2)	Pointer to start of first week block	IWEEKS	1
IA(NSC+3)	Pointer to start of current day (-1)	IDAY	1
	Pointer to start week block	NWB	0
IA(NWB)	Ending MO*100 + day for this week	NMODAY	1
IA(NWB+1)	Days of this week	NDAYS	8
IA(NWB+1)		ISUN	0 -8
IA(NWB+2)		IMON	0 -7
IA(NWB+3)		ITUE	0 -6
IA(NWB+4)		IWED	0 -5
IA(NWB+5)		ITHU	0 -4
IA(NWB+6)		IFRI	0 -3
IA(NWB+7)		ISAT	0 -2
IA(NWB+8)		IHOL	0 -1

4. Description by Subroutine

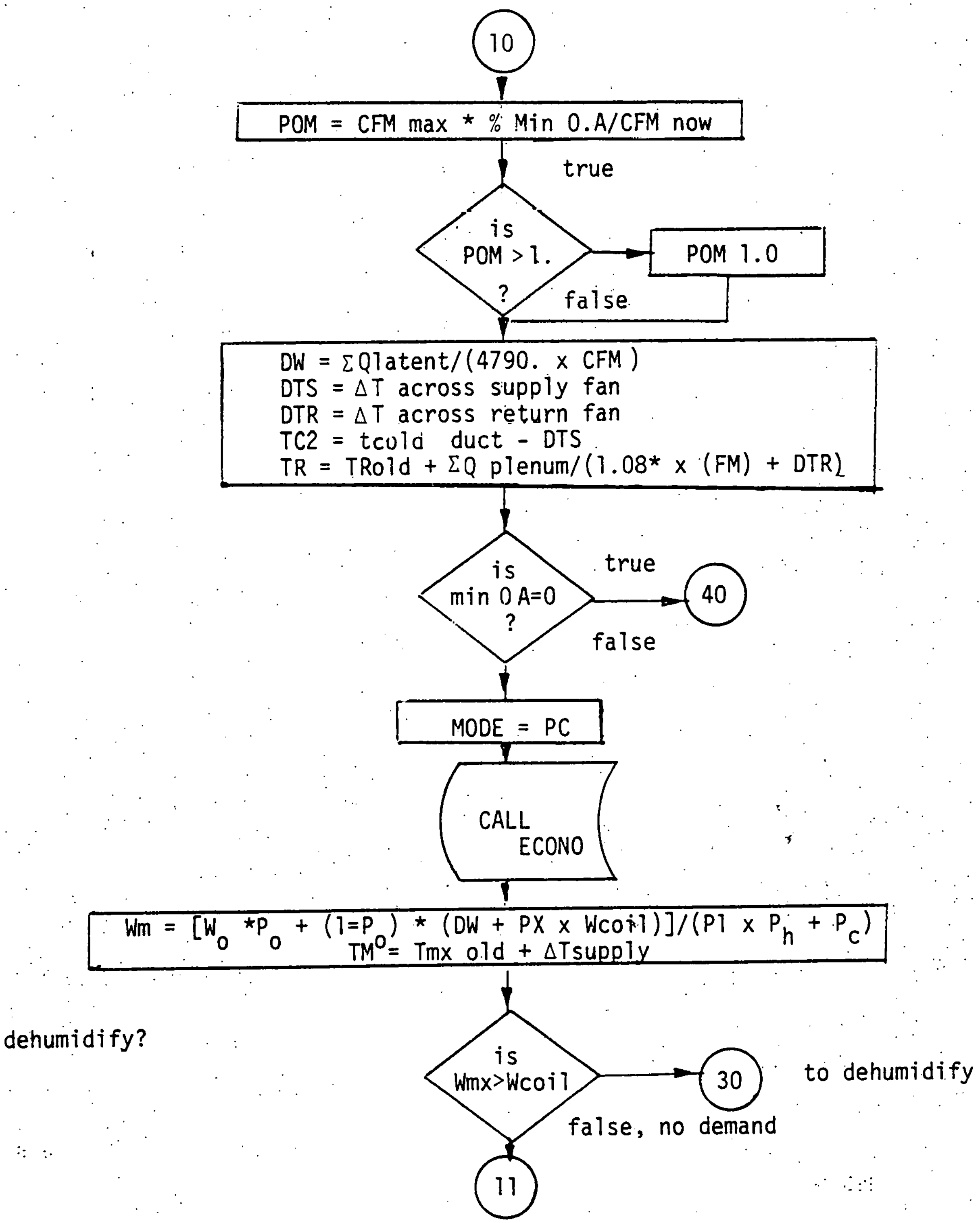
The following pages contain descriptions and flow charts for some of the subroutines in the Cal-ERDA SYSTEMS Program. The programs flow-charted thus far include the psychrometric calculations, the variable temperature routine, the economizer and deck temperature routines, the single- and double-duct main routines, the routines that calculate system flow rates, system heating and cooling loads, energy consumed by system fans, and the routine that sums zone loads for each system. The programs requiring flow charts are the heat pump, fan coil units, induction system, panel heating, IVEFI, and the data manipulation routines.

a. Subroutine DDSF.

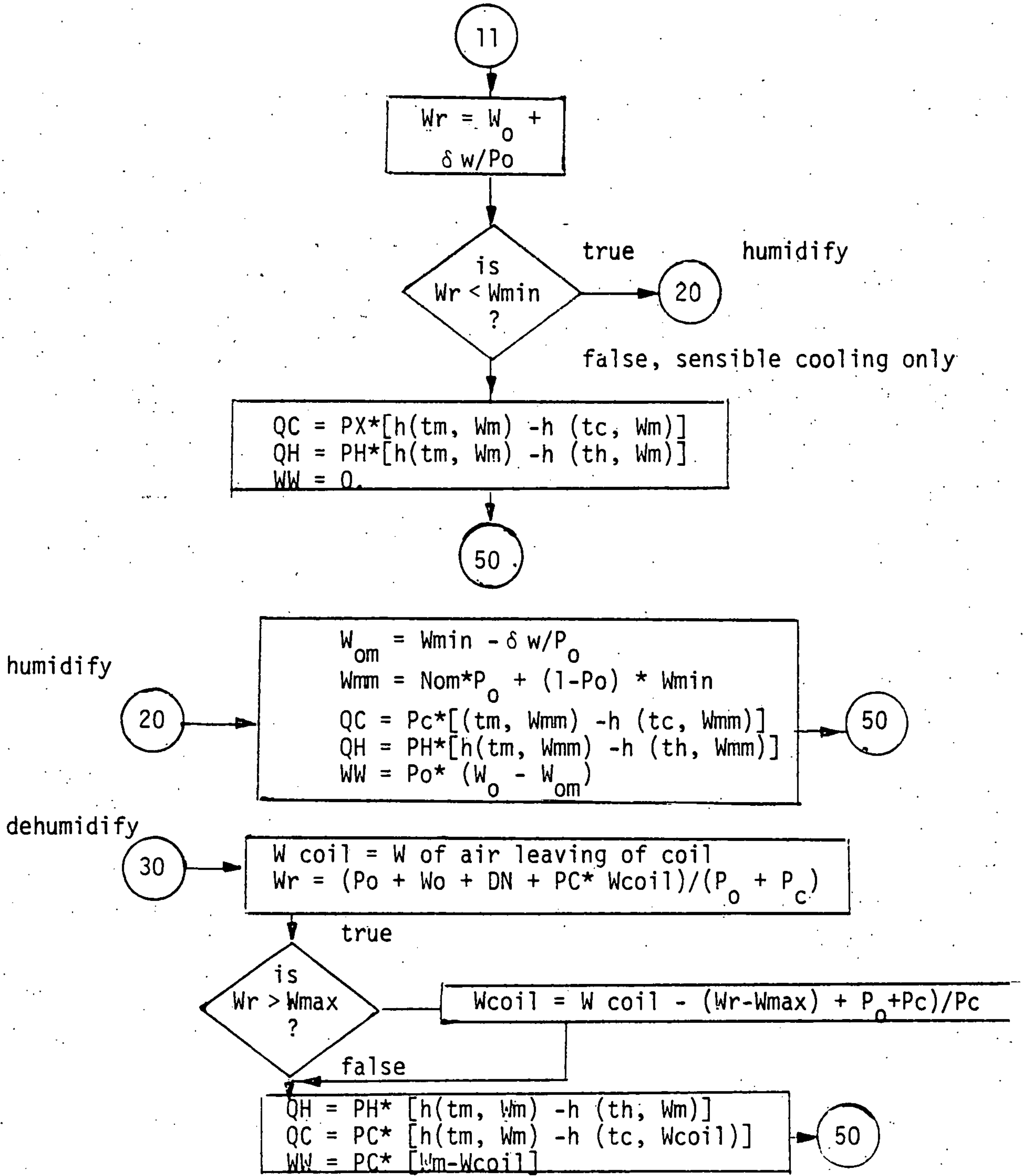
May 5, 1977



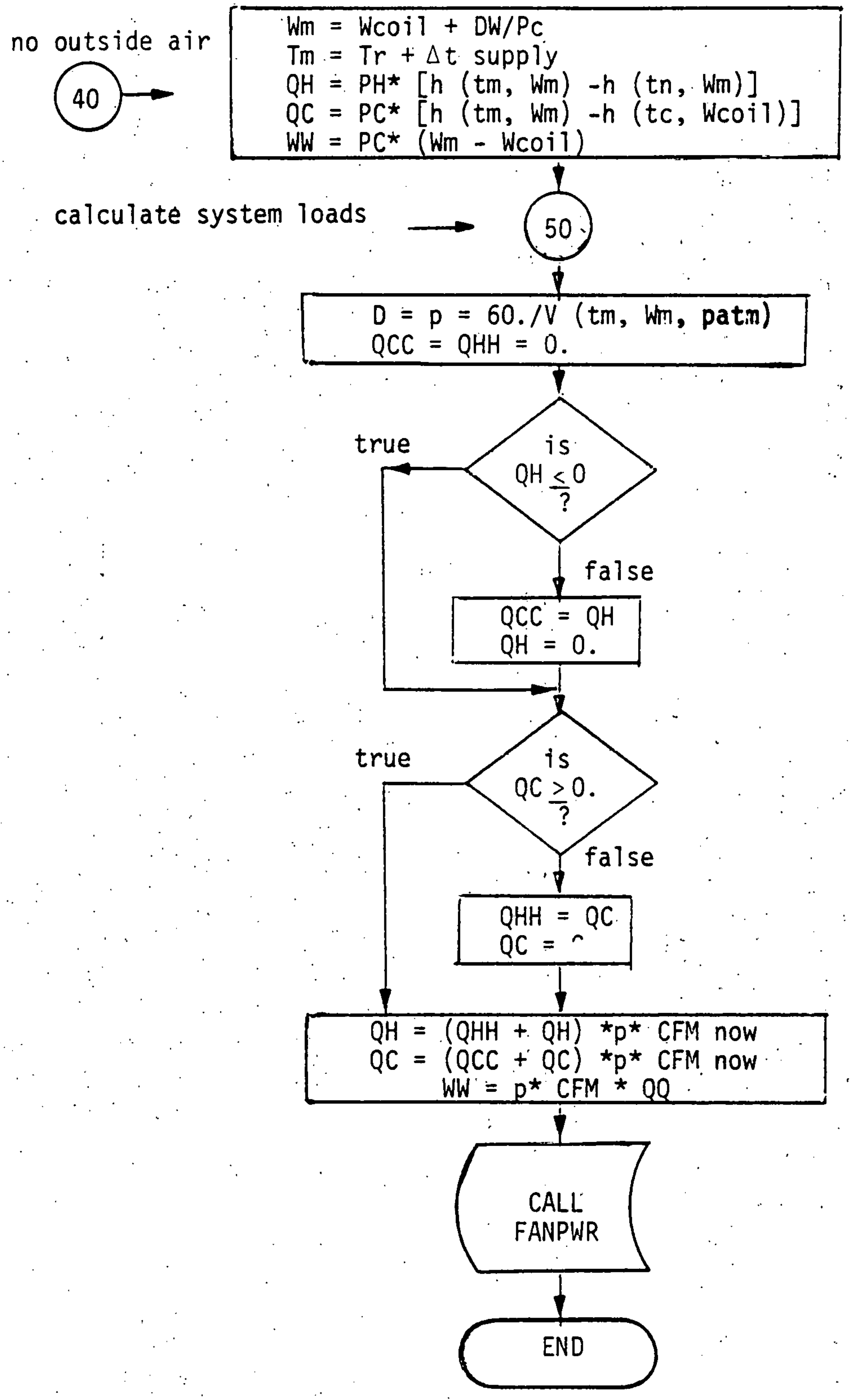
Subroutine DDSF - Continued


10
POM = CFM max * % Min 0.A/CFM now
true
is
POM > 1.
?
POM 1.0
false
DW = ΣQlatent/(4790. x CFM)
DTS = ΔT across supply fan
DTR = ΔT across return fan
TC2 = tcold duct - DTS
TR = TRold + ΣQ plenum/(1.08* x (FM) + DTR)
is
min 0 A=0
?
true
40
false
MODE = PC
CALL
ECONO
Wm = [W_o *P_o + (1=P_o) * (DW + PX x Wcoil)]/(P1 x P_h + P_c)
TM^o= Tmx old + ΔTsupply
dehumidify?
is
Wmx>Wcoil
30
to dehumidify
false, no demand
11

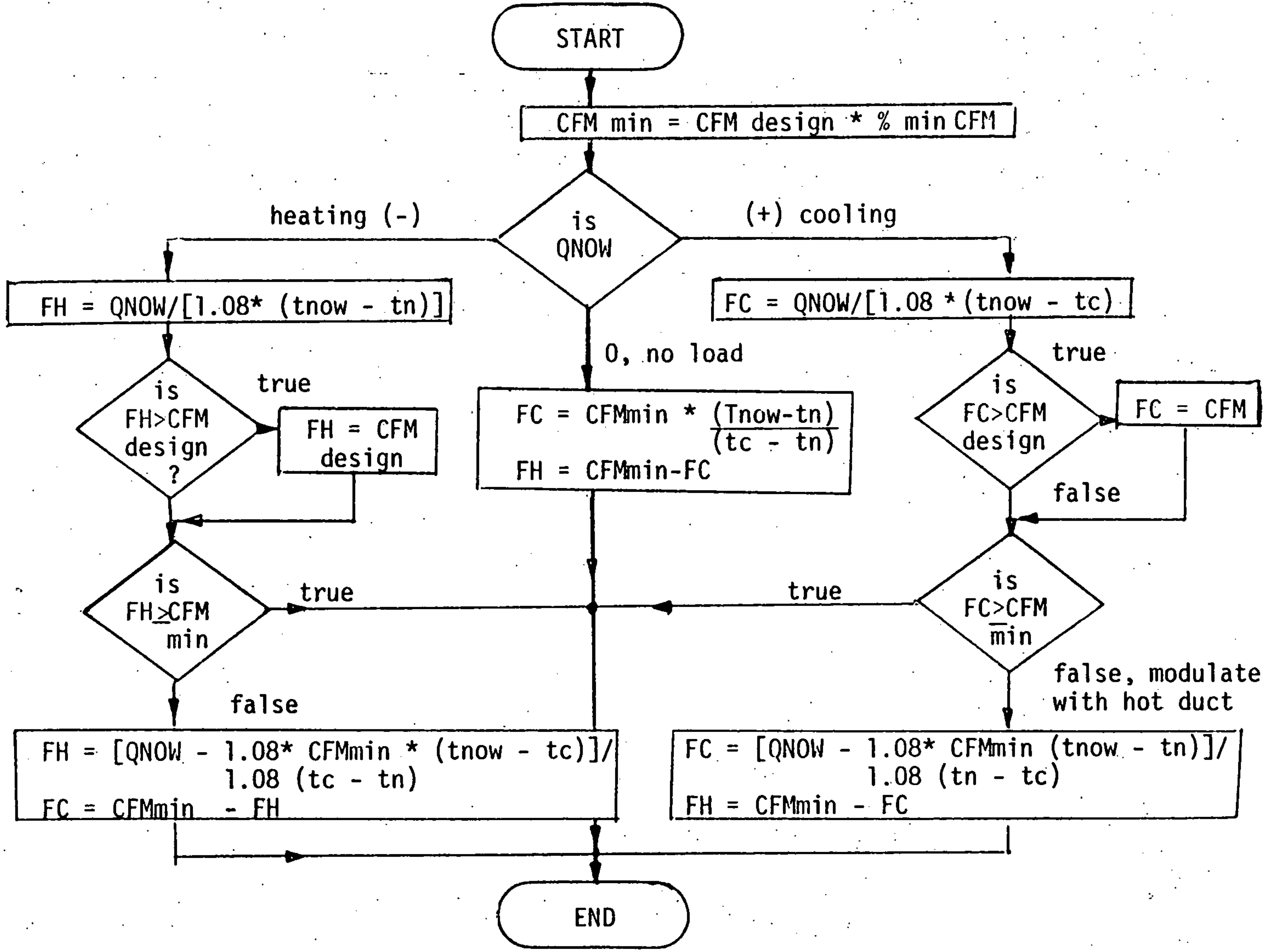
Subroutine DDSF - Continued


11
Wr = Wo + δ w/Po
is Wr < Wmin ?
true
humidify
20
false, sensible cooling only
QC = PX*[h(tm, Wm) -h (tc, Wm)]
QH = PH*[h(tm, Wm) -h (th, Wm)]
WW = 0.
50
humidify
20
Wom = Wmin - δ w/Po
Wmm = Nom*Po + (1-Po) * Wmin
QC = Pc*[(tm, Wmm) -h (tc, Wmm)]
QH = PH*[h(tm, Wmm) -h (th, Wmm)]
WW = Po* (Wo - Wom)
50
dehumidify
30
W coil = W of air leaving of coil
Wr = (Po + Wo + DN + PC* Wcoil)/(Po + Pc)
true
is Wr > Wmax ?
Wcoil = W coil - (Wr-Wmax) + Po+Pc)/Pc
false
QH = PH* [h(tm, Wm) -h (th, Wm)]
QC = PC* [h(tm, Wm) -h (tc, Wcoil)]
WW = PC* [Wm-Wcoil]
50

Subroutine DDSF - Continued

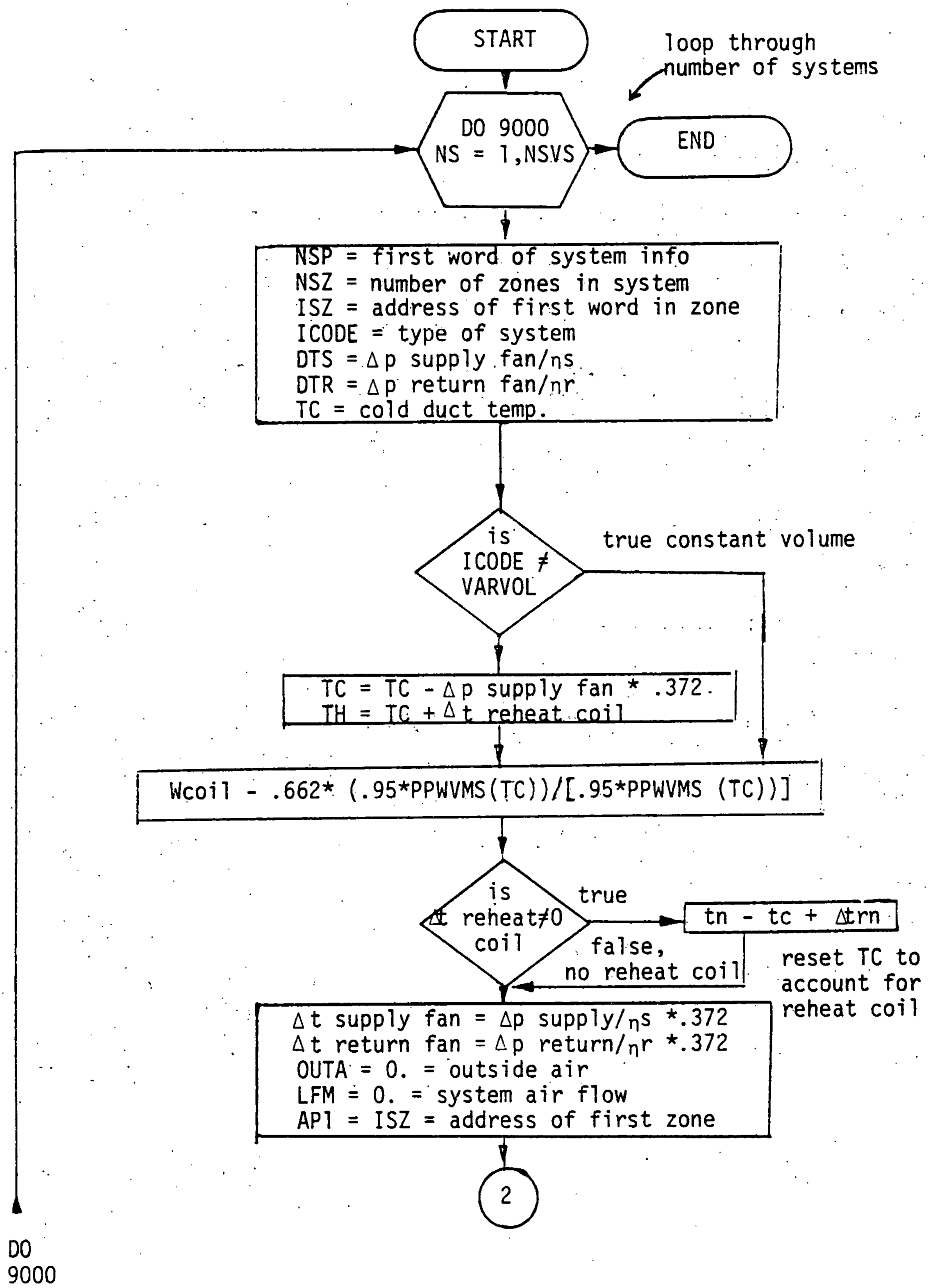


b. Subroutine DDVACV.

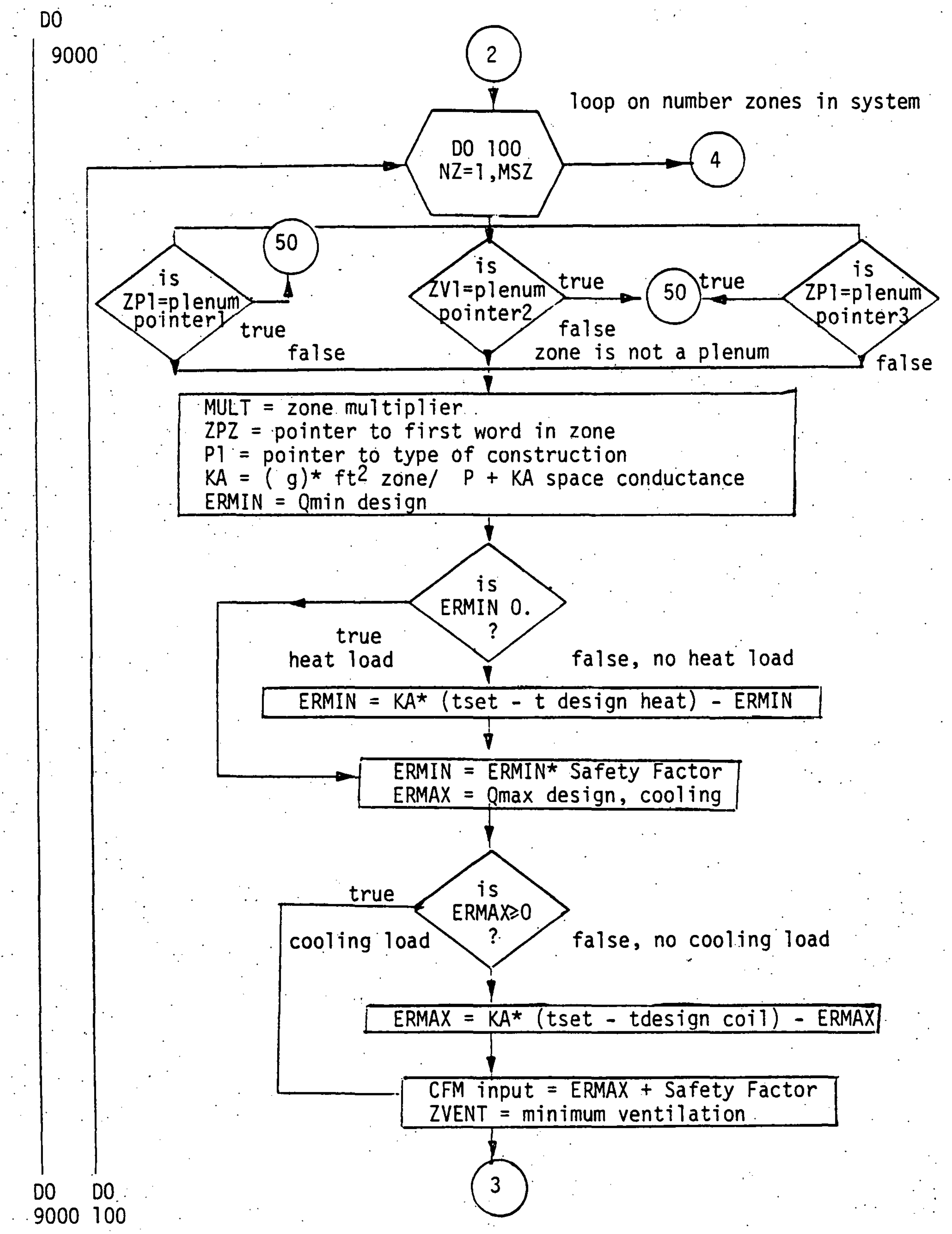

START
CFM min = CFM design * % min CFM
is QNOW
heating (-)
(+) cooling
0, no load
FH = QNOW/[1.08* (tnow - tn)]
FC = QNOW/[1.08 * (tnow - tc)
is FH>CFM design ?
true
FH = CFM design
is FC>CFM design
true
FC = CFM
false
FC = CFMmin * (Tnow-tn)/(tc - tn)
FH = CFMmin-FC
is FH≥CFM min
true
is FC>CFM min
true
false
false, modulate with hot duct
FH = [QNOW - 1.08* CFMmin * (tnow - tc)]/ 1.08 (tc - tn)
FC = CFMmin - FH
FC = [QNOW - 1.08* CFMmin (tnow - tn)]/ 1.08 (tn - tc)
FH = CFMmin - FC
END

c. Subroutine DESIGN.

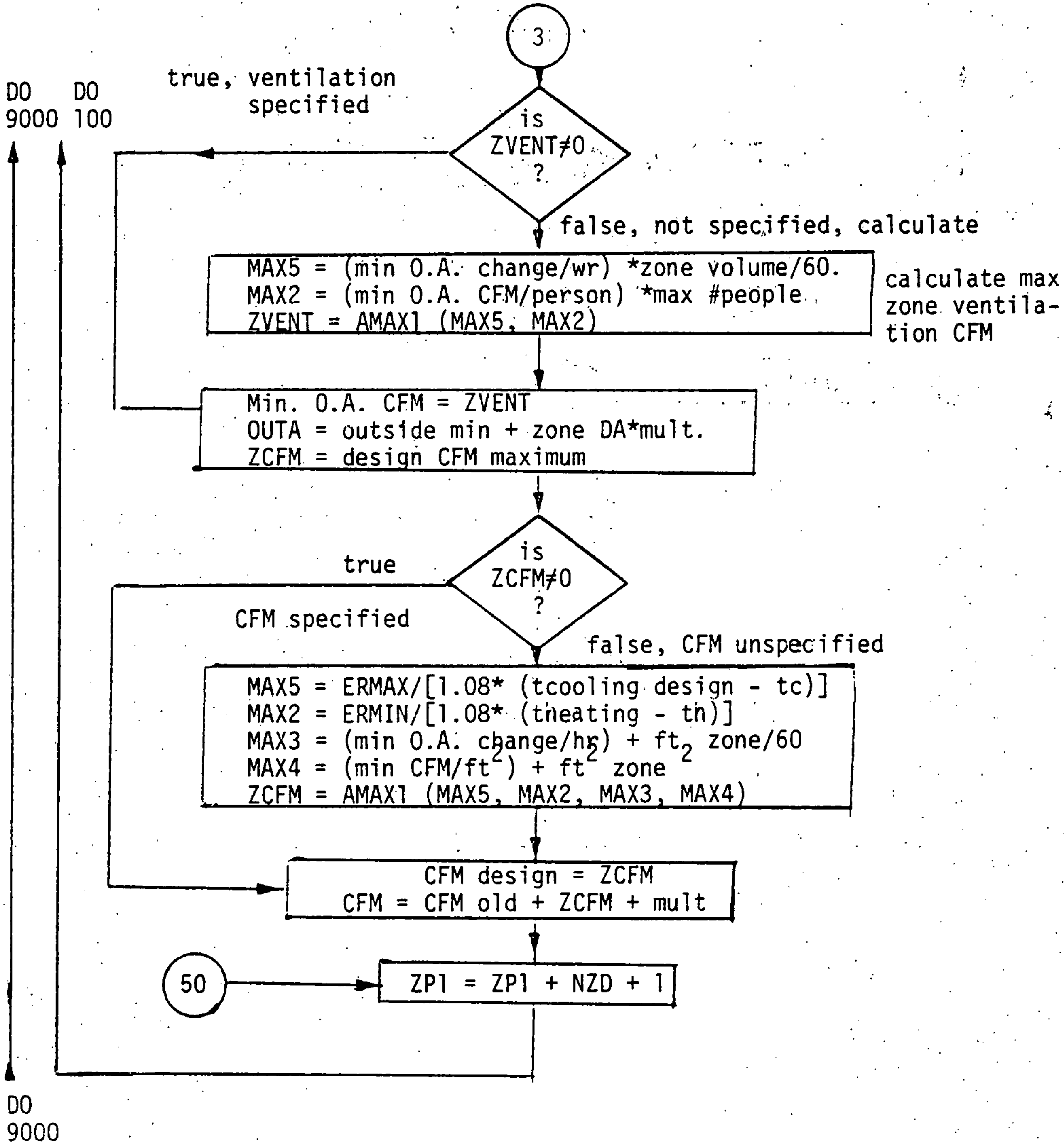
May 5, 1977


START
loop through number of systems
DO 9000 NS = 1,NSVS
END
NSP = first word of system info
NSZ = number of zones in system
ISZ = address of first word in zone
ICODE = type of system
DTS = Δp supply fan/ηs
DTR = Δp return fan/ηr
TC = cold duct temp.
is ICODE ≠ VARVOL
true constant volume
TC = TC - Δp supply fan * .372
TH = TC + Δt reheat coil
Wcoil - .662* (.95*PPWVMS(TC))/[.95*PPWVMS (TC))]
is Δt reheat≠0 coil
true
tn - tc + Δtrn
reset TC to account for reheat coil
false, no reheat coil
Δt supply fan = Δp supply/ηs *.372
Δt return fan = Δp return/ηr *.372
OUTA = 0. = outside air
LFM = 0. = system air flow
AP1 = ISZ = address of first zone
2
DO 9000

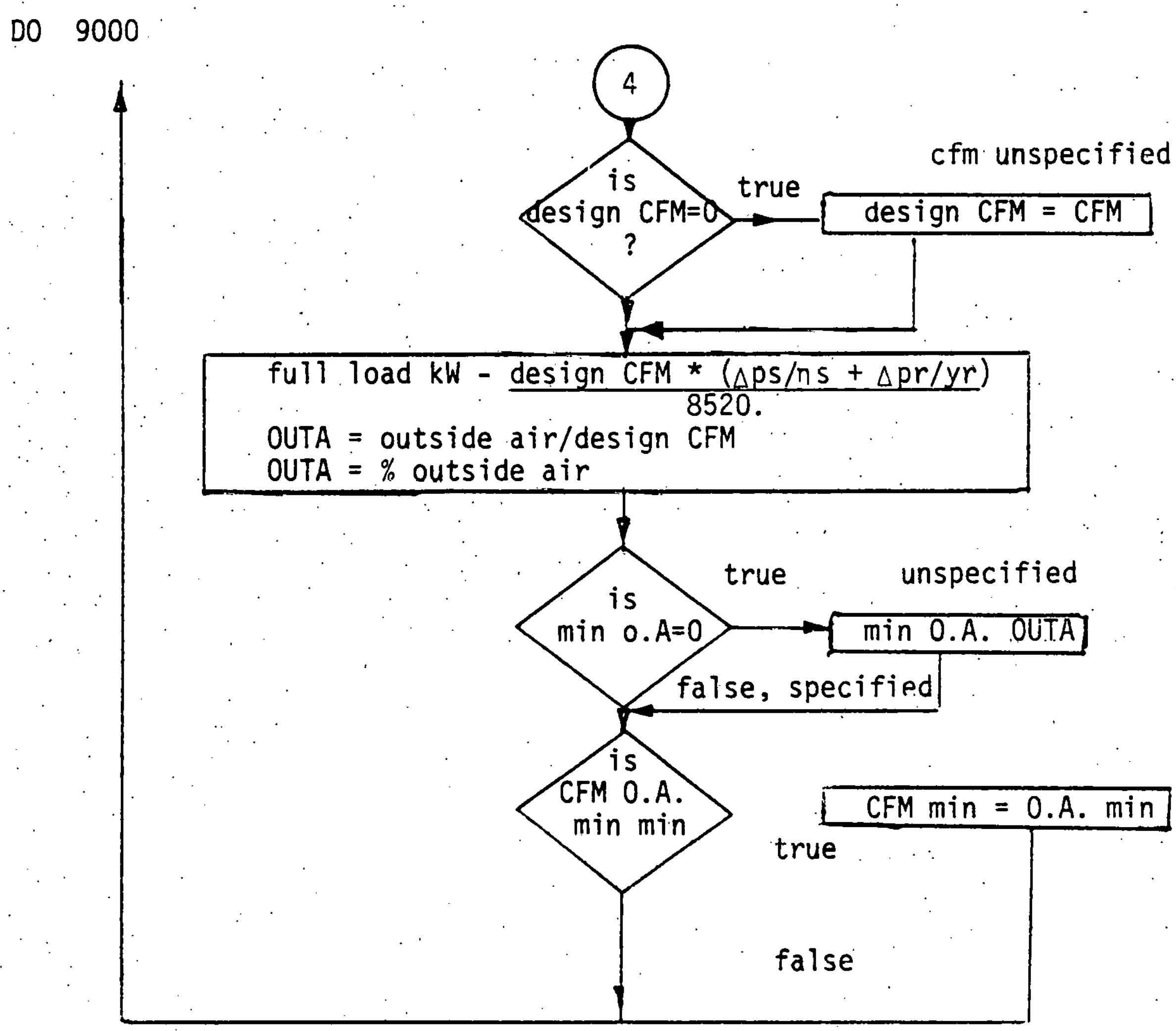
Subroutine DESIGN - Continued


DO
9000
2
loop on number zones in system
DO 100
NZ=1,MSZ
4
50
is
ZP1=plenum
pointer1
true
false
is
ZV1=plenum
pointer2
true
50
true
is
ZP1=plenum
pointer3
false
zone is not a plenum
false
MULT = zone multiplier
ZPZ = pointer to first word in zone
P1 = pointer to type of construction
KA = (g)* ft² zone/ P + KA space conductance
ERMIN = Qmin design
is
ERMIN 0.
?
true
heat load
false, no heat load
ERMIN = KA* (tset - t design heat) - ERMIN
ERMIN = ERMIN* Safety Factor
ERMAX = Qmax design, cooling
true
is
ERMAX≥0
?
cooling load
false, no cooling load
ERMAX = KA* (tset - tdesign coil) - ERMAX
CFM input = ERMAX + Safety Factor
ZVENT = minimum ventilation
3
DO
9000
DO
100

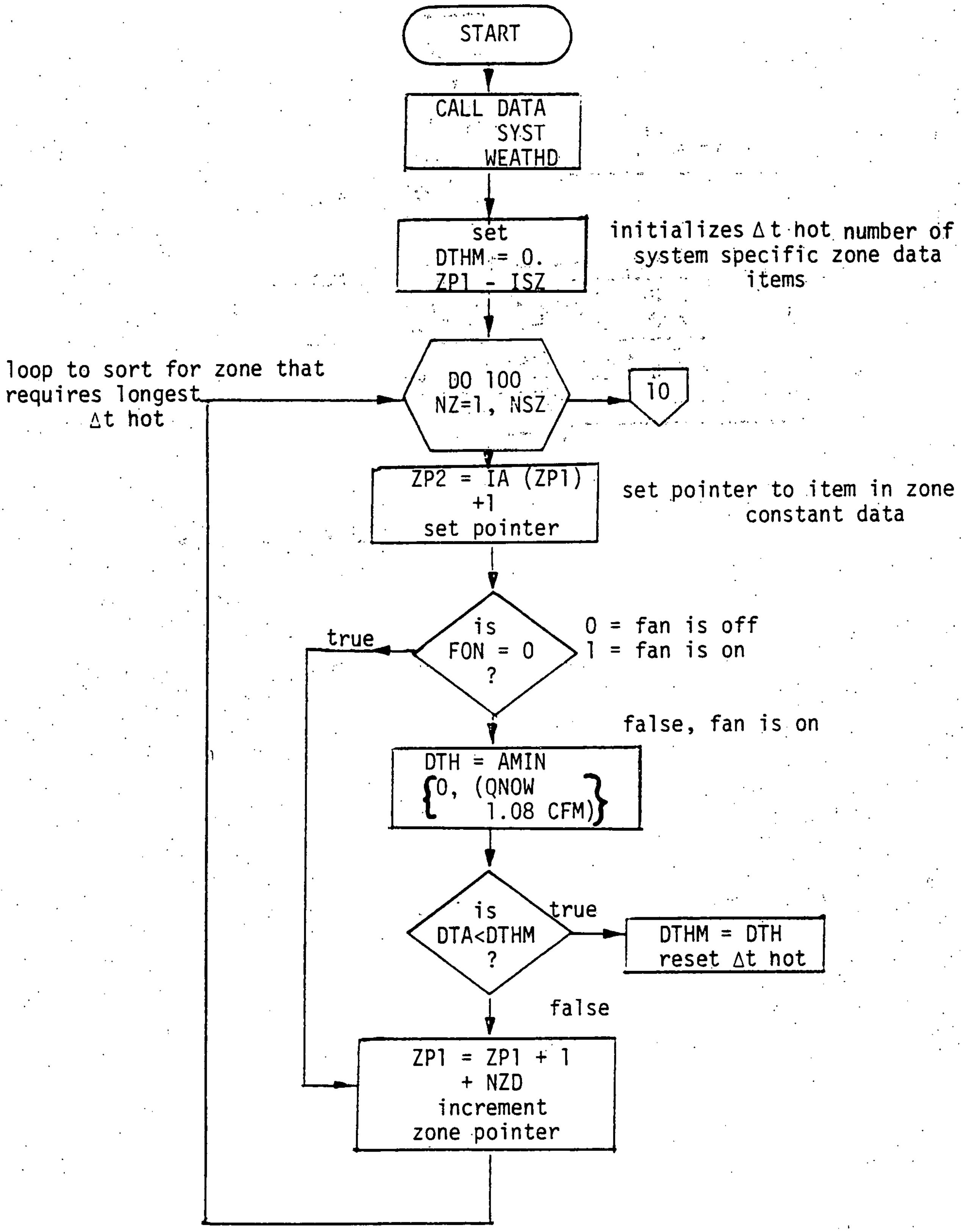
Subroutine DESIGN - Continued


3
DO 9000
DO 100
true, ventilation specified
is ZVENT≠0 ?
false, not specified, calculate
MAX5 = (min O.A. change/wr) *zone volume/60.
MAX2 = (min O.A. CFM/person) *max #people
ZVENT = AMAX1 (MAX5, MAX2)
calculate max zone ventilation CFM
Min. O.A. CFM = ZVENT
OUTA = outside min + zone DA*mult.
ZCFM = design CFM maximum
true
CFM specified
is ZCFM≠0 ?
false, CFM unspecified
MAX5 = ERMAX/[1.08* (tcooling design - tc)]
MAX2 = ERMIN/[1.08* (theating - th)]
MAX3 = (min O.A. change/hr) + ft2 zone/60
MAX4 = (min CFM/ft2) + ft2 zone
ZCFM = AMAX1 (MAX5, MAX2, MAX3, MAX4)
CFM design = ZCFM
CFM = CFM old + ZCFM + mult
50
ZP1 = ZP1 + NZD + 1
DO 9000

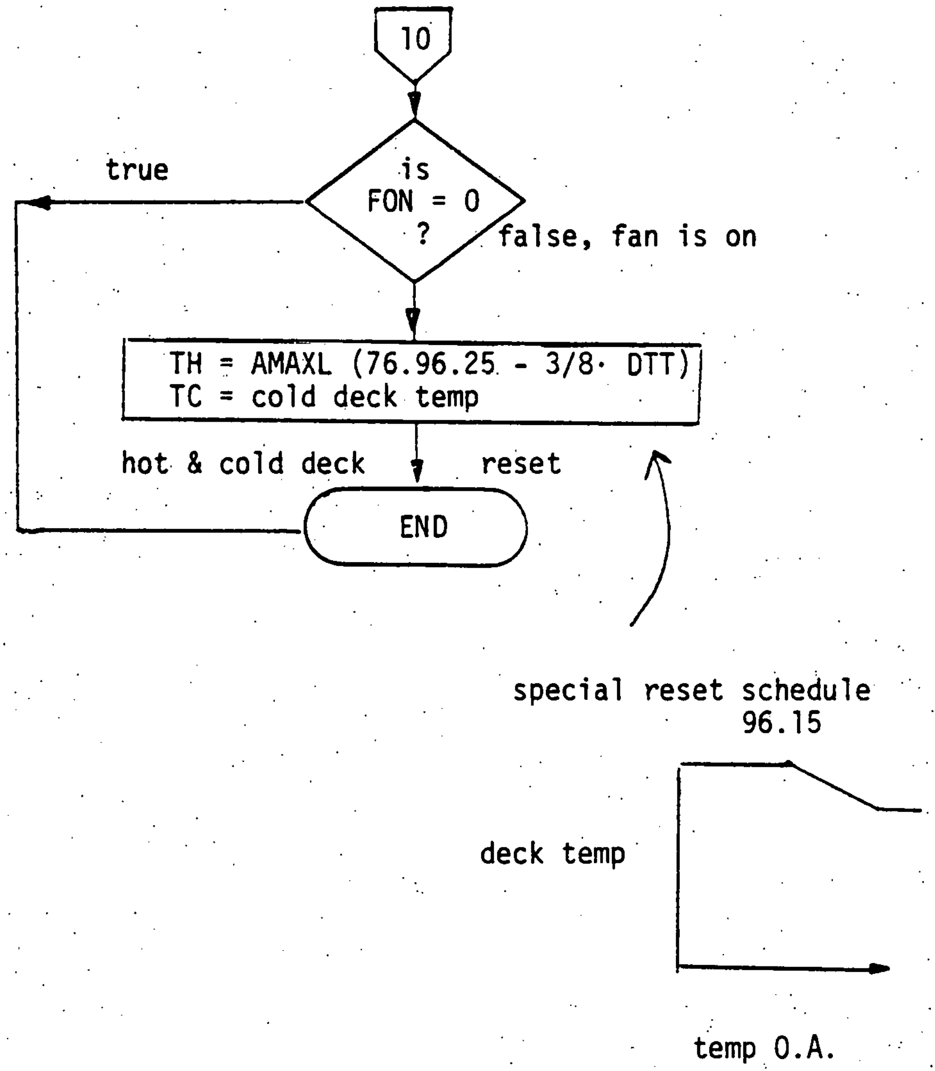
Subroutine DESIGN - Continued



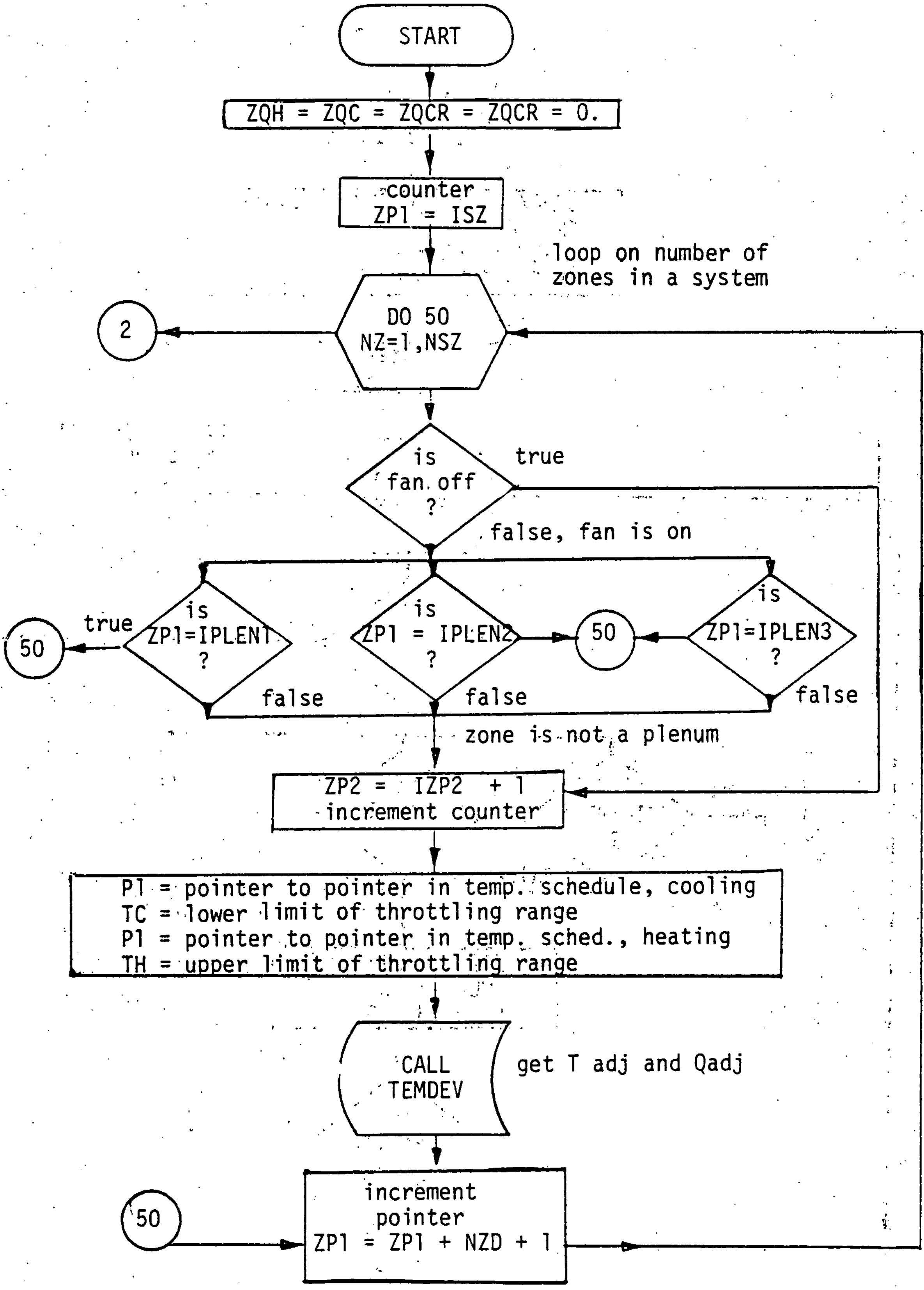
d. Subroutine DKTEMP. April 7, 1977



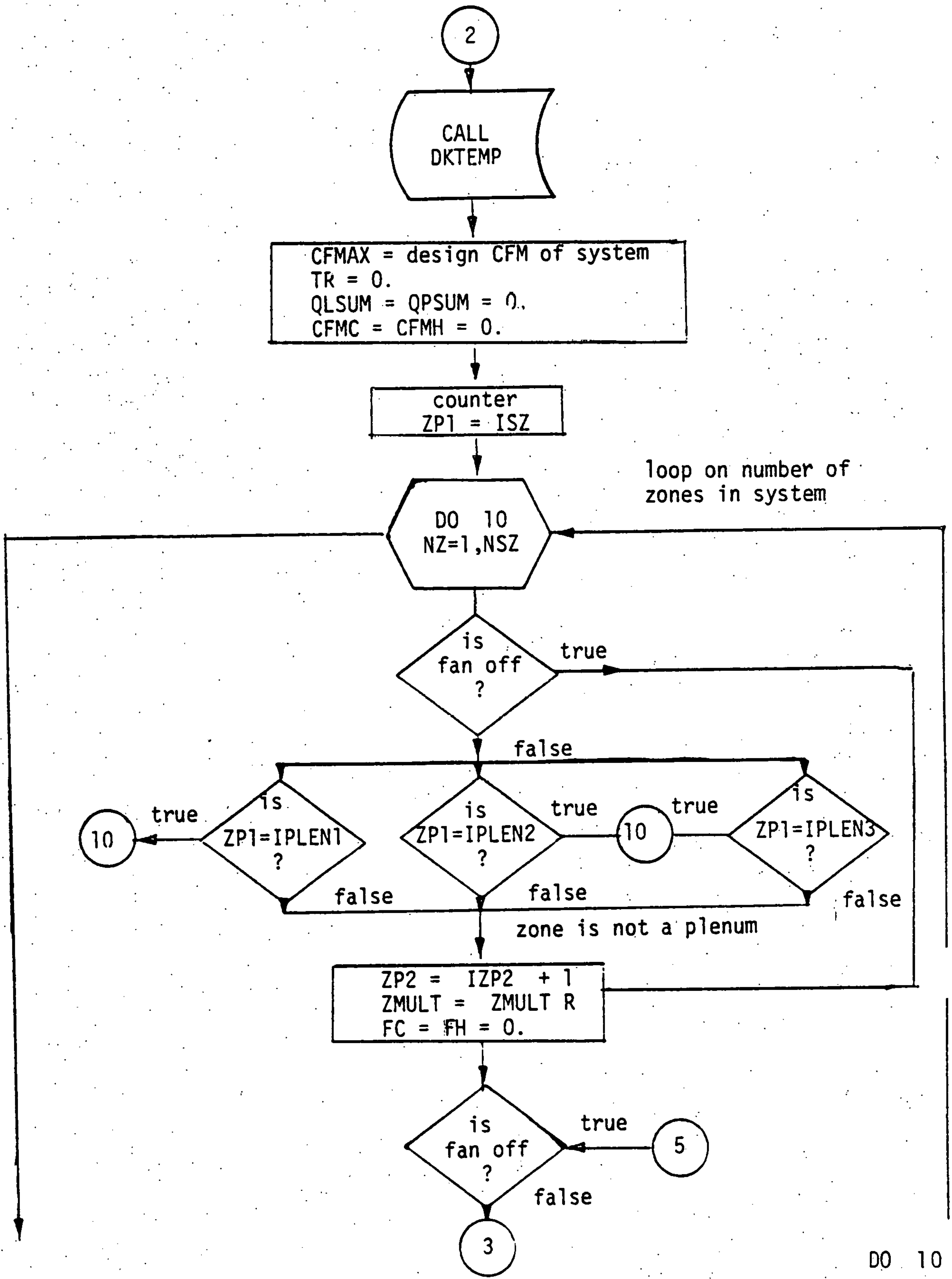
Subroutine DKTEMP - Continued



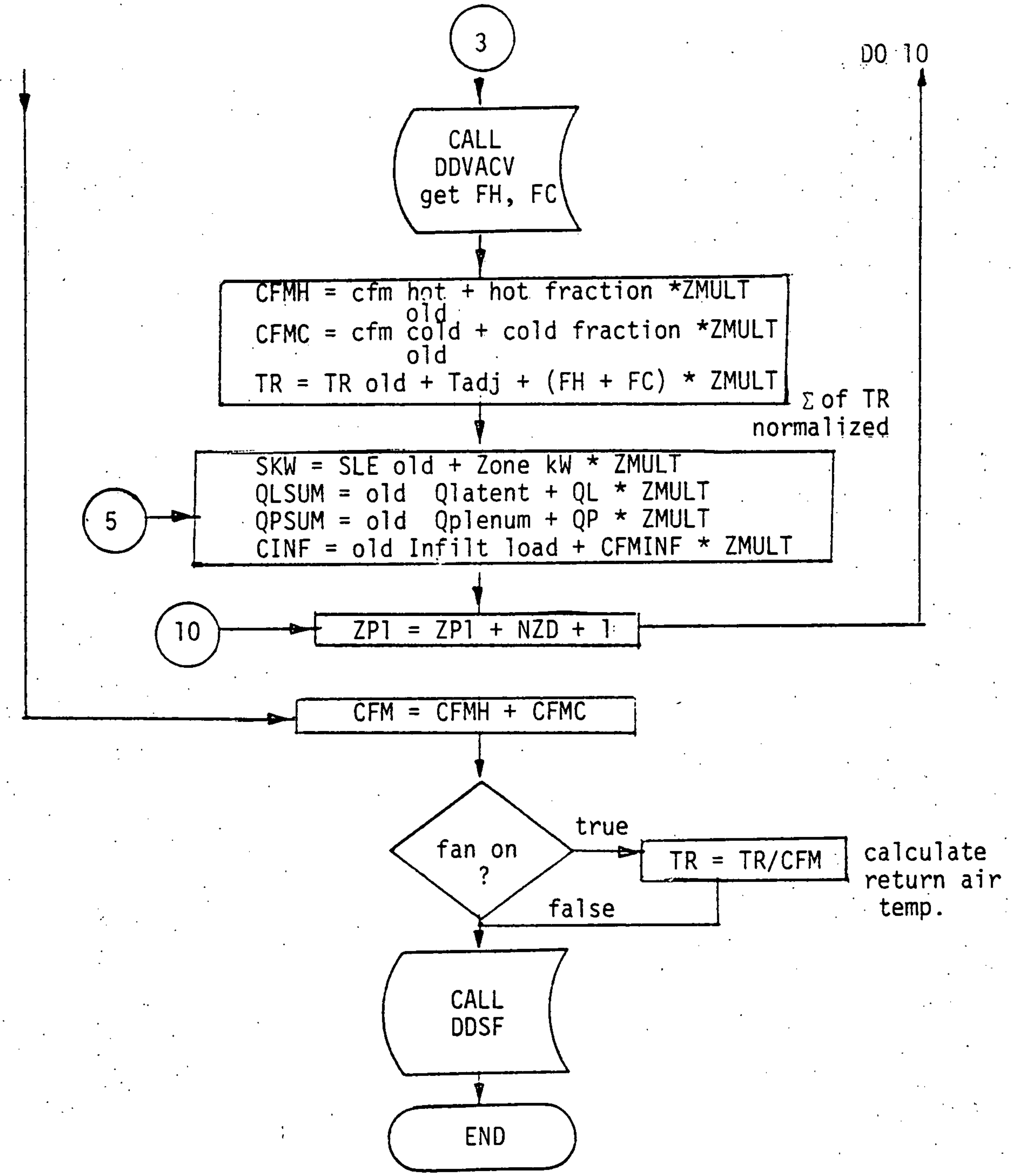
e. Subroutine DOUBLE. June 15, 1977



Subroutine DOUBLE - Continued



Subroutine DOUBLE - Continued

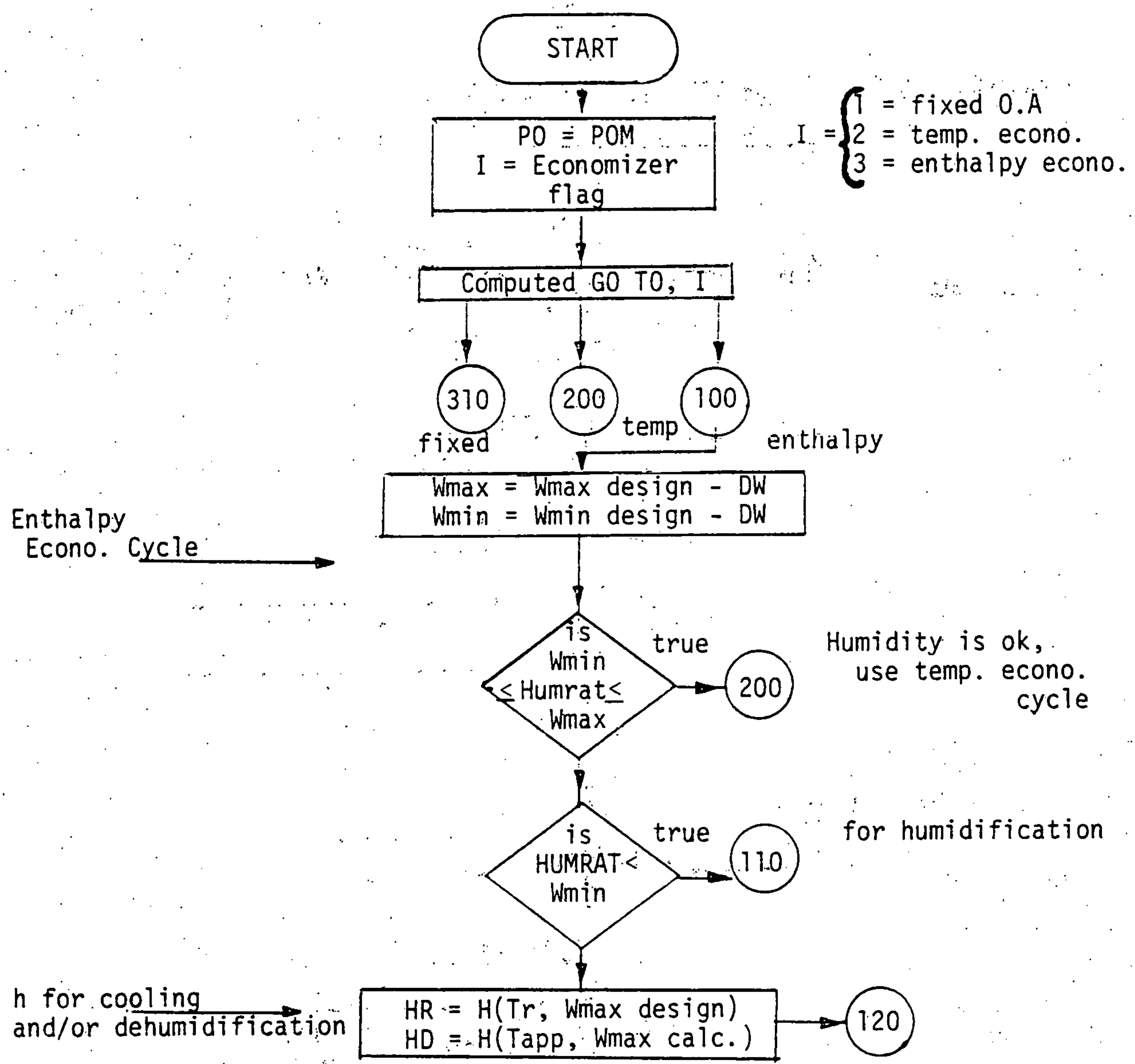


3
DO 10
CALL DDVACV get FH, FC
CFMH = cfm hot old + hot fraction *ZMULT
CFMC = cfm cold old + cold fraction *ZMULT
TR = TR old + Tadj + (FH + FC) * ZMULT
Σ of TR normalized
5
SKW = SLE old + Zone kW * ZMULT
QLSUM = old Qlatent + QL * ZMULT
QPSUM = old Qplenum + QP * ZMULT
CINF = old Infilt load + CFMINF * ZMULT
10
ZP1 = ZP1 + NZD + 1
CFM = CFMH + CFMC
fan on ?
true
TR = TR/CFM
calculate return air temp.
false
CALL DDSF
END

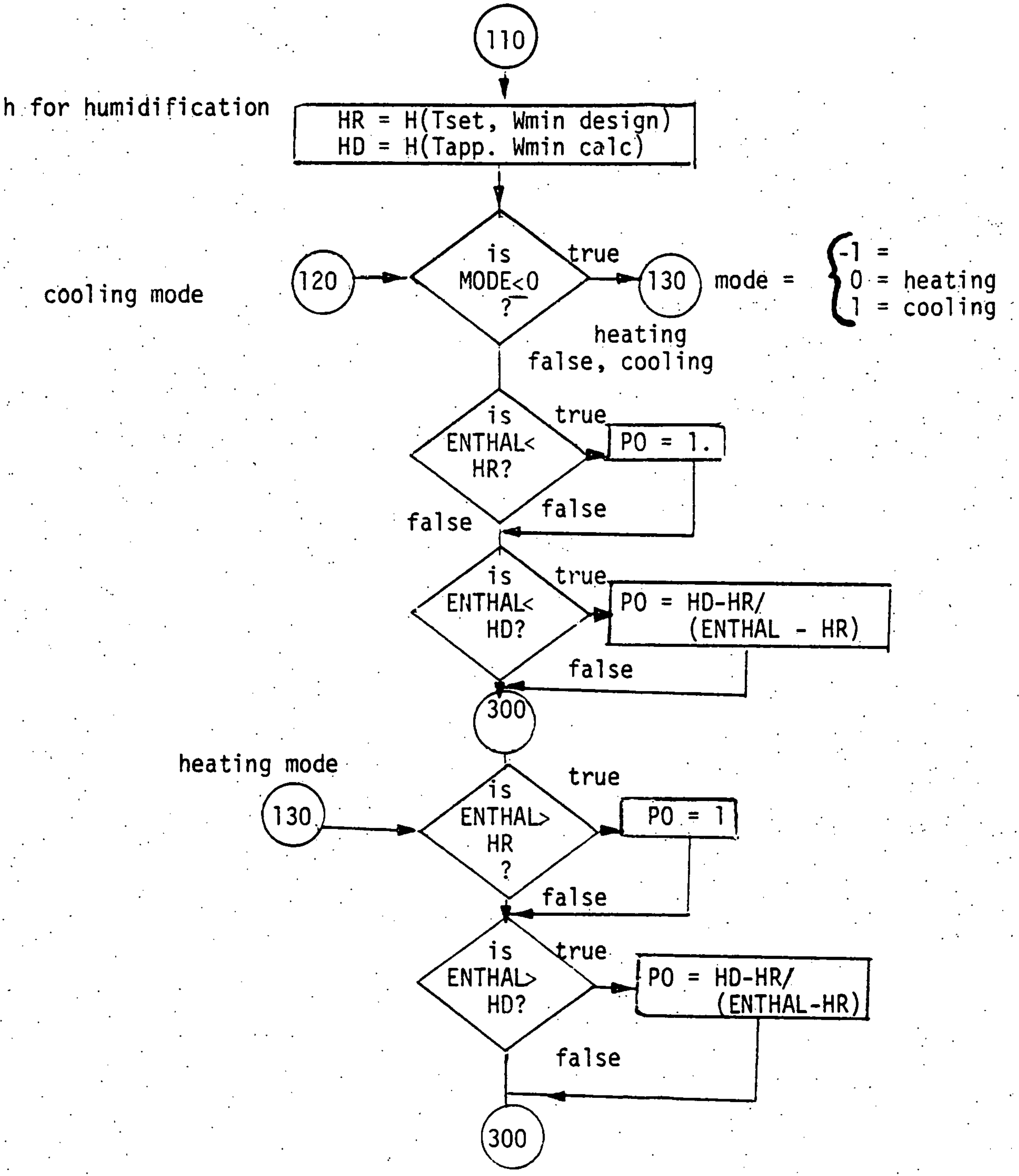
f. Subroutine DST.

See Chap. III.C.8 for a description of Subroutine DST.

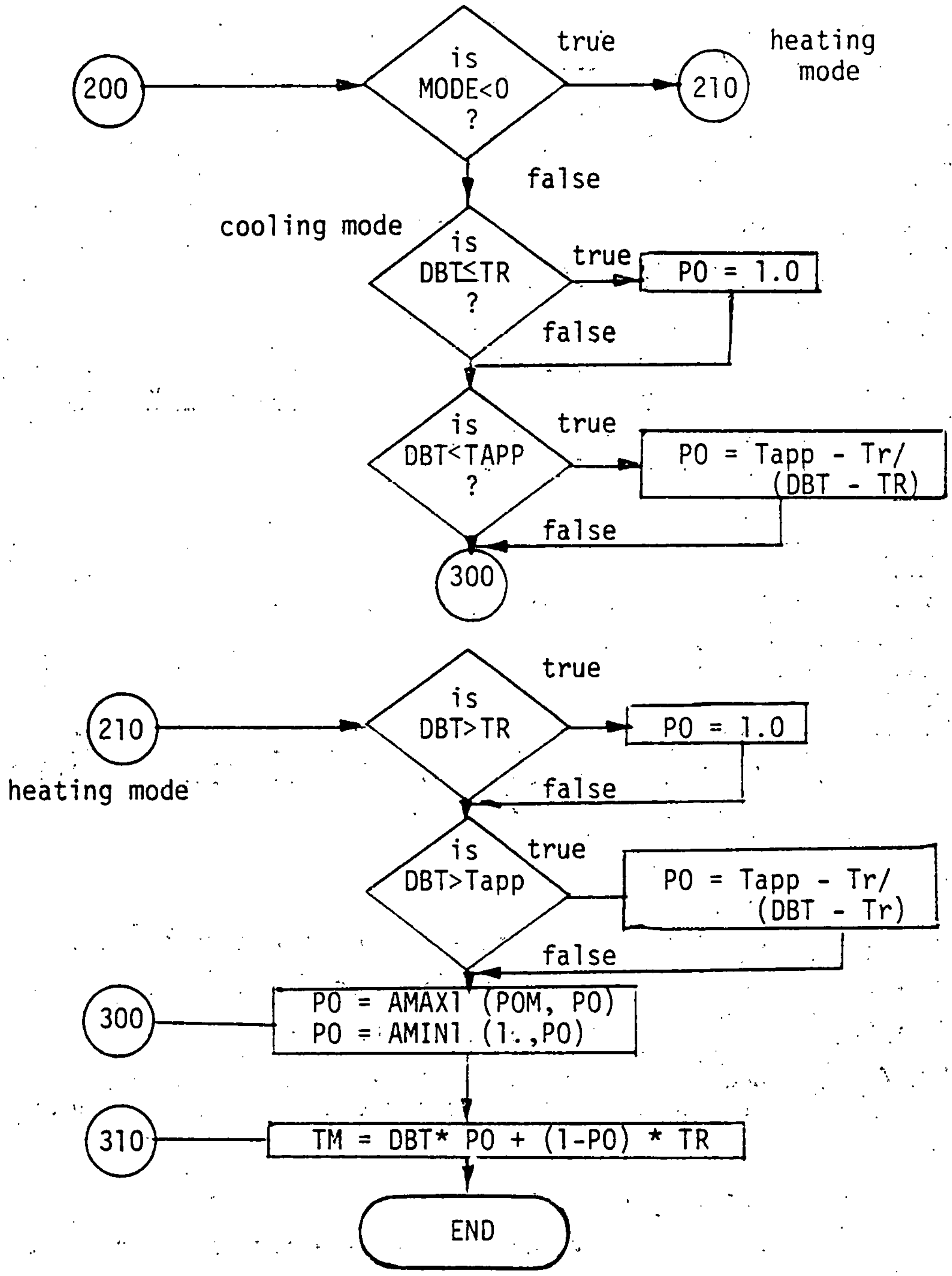
g. Subroutine ECONO. June 15, 1977



Subroutine ECONO - Continued



Subroutine ECONO - Continued



h. Subroutine FANPWR.

Description

The subroutine FANPWR calculates the energy consumed by a system fan operating at part load for three modes of capacity control: damper control, inlet vane control, and speed control.

Subprograms Calling This Subroutine

SUBROUTINE DDSF
SUBROUTINE SDSF
SUBROUTINE UNITHV

Subprogram Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Block	Variables Placed in Common Block

/DATA/	AA(NSP+20), IA(NSP+38)
	AA(NSP+5), AA(NSP+26)
		AA(NSP+5)

/SYST/	CFM	None

Declarations

```
INTEGER IA(5000), ZP1, ZP2, P1, P2, SN, ZN, PSS
EQUIVALENCE (AA(1),IA(1))
DIMENSION PTLD (4,3)
REAL MODE

DATA PTLD/0.0015302776, 0.0052080574, 1.1086242, -0.11635563,
          0.35071223, 0.3080535, -0.5413736, 0.87198832, 0.37073425,
          0.97250253, -0.4240761, 0.0/
```

Input

Name	Description
CFM	Current air flow rate through the fan (ft^3 /min)
AA(NSP+20)	Design air flow rate (ft^3 /min)
AA(NSP+26)	Efficiency of fan (dimensionless)
PTLD (4,3)	Coefficients of the second degree function used to calculate the per cent of rated capacity of a fan (dimensionless)

Output

Name	Description
AA(NSP+5)	Hourly energy consumption of a fan operating at part load

Calculation Procedure

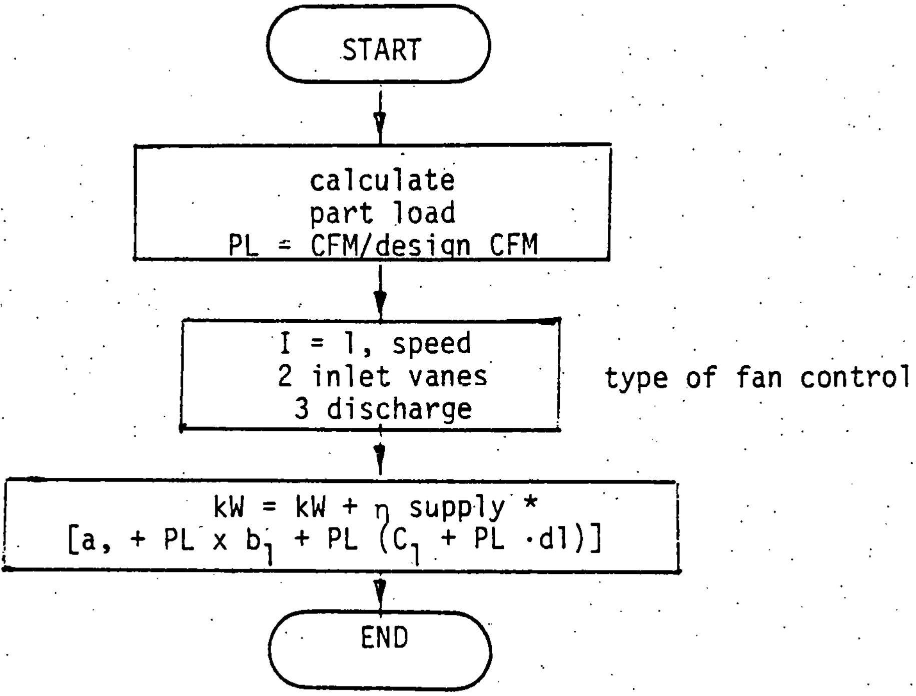
1. Calculate the percentage of fan part-load operation

2. Identify the method of capacity control.

3. Calculate the energy consumption of the fan using the fan efficiency, the per cent of part-load operation, and the appropriate coefficients for capacity control.

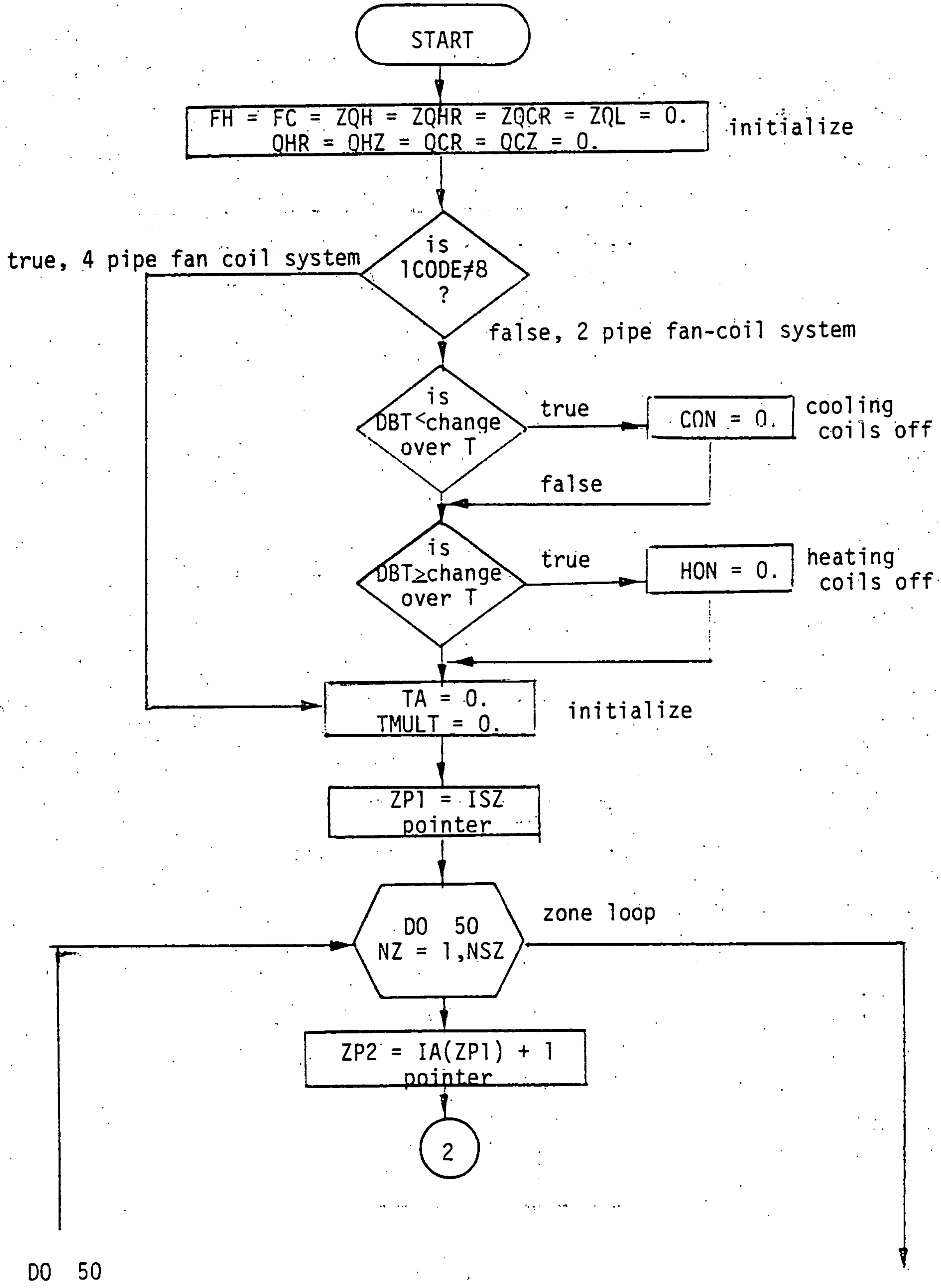
ASHRAE Verification

The method used in Subroutine FANPWR by the Cal-ERDA code to simulate the part-load operation of fans is the same as the qualitative description given in ASHRAE (Ref. 2).

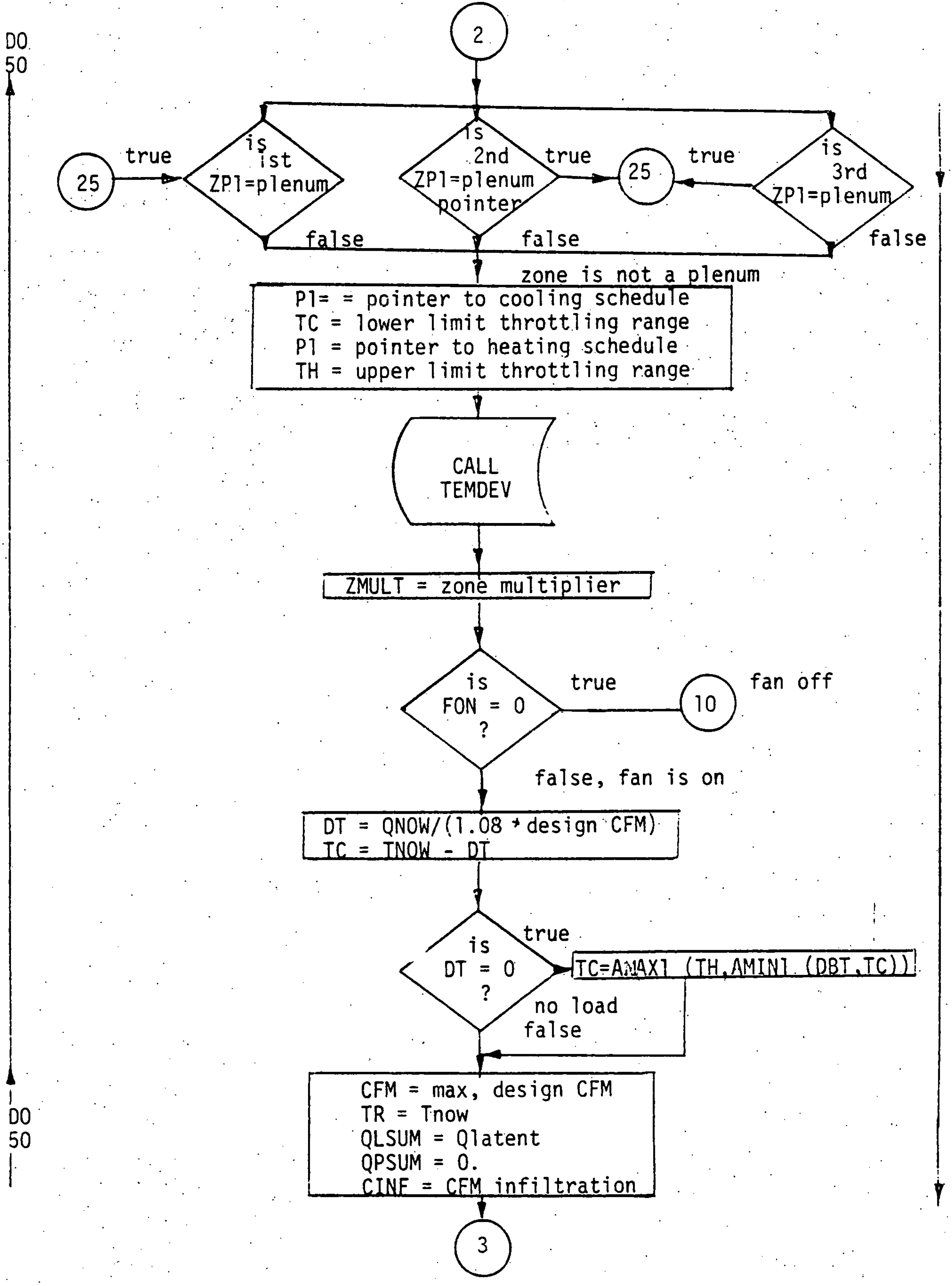

START
calculate
part load
PL = CFM/design CFM
I = 1, speed
2 inlet vanes
3 discharge
type of fan control
kW = kW + η supply *
[a, + PL x b1 + PL (C1 + PL ·d1)]
END

i. Subroutine FCOIL.

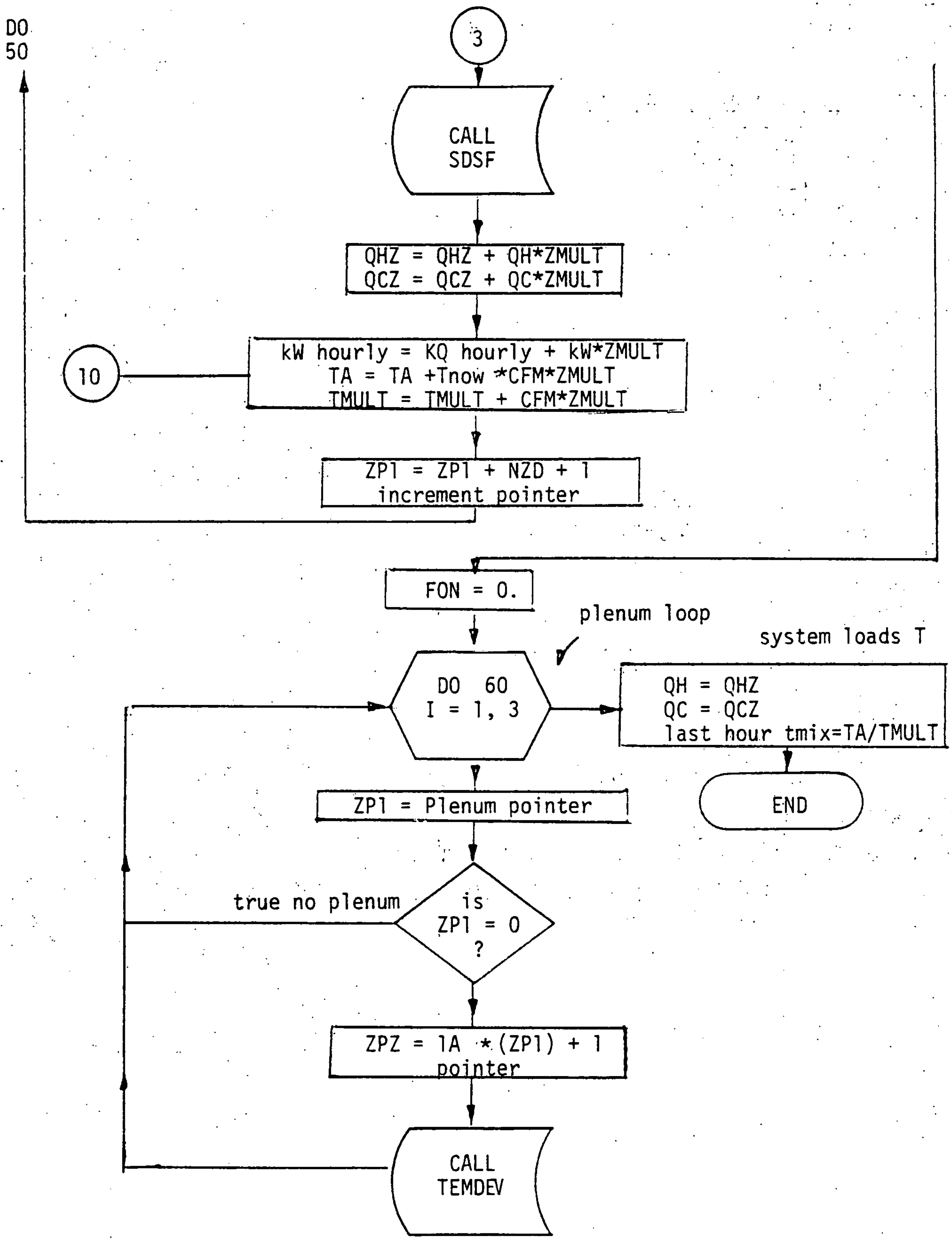
June 27, 1977



Subroutine FCOIL - Continued



Subroutine FCOIL - Continued



j. Function H.

Description

The function subprogram H calculates the enthalpy of moist air in Btu/lb of dry air.

Subprograms Calling by This Function Subprogram

SUBROUTINE ECONO
SUBROUTINE DDSF
SUBROUTINE SDSF
SUBROUTINE UNITHV

Subprogram Called by This Function Subprogram

None

Common Blocks

None

Declarations

None

Input

Name	Description
DB	Dry-bulb temperature (°F)
W	Humidity ratio of moist air (lb H_2O/lb dry air)

Output

Name	Description
H	Enthalpy of moist air (Btu/lb dry air)

Calculation Procedure

Compute H using DB and W.

H = 0.24 * DB + (1061. + 0.444 * DB) * W

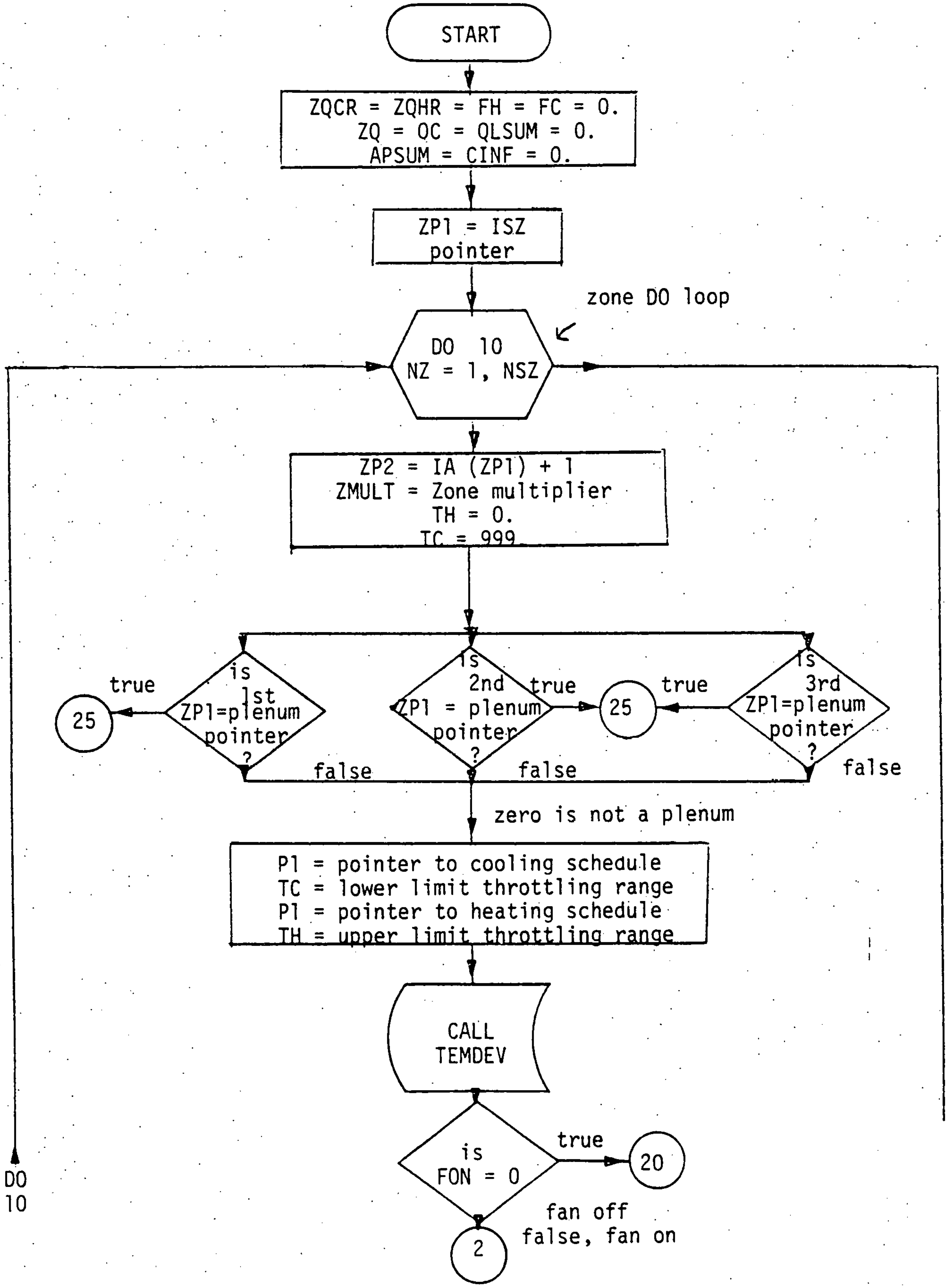
ASHRAE Verification

The equation used in the H function subprogram by the Cal-ERDA code is the same as the H equation in ASHRAE (Ref. 1).

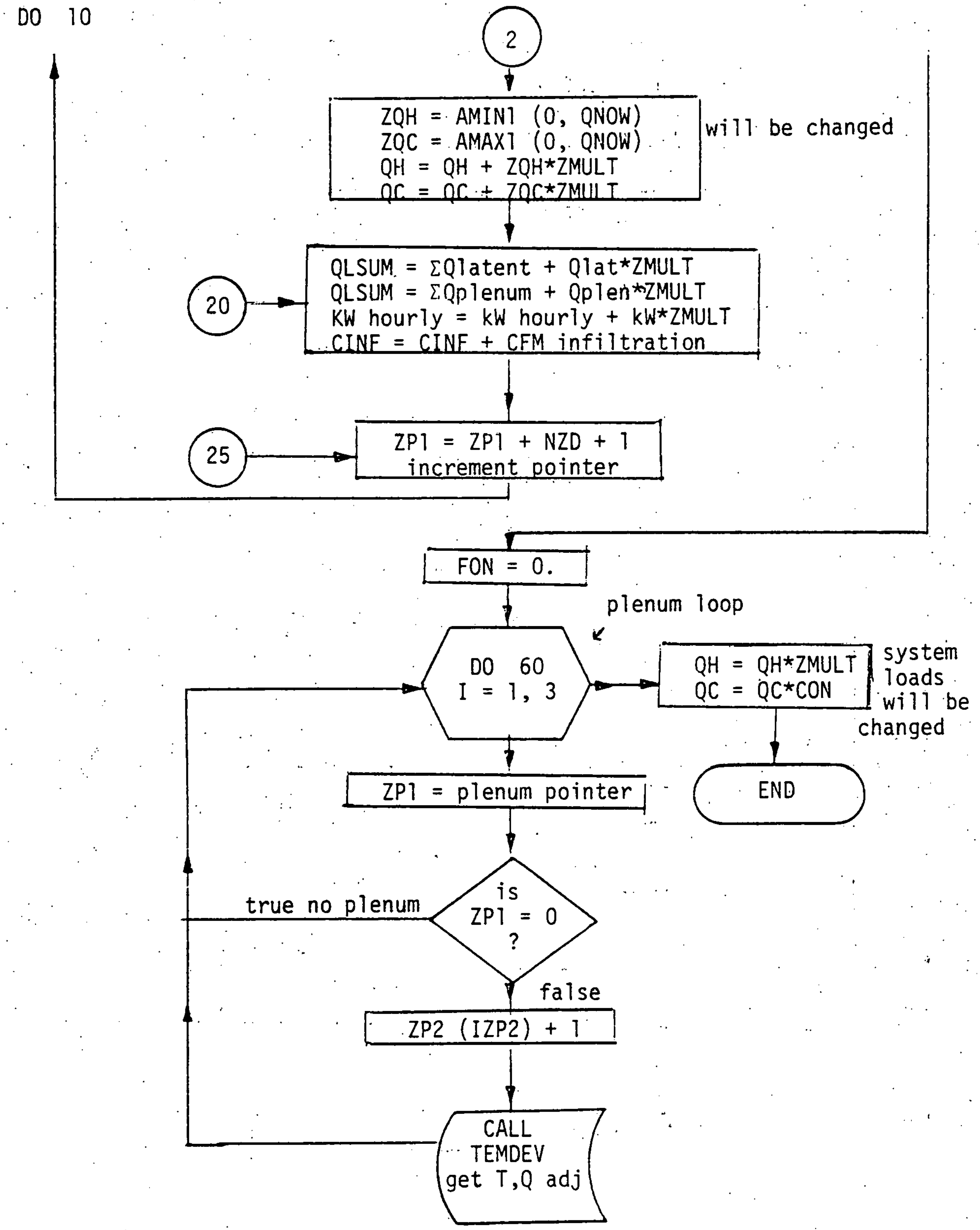
k. Subroutine HOLDAY.

See Chap. III.C.13 for a description of Subroutine HOLDAY.

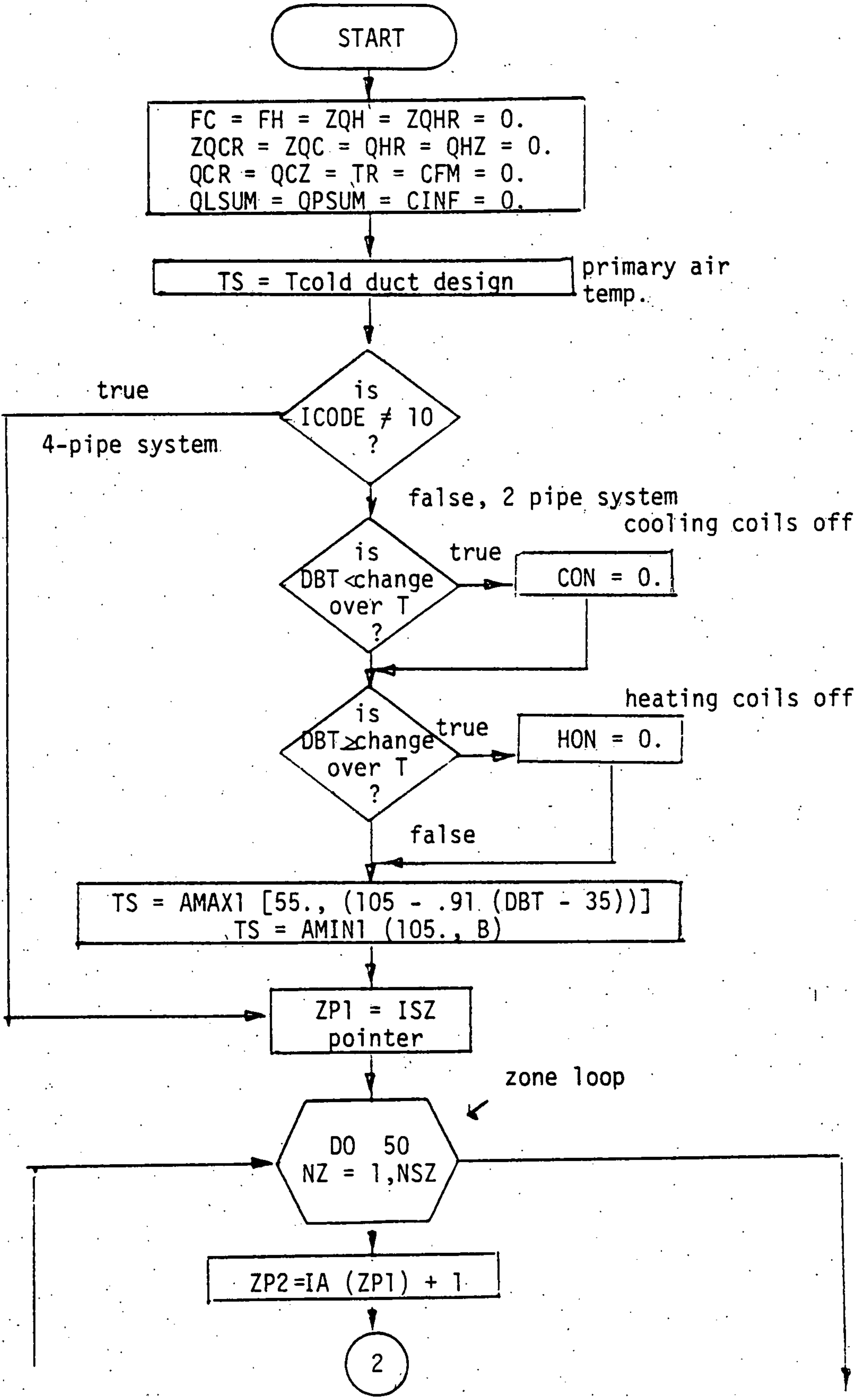
1. Subroutine HTPUMP. June 15, 1977



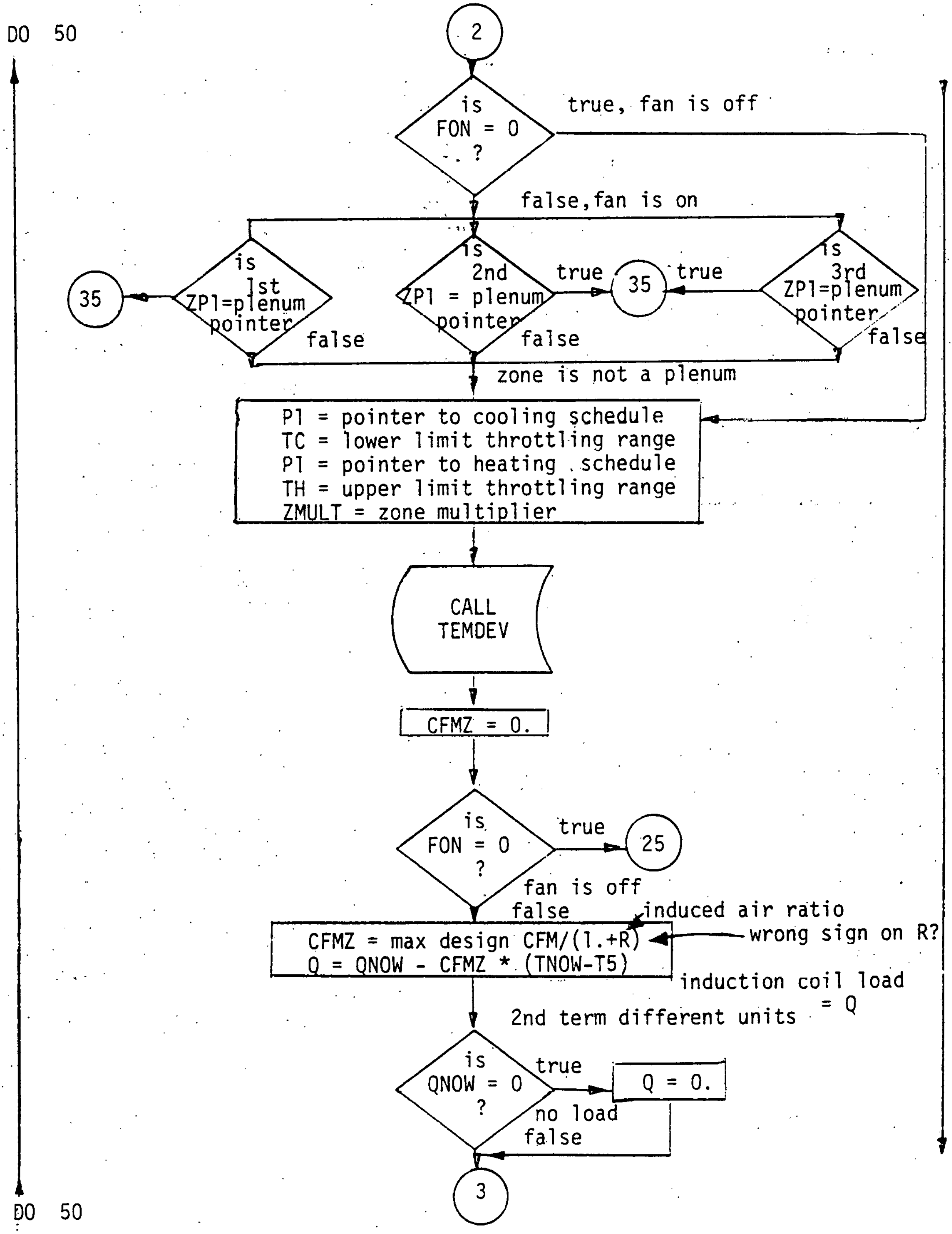
Subroutine HTPUMP - Continued



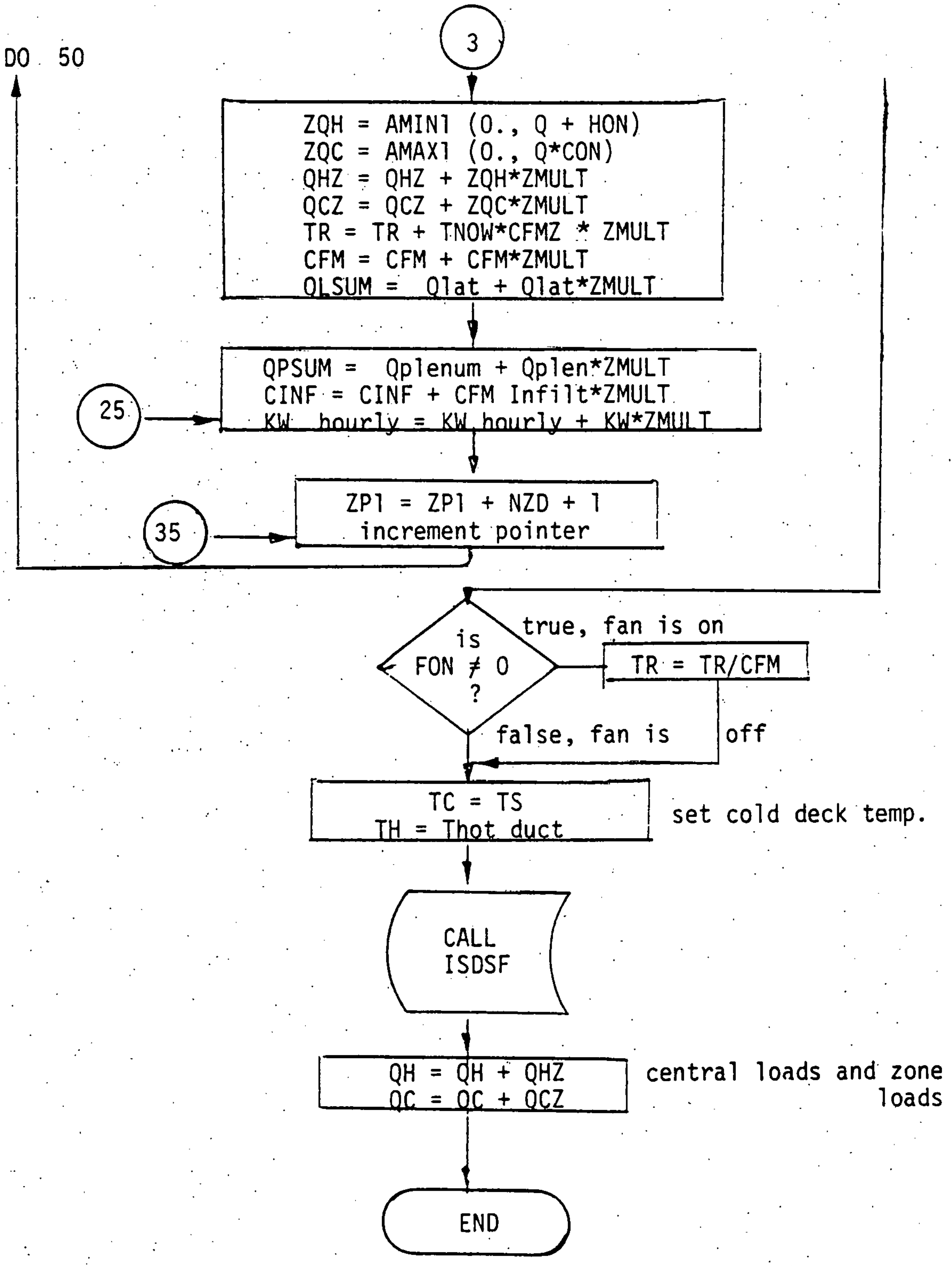
m. Subroutine INDUC. June 15, 1977



Subroutine INDUC - Continued



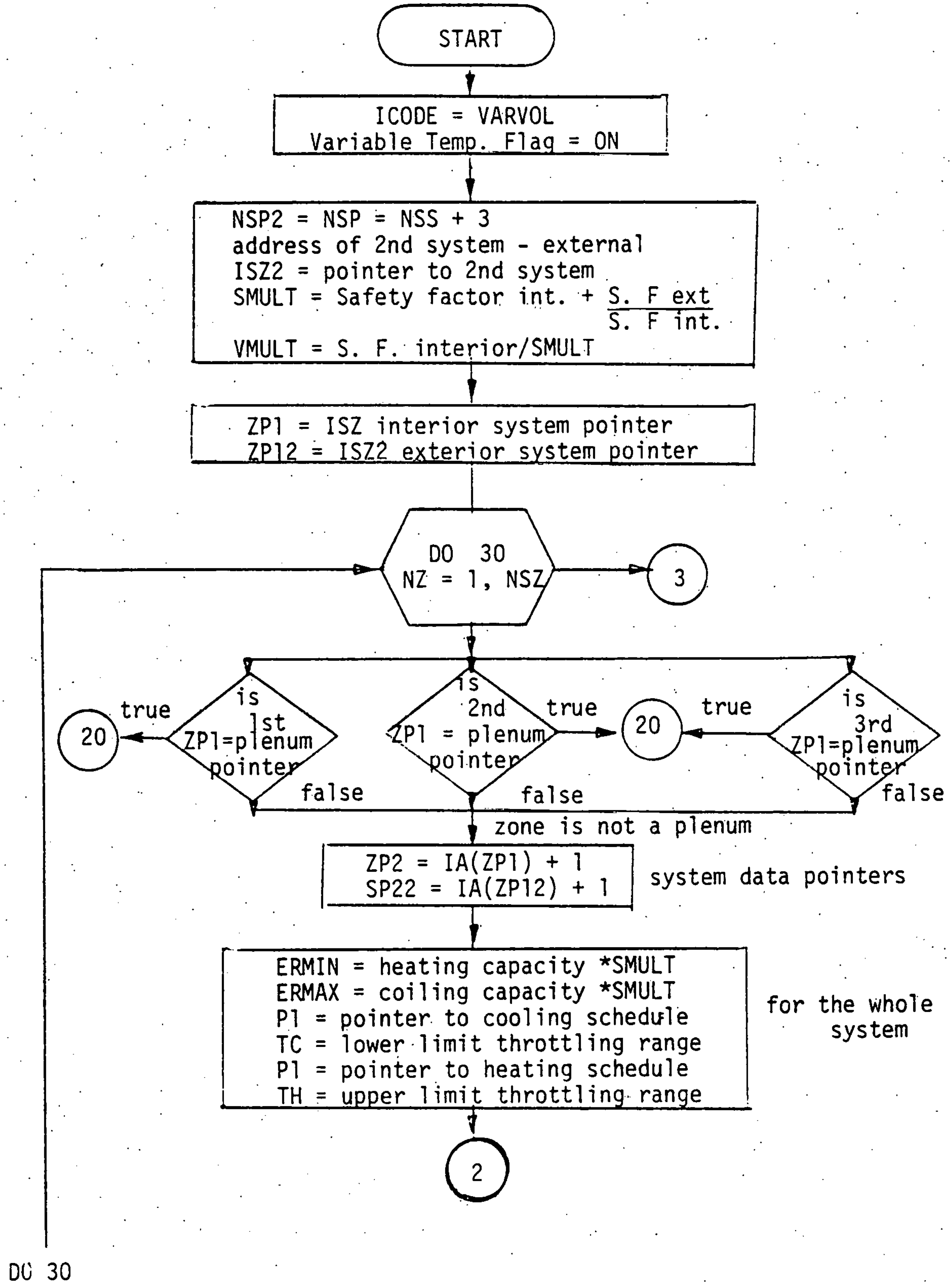
Subroutine INDUC - Continued



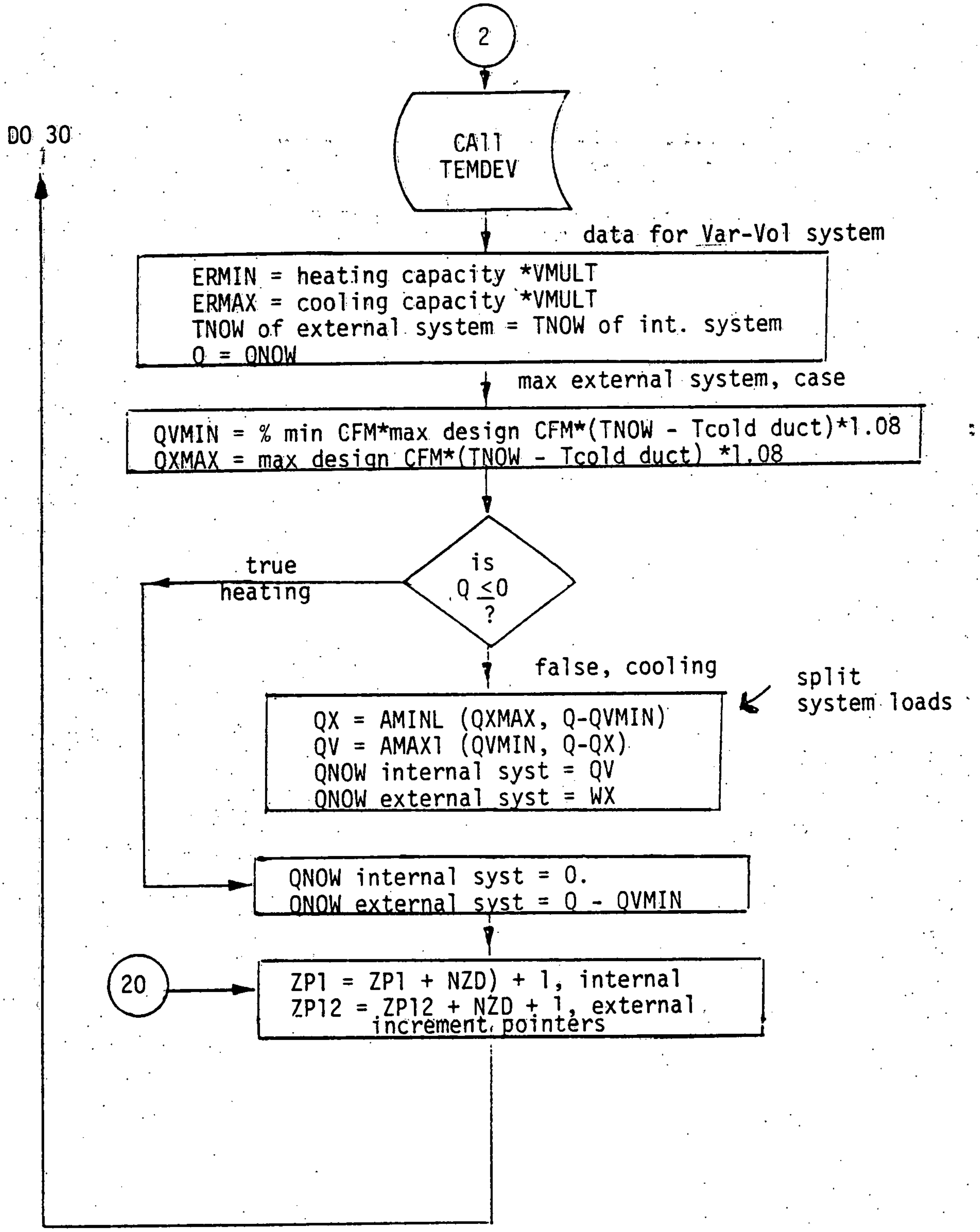
n. Subroutine ISDSF. June 15, 1977

Subroutine ISDSF is exactly the same as Subroutine SDSF. It will be changed in the future to handle the induction single-duct system.

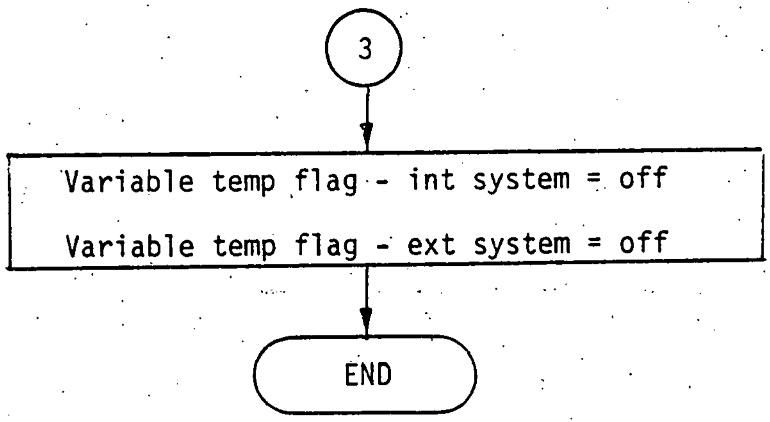
o. Subroutine IVEFI. June 15, 1977


START
ICODE = VARVOL
Variable Temp. Flag = ON
NSP2 = NSP = NSS + 3
address of 2nd system - external
ISZ2 = pointer to 2nd system
SMULT = Safety factor int. + S. F ext / S. F int.
VMULT = S. F. interior/SMULT
ZP1 = ISZ interior system pointer
ZP12 = ISZ2 exterior system pointer
DO 30
NZ = 1, NSZ
3
is 1st ZP1=plenum pointer
true
20
false
is 2nd ZP1 = plenum pointer
true
20
false
is 3rd ZP1=plenum pointer
true
false
zone is not a plenum
ZP2 = IA(ZP1) + 1
SP22 = IA(ZP12) + 1
system data pointers
ERMIN = heating capacity *SMULT
ERMAX = coiling capacity *SMULT
P1 = pointer to cooling schedule
TC = lower limit throttling range
P1 = pointer to heating schedule
TH = upper limit throttling range
for the whole system
2
DC 30

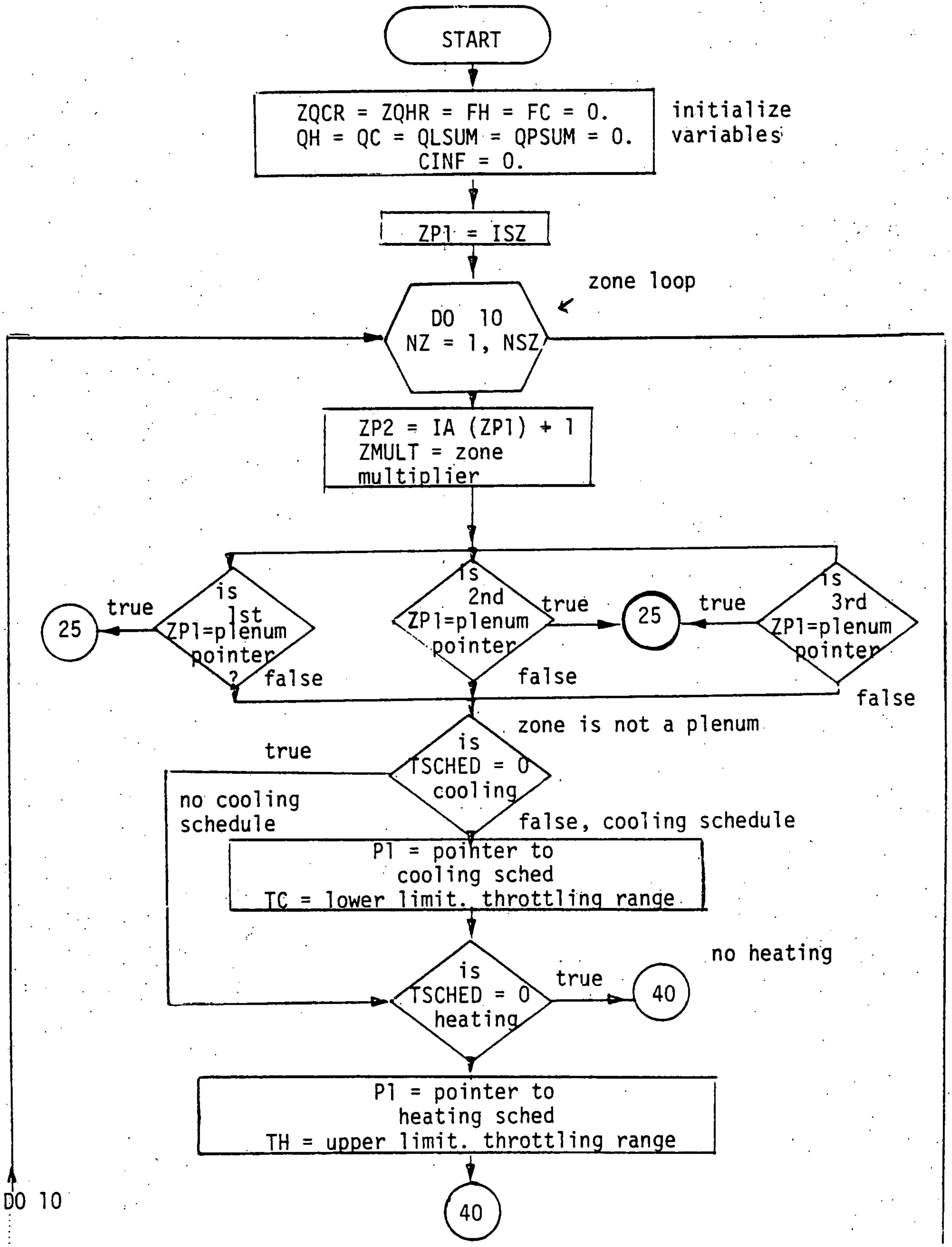
Subroutine IVEFI - Continued


2
DO 30
CAll TEMDEV
data for Var-Vol system
ERMIN = heating capacity *VMULT
ERMAX = cooling capacity *VMULT
TNOW of external system = TNOW of int. system
Q = QNOW
max external system, case
QVMIN = % min CFM*max design CFM*(TNOW - Tcold duct)*1.08
QXMAX = max design CFM*(TNOW - Tcold duct) *1.08
is Q ≤0 ?
true heating
false, cooling
split system loads
QX = AMINL (QXMAX, Q-QVMIN)
QV = AMAX1 (QVMIN, Q-QX)
QNOW internal syst = QV
QNOW external syst = WX
QNOW internal syst = 0.
QNOW external syst = Q - QVMIN
20
ZP1 = ZP1 + NZD) + 1, internal
ZP12 = ZP12 + NZD + 1, external
increment pointers

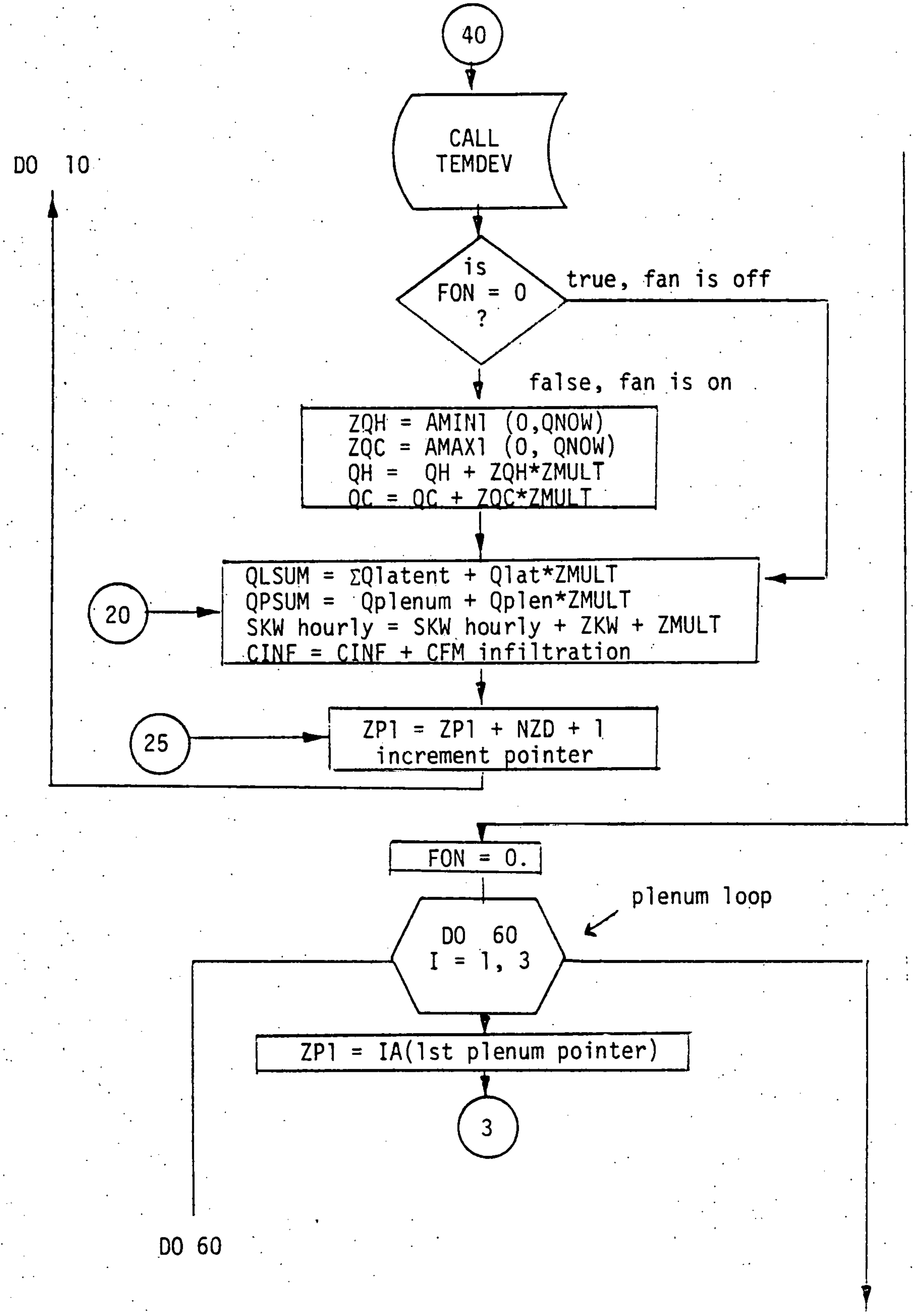
Subroutine IVEFI - Continued


3
Variable temp flag - int system = off
Variable temp flag - ext system = off
END

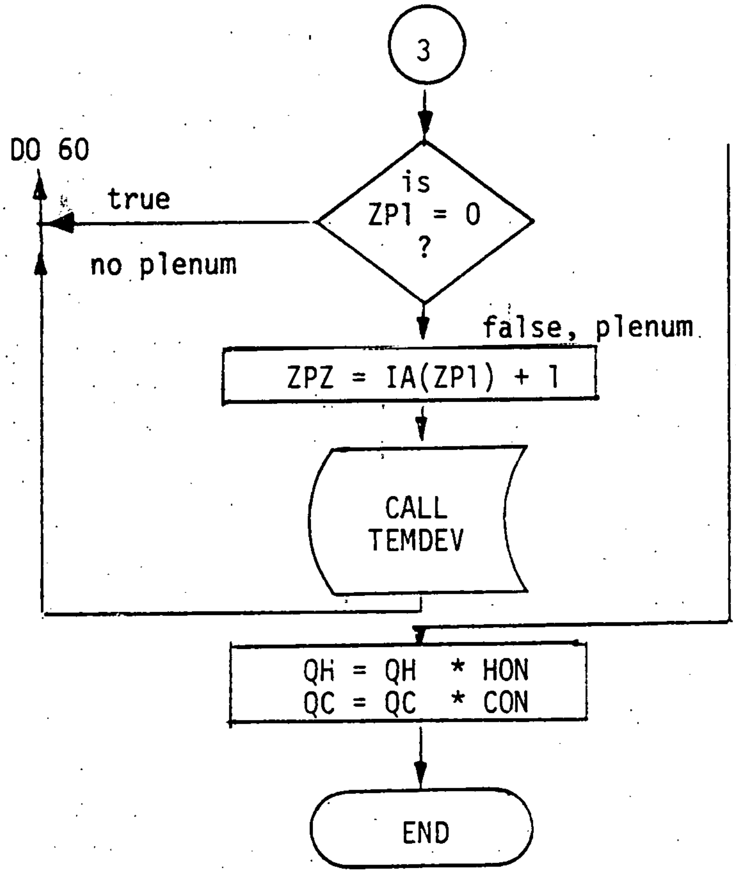
p. Subroutine PANEL. June 15, 1977



Subroutine PANEL - Continued



Subroutine PANEL - Continued



q. Function PPWVMS.

FUNCTION PPWVMS (TEMP)

Description

The function subprogram PPWVMS calculates the partial pressure of water vapor in moisture-saturated air in inches of mercury.

Subprograms Calling This Function

SUBROUTINE WDREAD
SUBROUTINE DESIGN

Subprograms Called by This Function

None

Common Block

None

Declarations

```
DIMENSION A (6), B (4)
DATA A/-7.90298, 5.02808, -1.3816E-7, 11.344, 8.1328E-3, -3.49149/
DATA B/-9.09718, -3.56654, 0.876793, 0.0060272/
```

Input

Name	Description
TEMP	Outside air temperature (°F)

Output

Name	Description
PPWVMS	Partial pressure of water vapor in moisture-saturated air (in. Hg)

Calculation Procedure

1. Obtain outside air temperature from the argument.
2. Transform outside air temperature to degrees Kelvin.
3. Check the temperature against 273.16 K.
   a. If temperature is greater than 273.16 K, then set
   Z = 373.16/T.

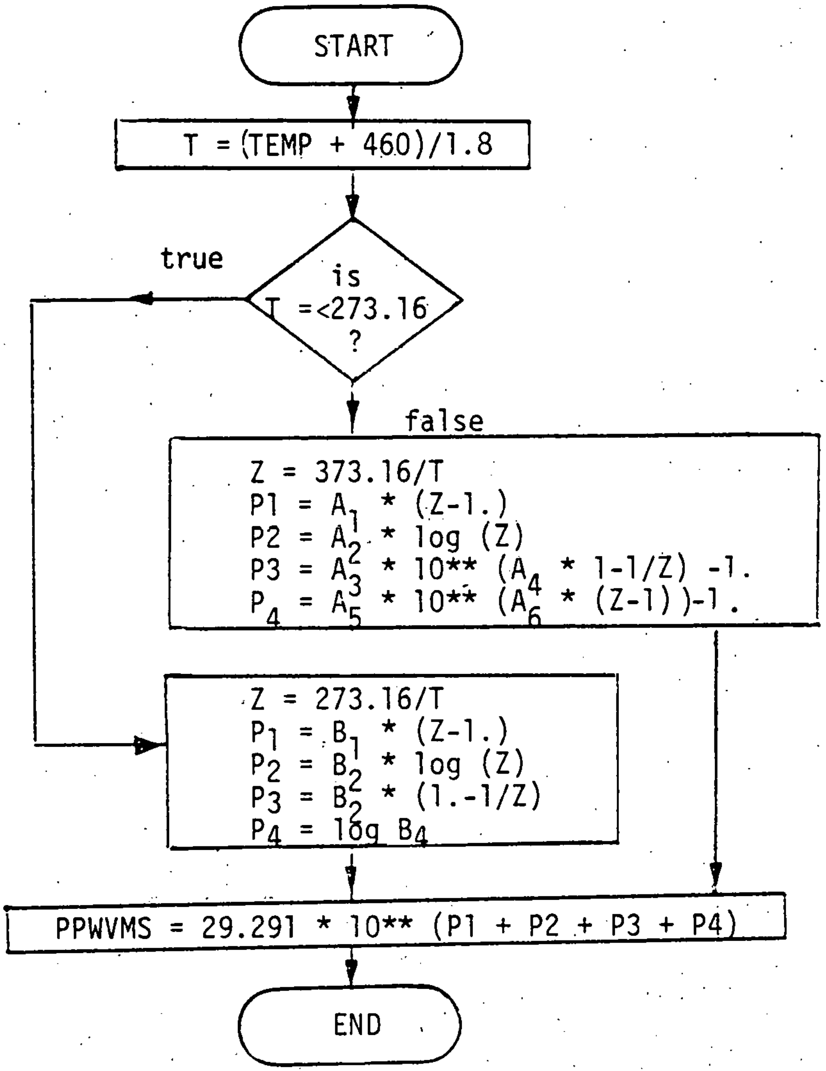
   Calculate P1, P2, P3, P4.

   b. If temperature is less than 273.16 K, then set
   Z = 273.16/T.

   Calculate P1, P2, P3, P4.
4. Calculate the partial pressure of water vapor in moisture-saturated air.

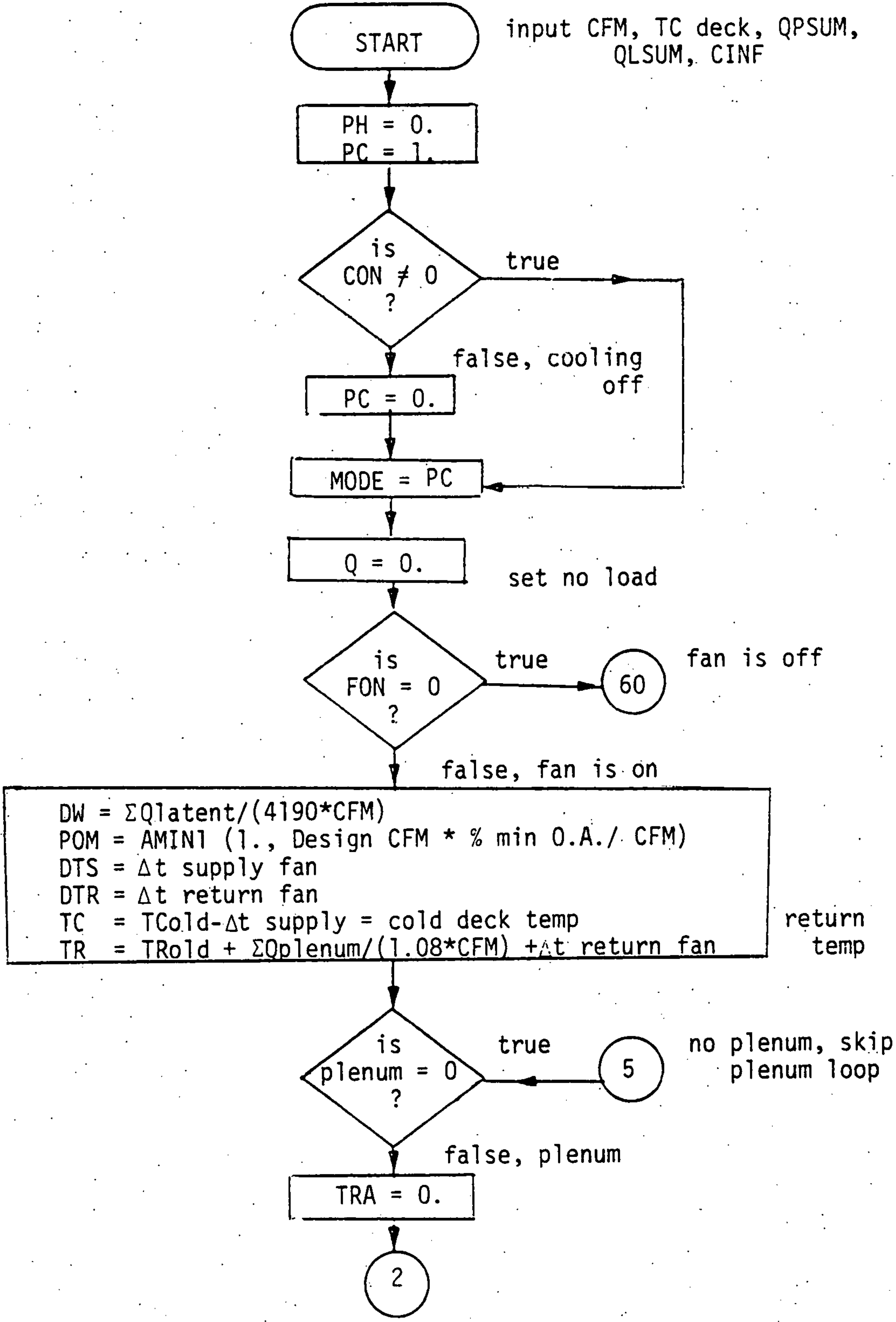
ASHRAE Verification

The algorithm used in the PPWVMS function subprogram by the Cal-ERDA code is the same as the PVS algorithm in ASHRAE (Ref. 1).

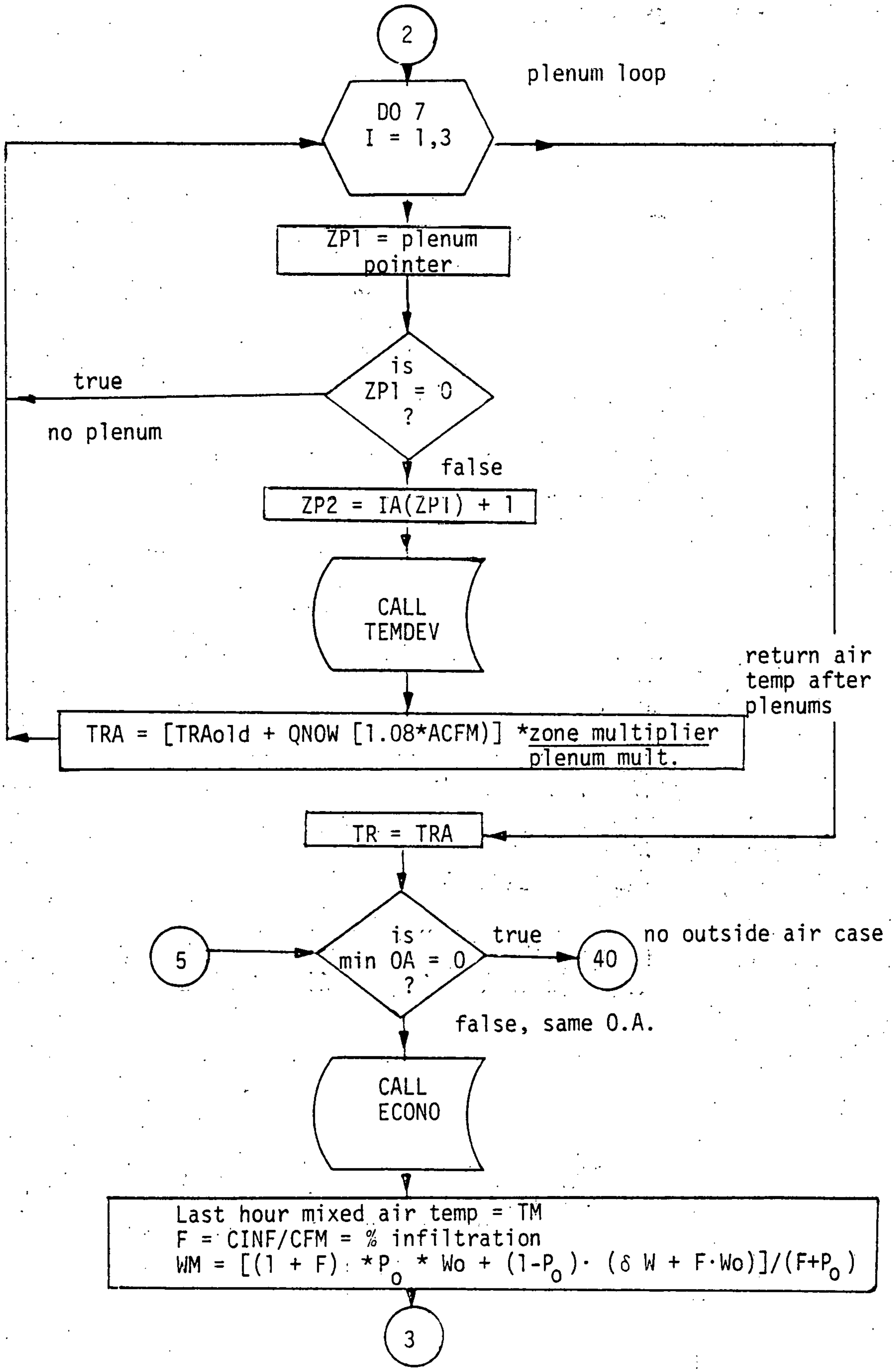

START
T = (TEMP + 460)/1.8
is T =<273.16 ?
true
false
Z = 373.16/T
$P1 = A_1 * (Z-1.)$
$P2 = A_2 * \log (Z)$
$P3 = A_3 * 10** (A_4 * 1-1/Z) -1.$
$P_4 = A_5 * 10** (A_6 * (Z-1))-1 .$
Z = 273.16/T
$P_1 = B_1 * (Z-1.)$
$P_2 = B_2 * \log (Z)$
$P_3 = B_2 * (1.-1/Z)$
$P_4 = \log B_4$
PPWVMS = 29.291 * 10** (P1 + P2 + P3 + P4)
END

r. Subroutine SDSF.

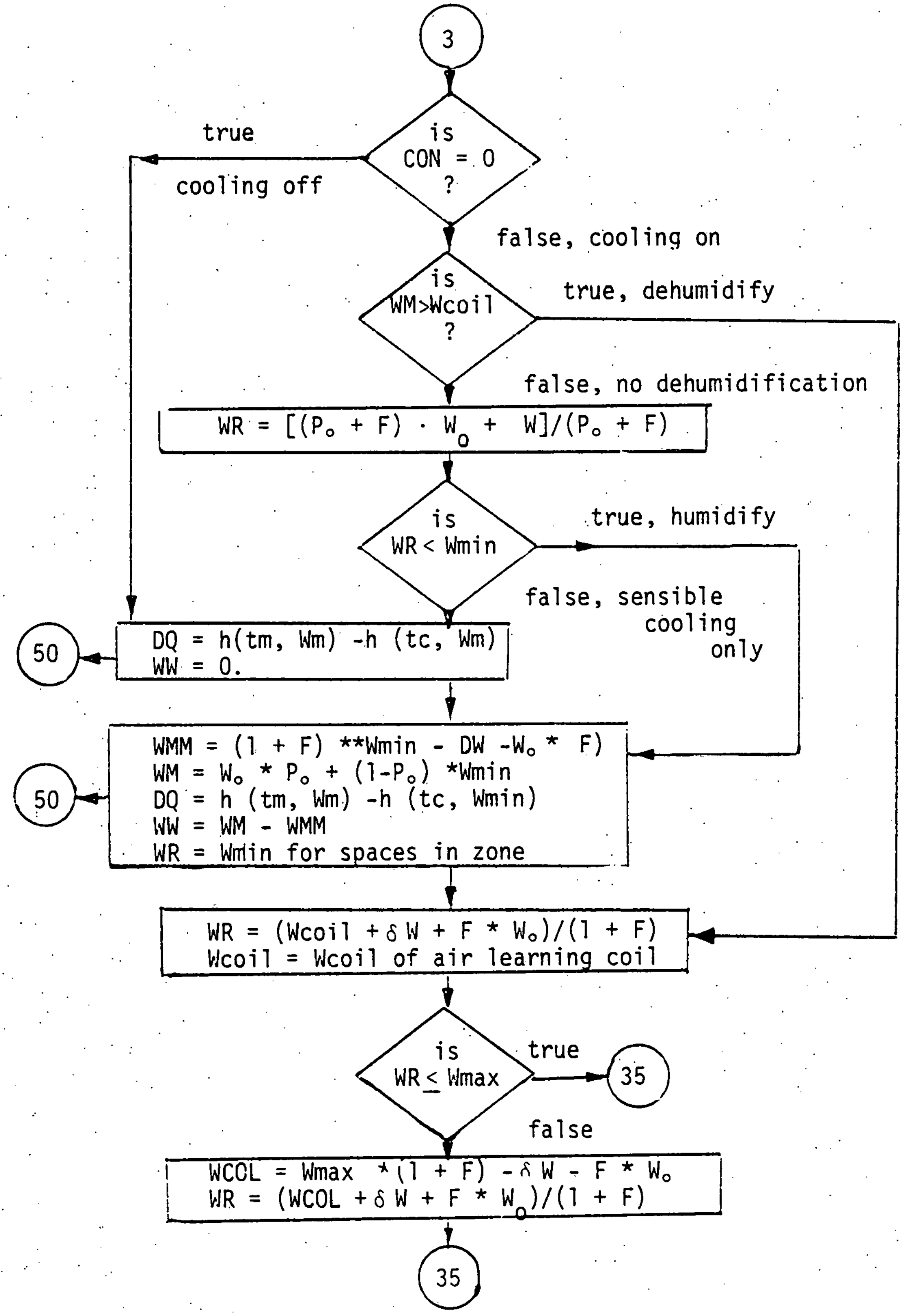
June 15, 1977



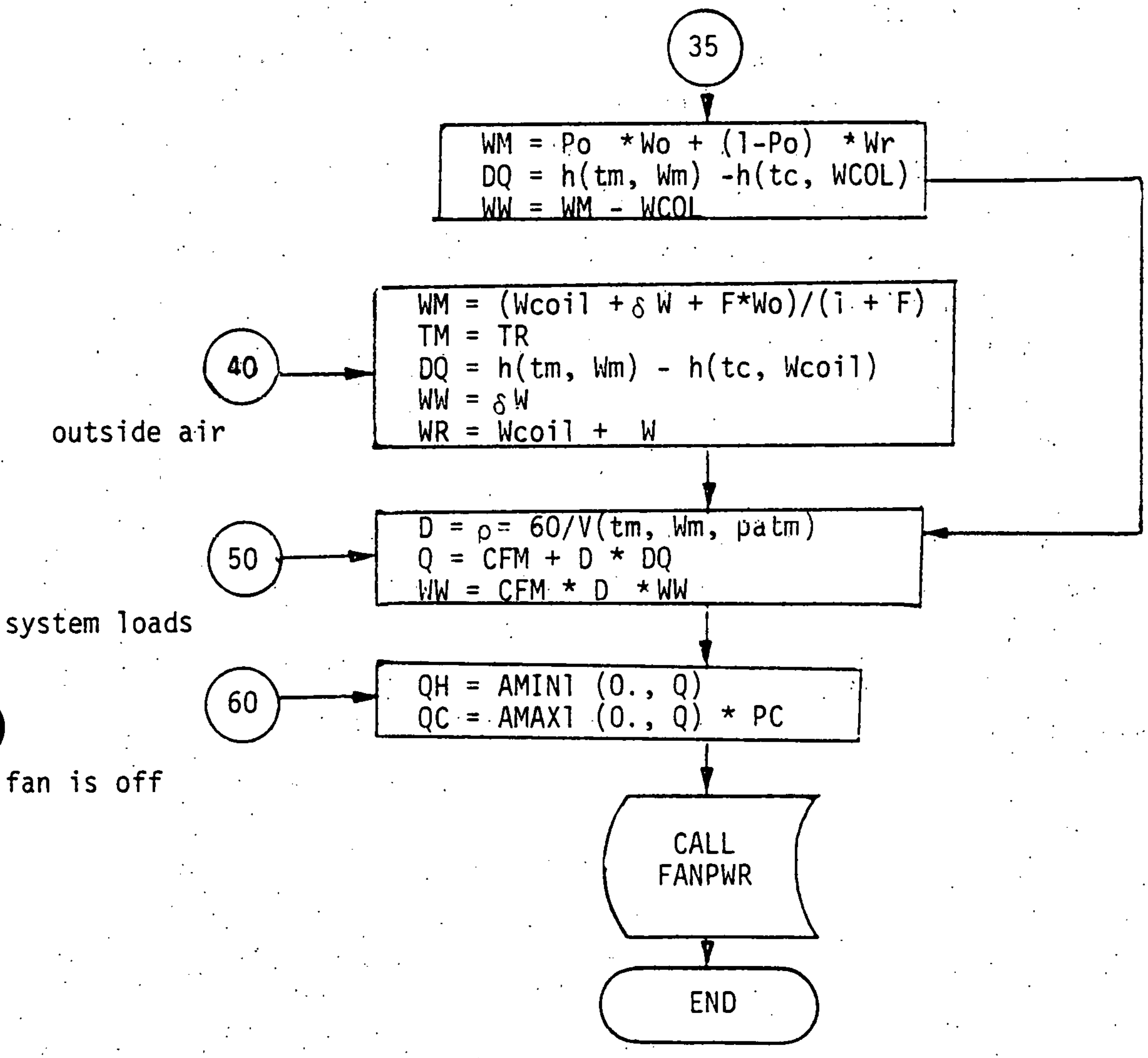
Subroutine SDSF - Continued


2
plenum loop
DO 7
I = 1,3
ZP1 = plenum pointer
is ZP1 = 0 ?
true
no plenum
false
ZP2 = IA(ZP1) + 1
CALL TEMDEV
TRA = [TRAold + QNOW [1.08*ACFM)] *zone multiplier / plenum mult.
return air temp after plenums
TR = TRA
5
is min OA = 0 ?
true
40
no outside air case
false, same O.A.
CALL ECONO
Last hour mixed air temp = TM
F = CINF/CFM = % infiltration
WM = [(1 + F) *P$_o$ * Wo + (1-P$_o$)· (δ W + F·Wo)]/(F+P$_o$)
3

Subroutine SDSF - Continued


3
is CON = 0 ?
true
cooling off
false, cooling on
is WM>Wcoil ?
true, dehumidify
false, no dehumidification
WR = [(P∘ + F) · W0 + W]/(P∘ + F)
is WR < Wmin
true, humidify
false, sensible cooling only
DQ = h(tm, Wm) -h (tc, Wm)
WW = 0.
50
WMM = (1 + F) **Wmin - DW -W∘ * F)
WM = W∘ * P∘ + (1-P∘) *Wmin
DQ = h (tm, Wm) -h (tc, Wmin)
WW = WM - WMM
WR = Wmin for spaces in zone
50
WR = (Wcoil + δ W + F * W∘)/(1 + F)
Wcoil = Wcoil of air learning coil
is WR ≤ Wmax
true
35
false
WCOL = Wmax * (1 + F) - δ W - F * W∘
WR = (WCOL + δ W + F * W0)/(1 + F)
35

Subroutine SDSF - Continued



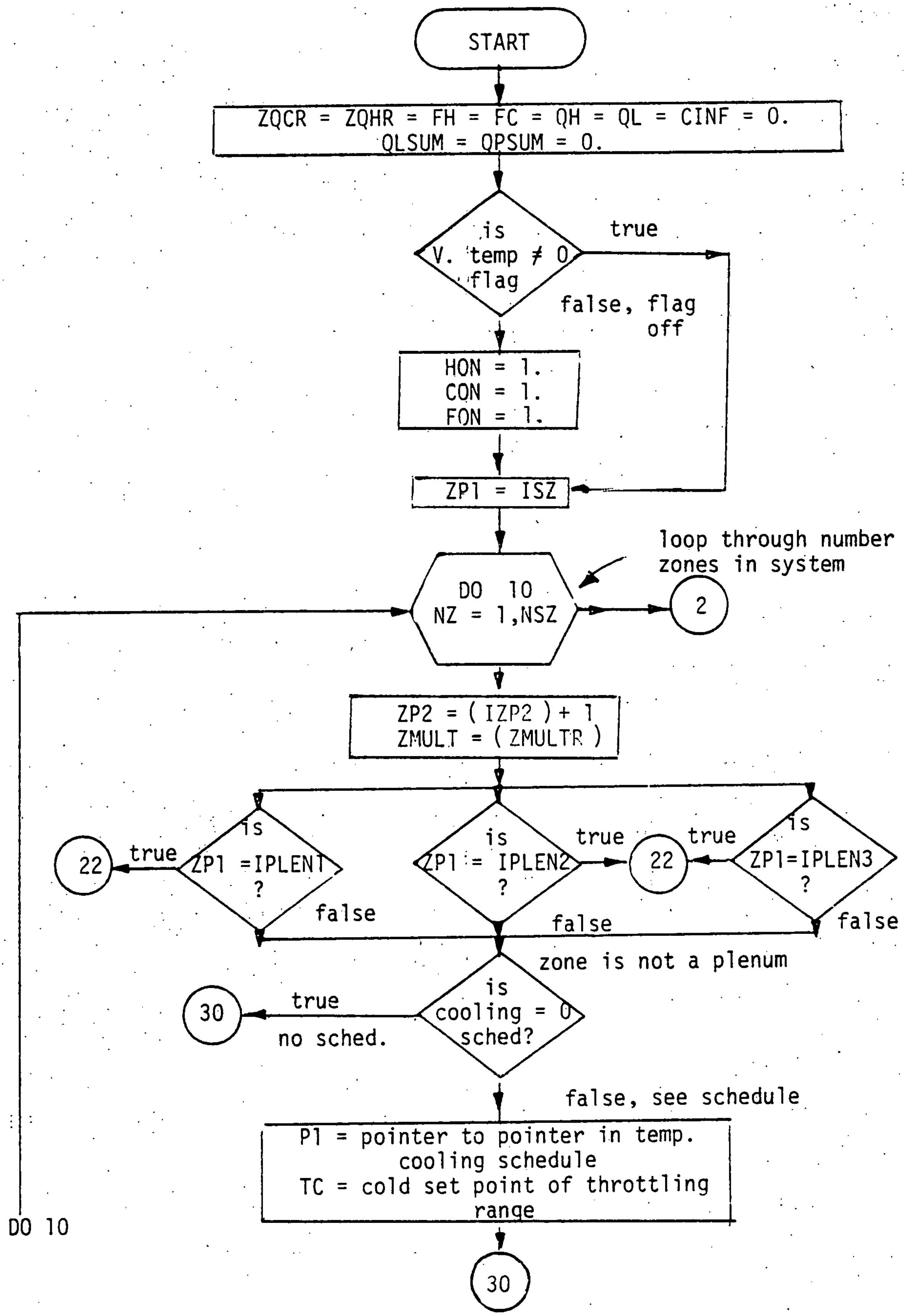
s. Subroutine SUM.

Subroutine SUM is called by the SYSTEMS Program to provide an annual system loads summary report. Parameters and variables used in the subroutine are set to zero. The routine enters a DO loop on all the zones in this system. If a zone is a plenum, the zone number is incremented with no loads summed. If the zone is not a plenum, the hot and cold set points of the throttling range are obtained and subroutine TEMDEV is called. The adjusted zone temperature and load returned from TEMDEV are used in the calculation of zone heating and cooling loads, system heating and cooling loads, and in the sums of latent, plenum, electrical, and infiltration loads.

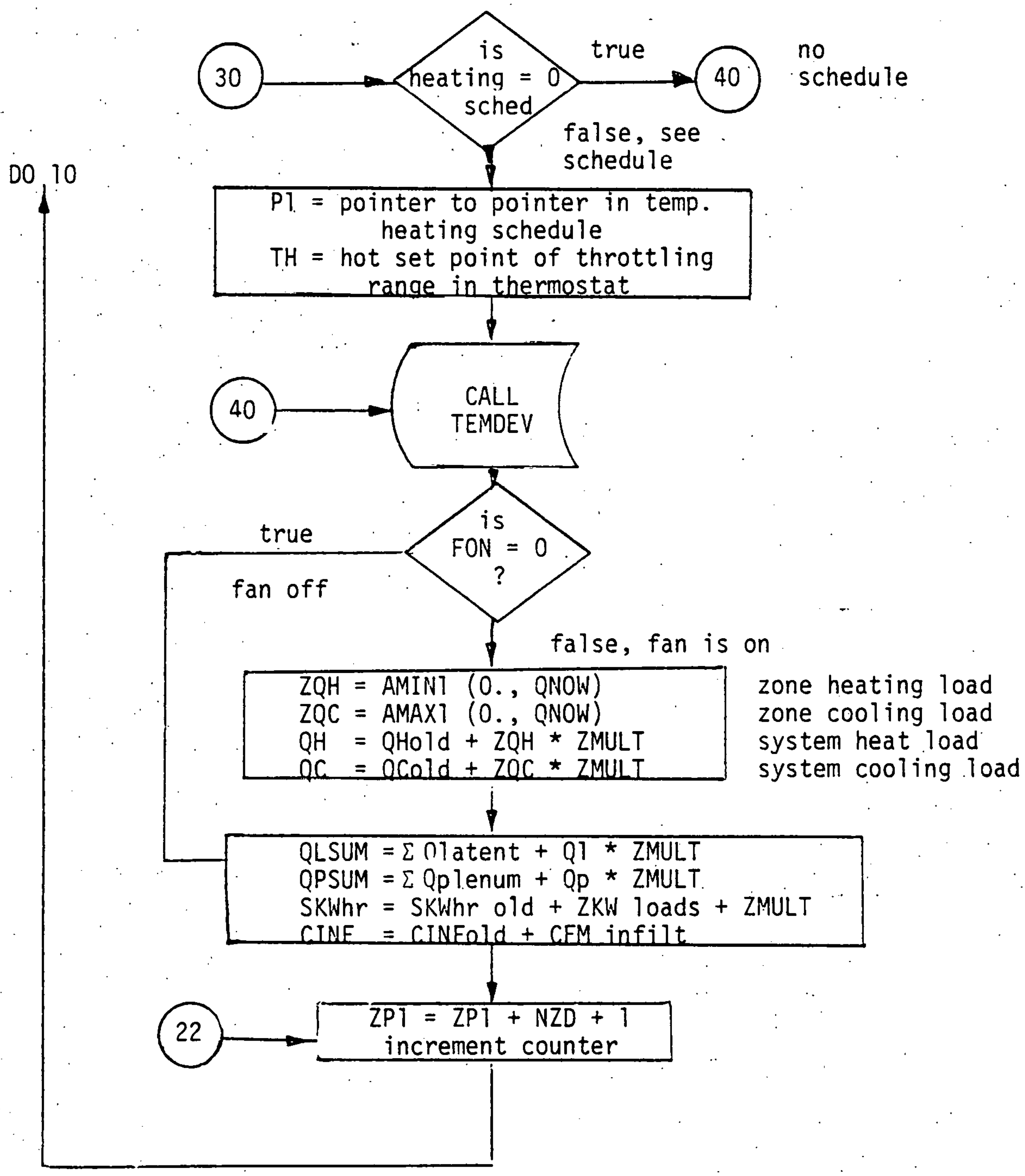
The system heating load is modified by the addition of last hour's total plenum load. A check is made on the sign of the heating load to insure that the summed load is given the correct label. The fan is then turned off.

The subroutine enters a plenum DO loop. Subroutine TEMDEV is called and returns the adjusted temperature and load for use in the next hour's sum of system heating load.

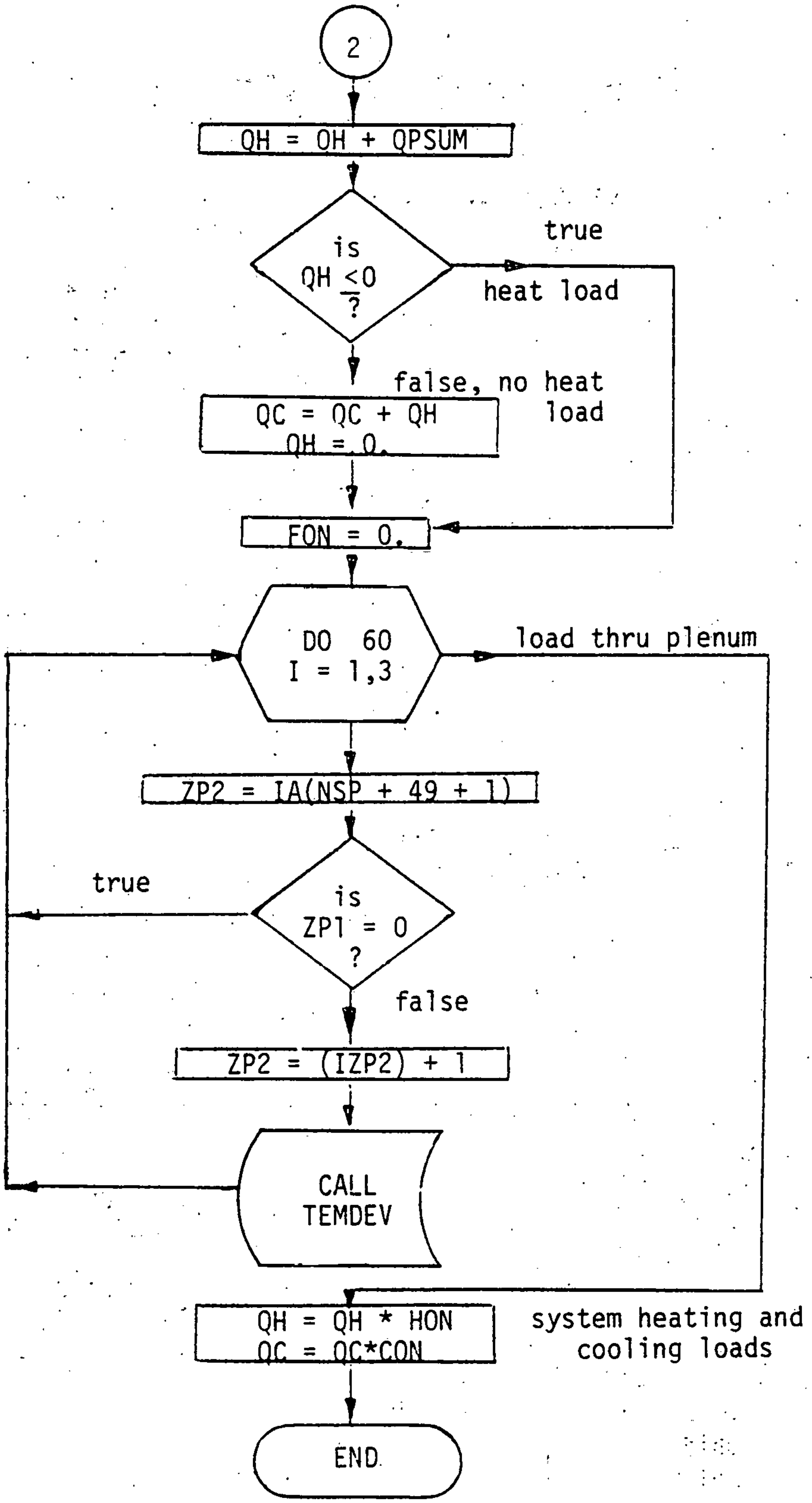
The routine then calculates system heating and cooling loads from the previously calculated values multiplied by the on/off heating and cooling system flags.


START
ZQCR = ZQHR = FH = FC = QH = QL = CINF = 0.
QLSUM = QPSUM = 0.
is V. temp ≠ 0 flag
true
false, flag off
HON = 1.
CON = 1.
FON = 1.
ZP1 = ISZ
loop through number zones in system
DO 10 NZ = 1,NSZ
2
ZP2 = (IZP2) + 1
ZMULT = (ZMULTR)
is ZP1 =IPLEN1 ?
true
22
false
is ZP1 = IPLEN2 ?
true
22
true
is ZP1=IPLEN3 ?
false
false
zone is not a plenum
is cooling = 0 sched?
true
no sched.
30
false, see schedule
P1 = pointer to pointer in temp. cooling schedule
TC = cold set point of throttling range
30
DO 10

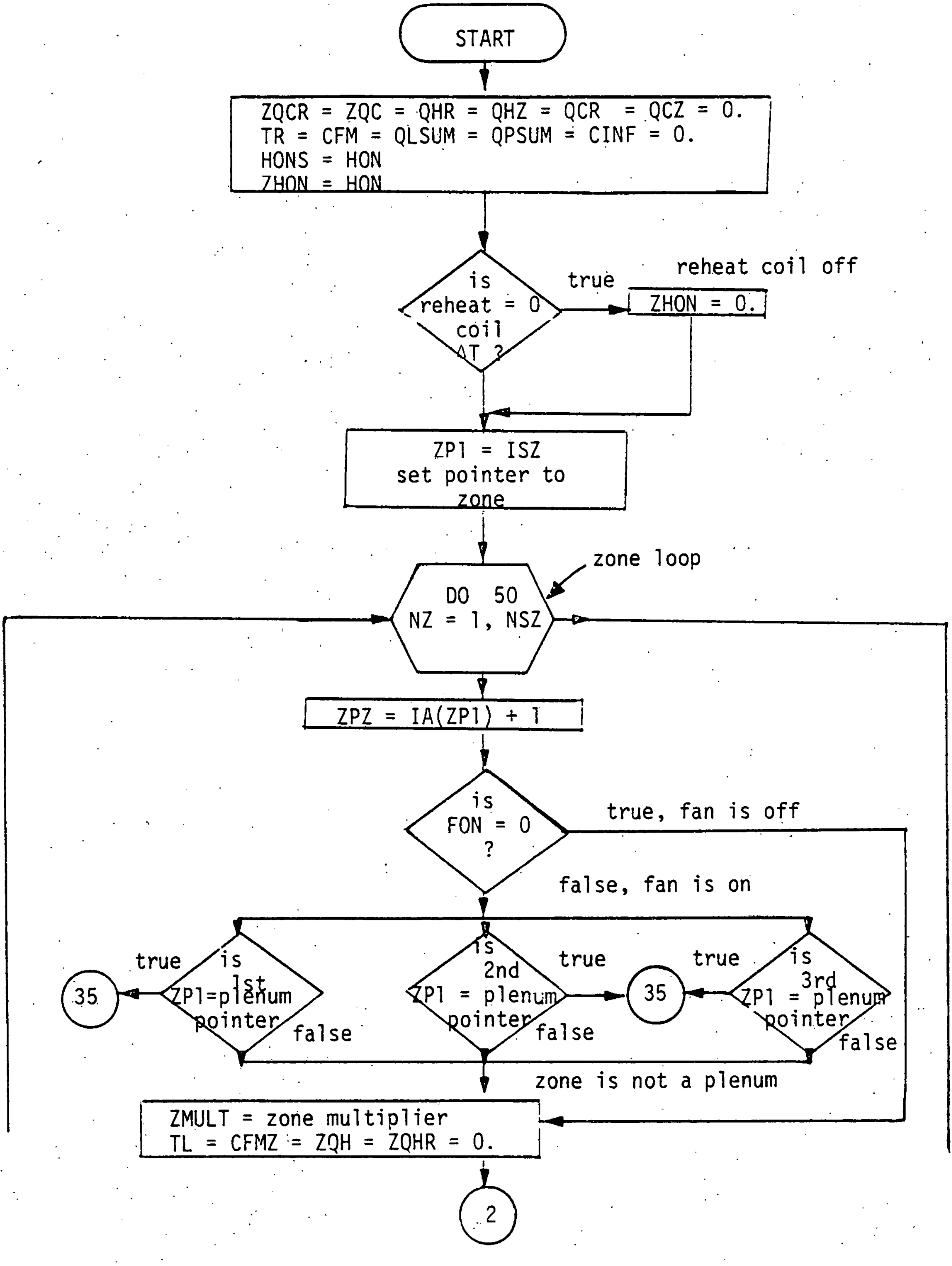
Subroutine SUM - Continued



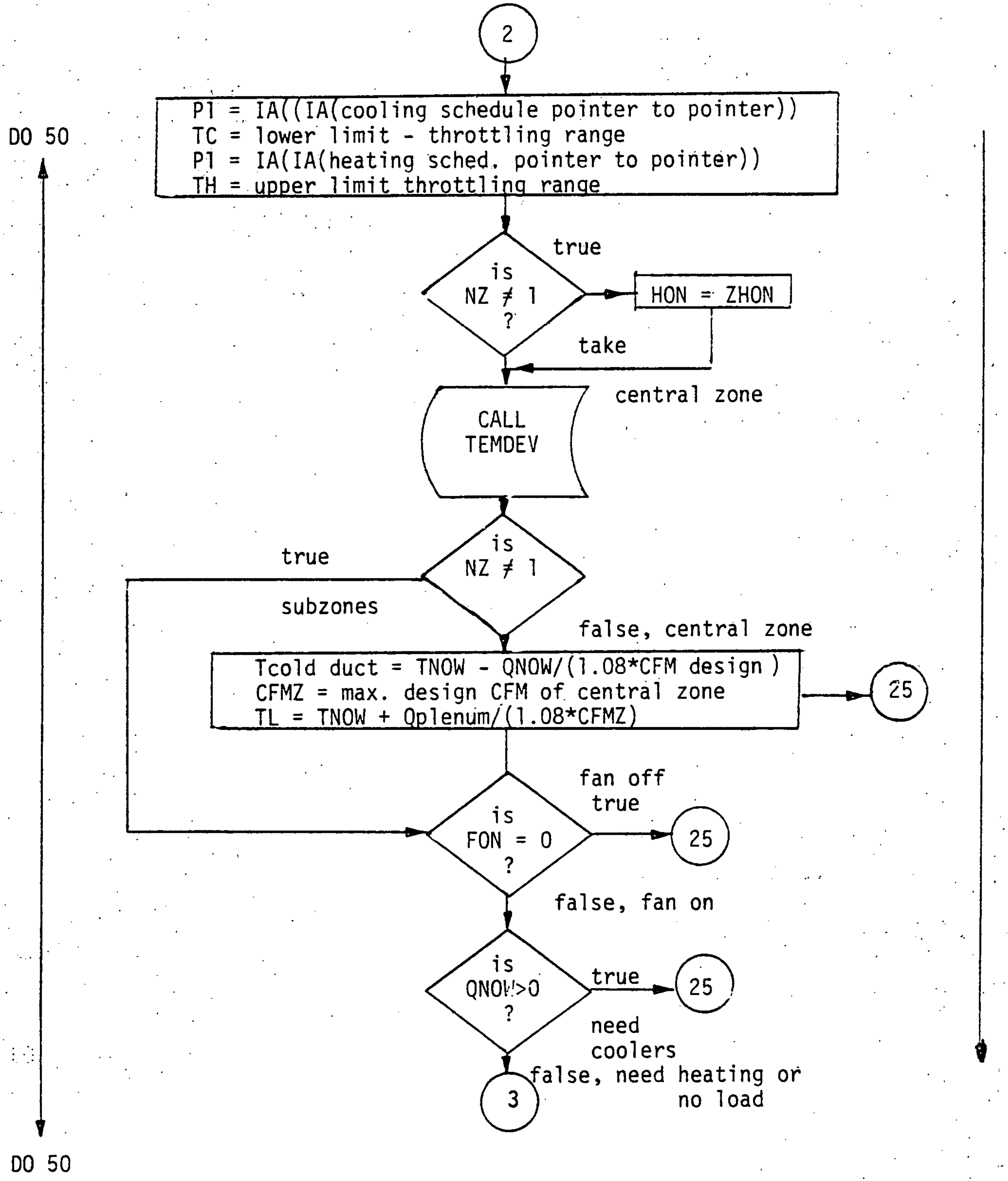
Subroutine SUM - Continued



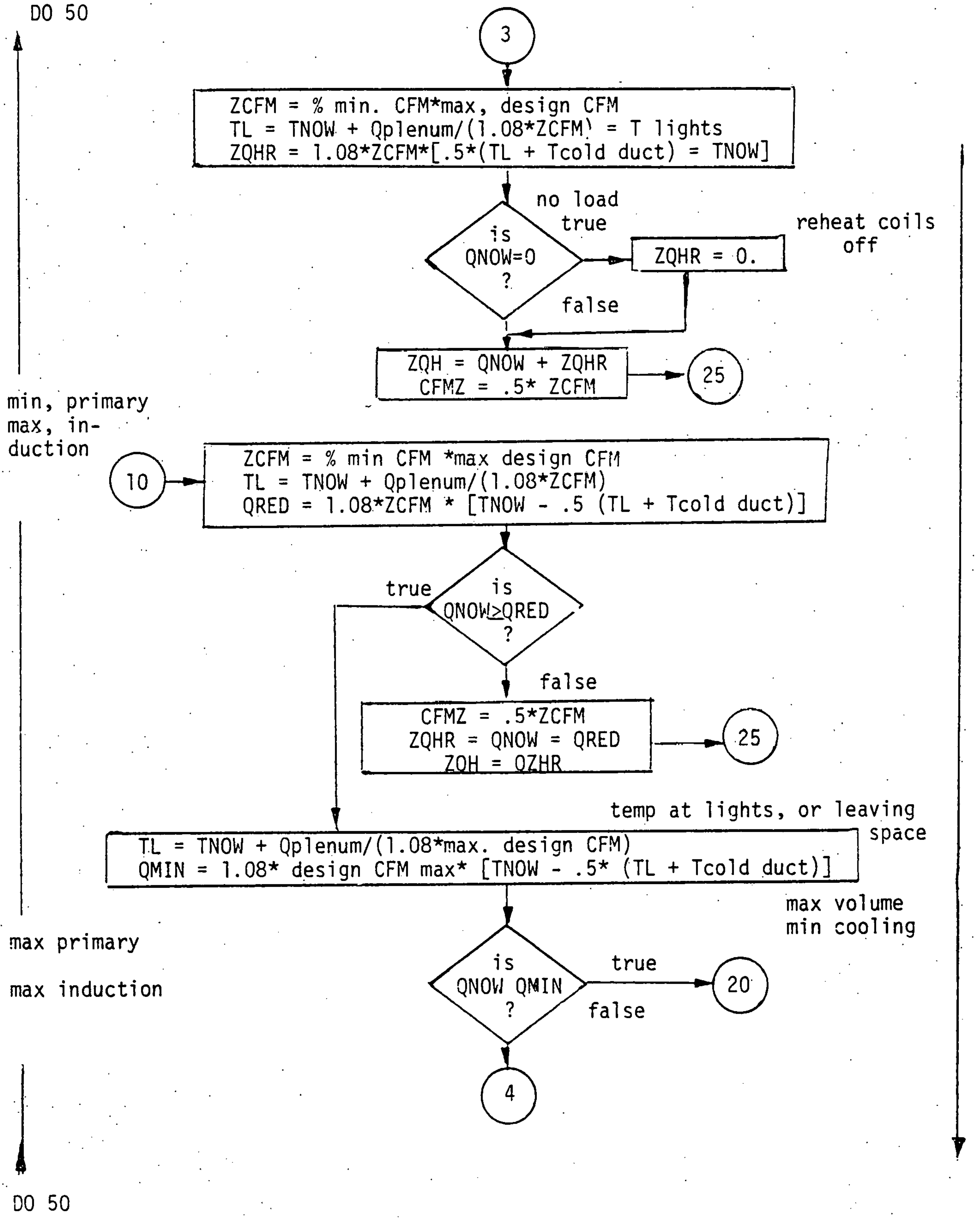
t. Subroutine SZVV.



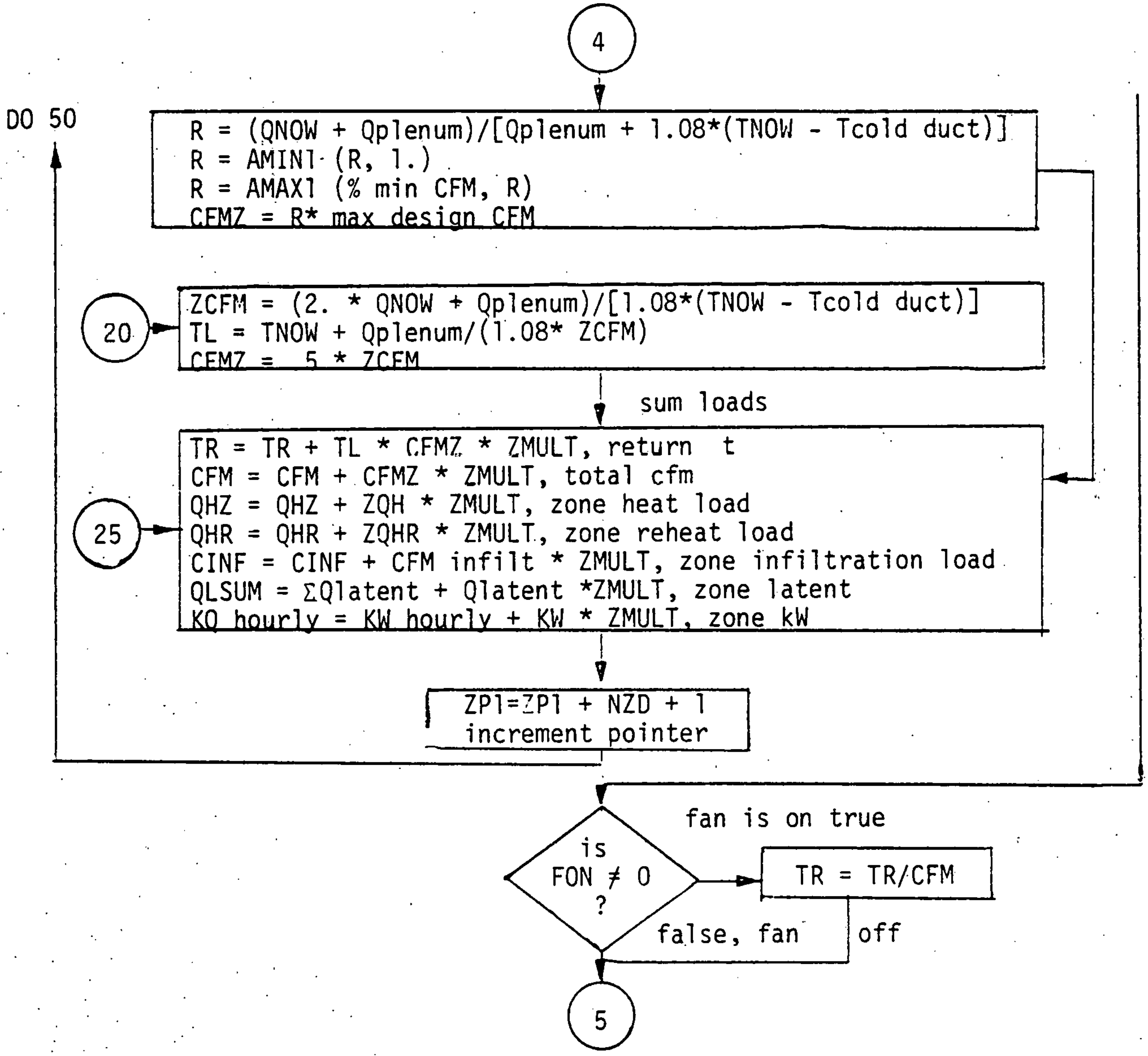
Subroutine SZVV - Continued


2
P1 = IA((IA(cooling schedule pointer to pointer))
TC = lower limit - throttling range
P1 = IA(IA(heating sched. pointer to pointer))
TH = upper limit throttling range
DO 50
is
NZ ≠ 1
?
true
HON = ZHON
take
central zone
CALL
TEMDEV
is
NZ ≠ 1
true
subzones
false, central zone
Tcold duct = TNOW - QNOW/(1.08*CFM design)
CFMZ = max. design CFM of central zone
TL = TNOW + Qplenum/(1.08*CFMZ)
25
is
FON = 0
?
fan off
true
25
false, fan on
is
QNOW>0
?
true
25
need
coolers
false, need heating or
no load
3
DO 50

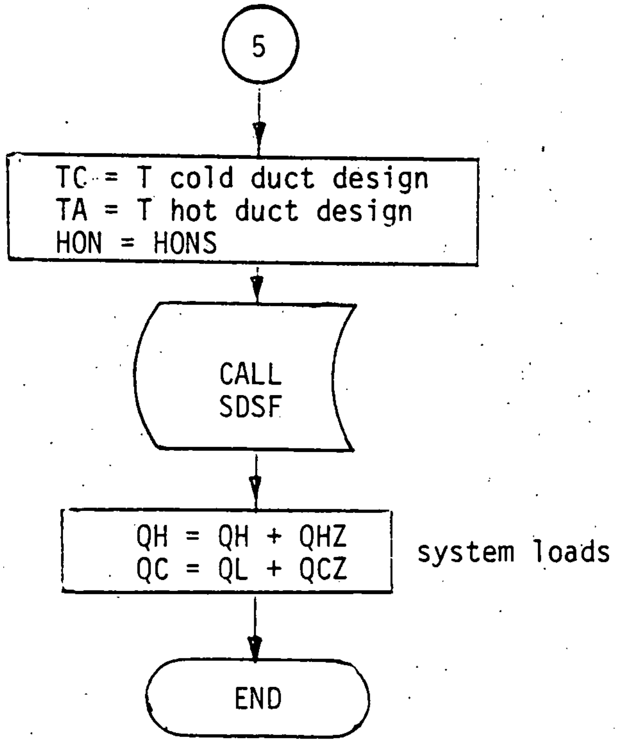
Subroutine SZVV - Continued


DO 50
3
ZCFM = % min. CFM*max, design CFM
TL = TNOW + Qplenum/(1.08*ZCFM) = T lights
ZQHR = 1.08*ZCFM*[.5*(TL + Tcold duct) = TNOW]
no load
true
is
QNOW=0
?
ZQHR = 0.
reheat coils
off
false
ZQH = QNOW + ZQHR
CFMZ = .5* ZCFM
25
min, primary
max, induction
10
ZCFM = % min CFM *max design CFM
TL = TNOW + Qplenum/(1.08*ZCFM)
QRED = 1.08*ZCFM * [TNOW - .5 (TL + Tcold duct)]
true
is
QNOW≥QRED
?
false
CFMZ = .5*ZCFM
ZQHR = QNOW = QRED
ZOH = OZHR
25
temp at lights, or leaving
space
TL = TNOW + Qplenum/(1.08*max. design CFM)
QMIN = 1.08* design CFM max* [TNOW - .5* (TL + Tcold duct)]
max volume
min cooling
max primary
max induction
is
QNOW QMIN
?
true
false
20
4
DO 50

Subroutine SZVV - Continued

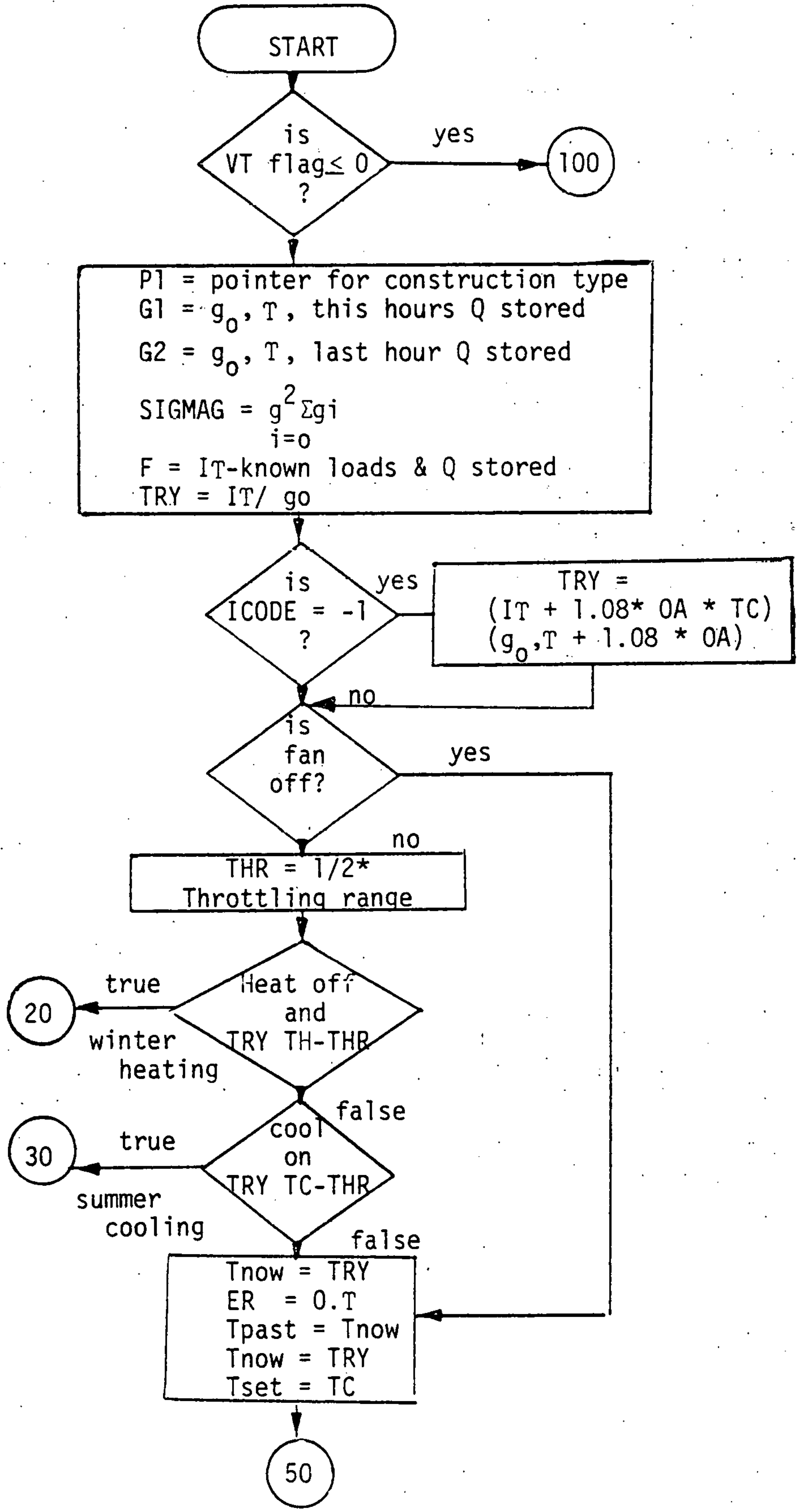

4
DO 50
R = (QNOW + Qplenum)/[Qplenum + 1.08*(TNOW - Tcold duct)]
R = AMIN1 (R, 1.)
R = AMAX1 (% min CFM, R)
CFMZ = R* max design CFM
20
ZCFM = (2. * QNOW + Qplenum)/[1.08*(TNOW - Tcold duct)]
TL = TNOW + Qplenum/(1.08* ZCFM)
CFMZ = .5 * ZCFM
sum loads
25
TR = TR + TL * CFMZ * ZMULT, return t
CFM = CFM + CFMZ * ZMULT, total cfm
QHZ = QHZ + ZQH * ZMULT, zone heat load
QHR = QHR + ZQHR * ZMULT, zone reheat load
CINF = CINF + CFM infilt * ZMULT, zone infiltration load
QLSUM = ΣQlatent + Qlatent *ZMULT, zone latent
KQ hourly = KW hourly + KW * ZMULT, zone kW
ZP1=ZP1 + NZD + 1
increment pointer
is
FON ≠ 0
?
fan is on true
TR = TR/CFM
false, fan off
5

Subroutine SZVV - Continued

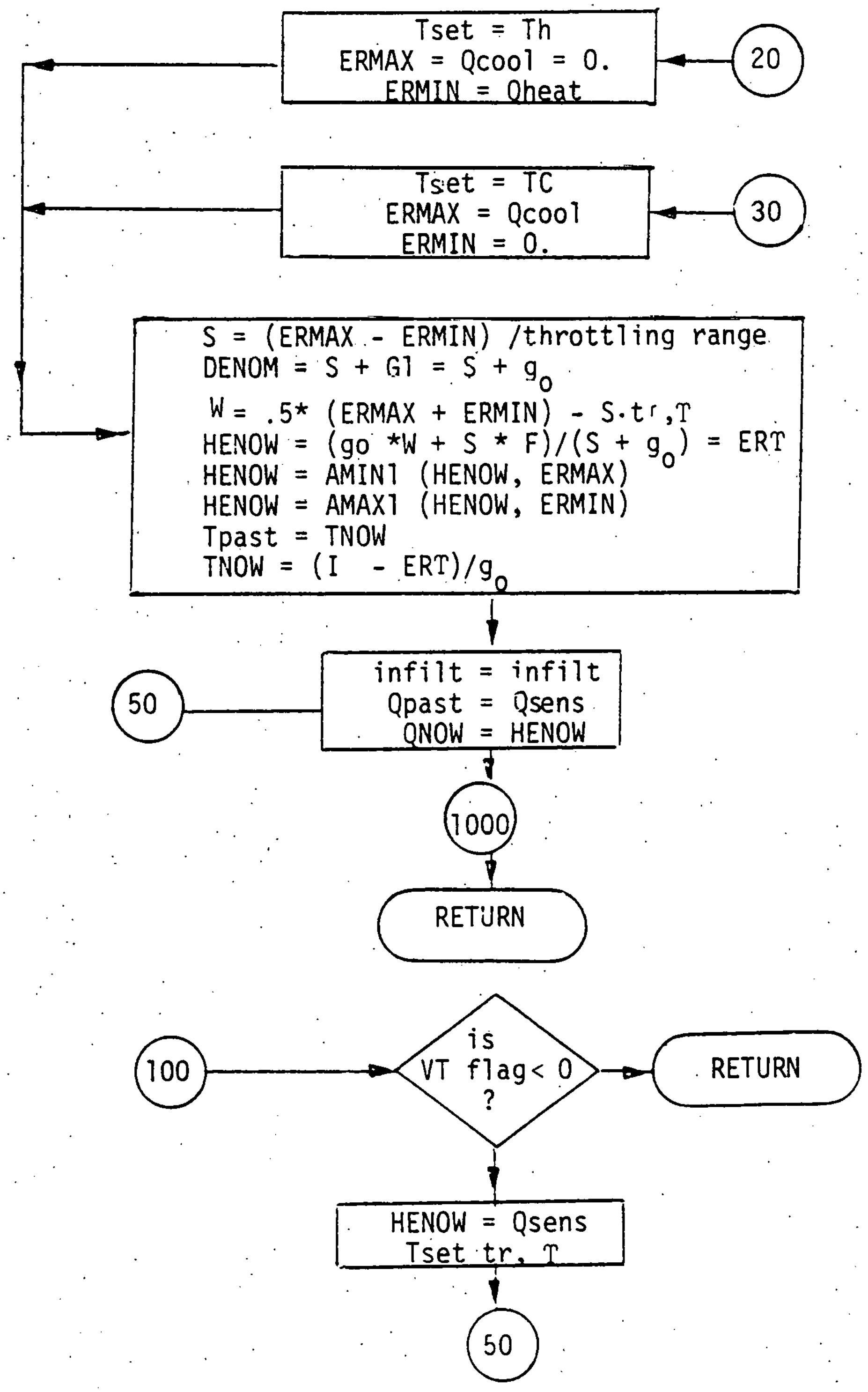


u. Subroutine TEMDEV.

April 7, 1977

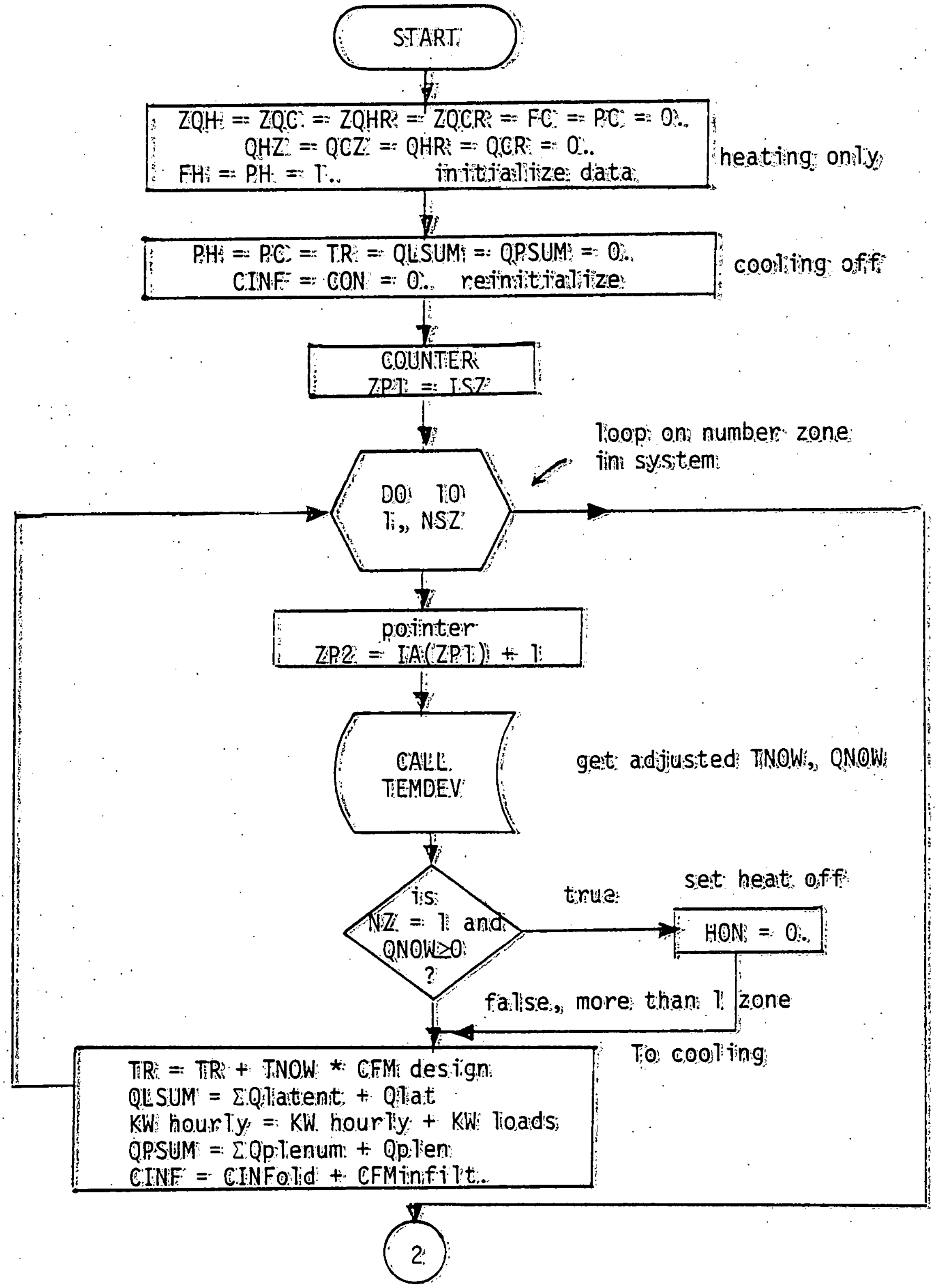


Subroutine TEMDEV - Continued

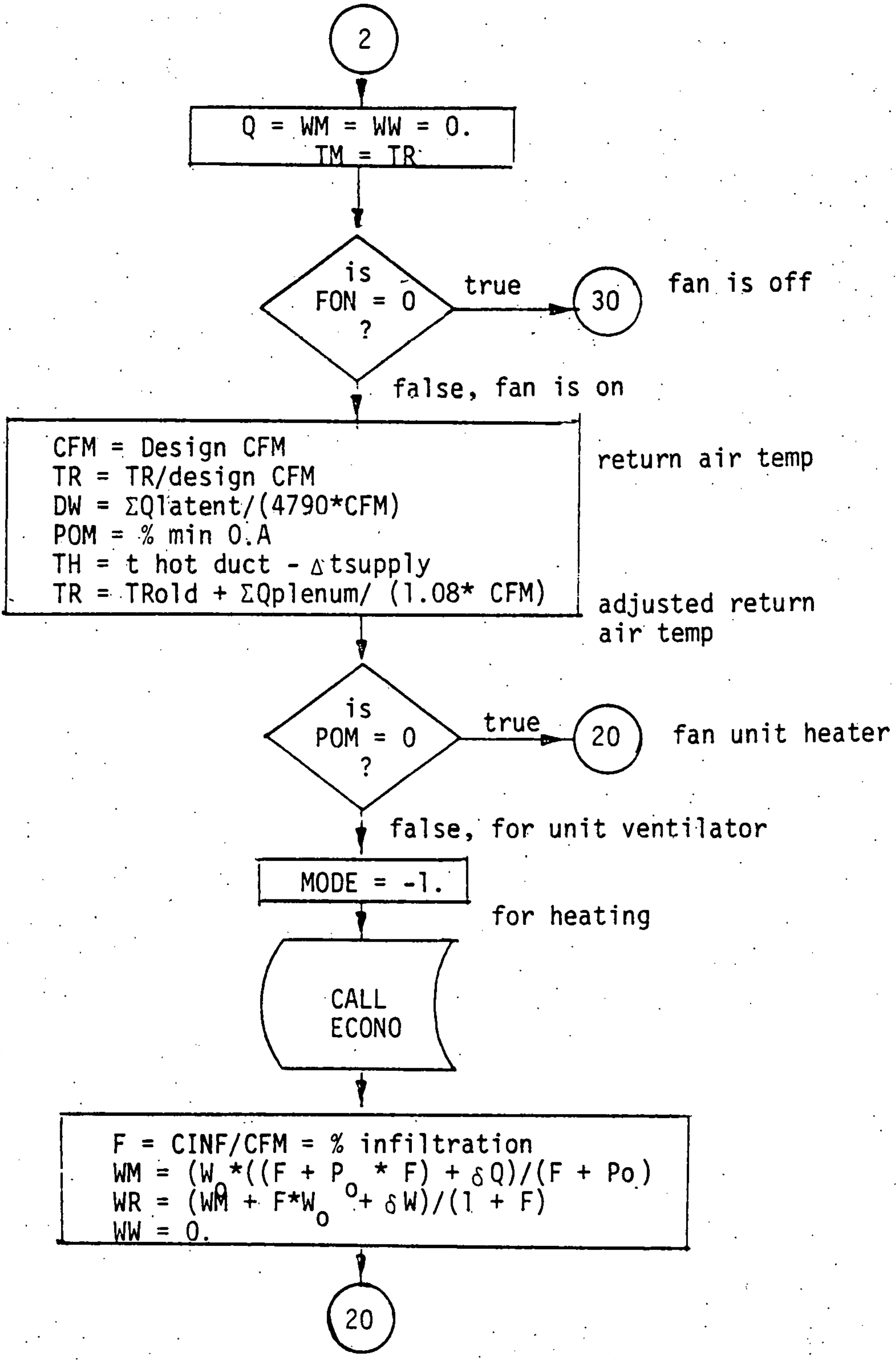


V. Subroutine UNITHV

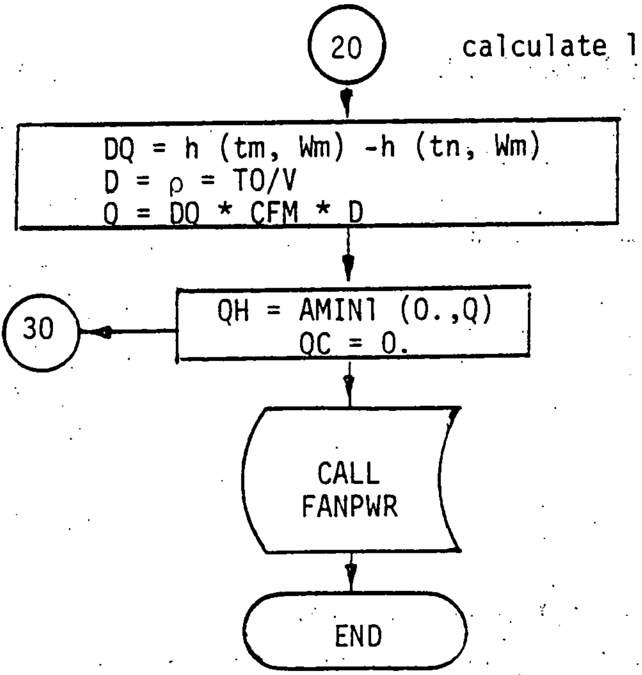
May 5, 1977


START
ZQH = ZQC = ZQHR = ZQCR = FC = PC = 0.
QHZ = QCZ = QHR = QCR = 0.
FH = PH = 1. initialize data
heating only
PH = PC = TR = QLSUM = QPSUM = 0.
CINF = CON = 0. reinitialize
cooling off
COUNTER
ZP1 = ISZ
loop on number zone in system
DO 10
1, NSZ
pointer
ZP2 = IA(ZP1) + 1
CALL
TEMDEV
get adjusted TNOW, QNOW
is
NZ = 1 and
QNOW≥0
?
true
set heat off
HON = 0.
false, more than 1 zone
To cooling
TR = TR + TNOW * CFM design
QLSUM = ΣQlatent + Qlat
KW hourly = KW hourly + KW loads
QPSUM = ΣQplenum + Qplen
CINF = CINFold + CFMinfilt.
2

Subroutine UNITHV - Continued



Subroutine UNITHV - Continued



calculate loads hot deck temp.

w. Function V.

Description

The function subprogram V calculates the specific volume of moist air in ft^3/lb of dry air.

Subprograms Calling This Function

SUBROUTINE DDSF
SUBROUTINE SDSF
SUBROUTINE UNITHV

Subprogram Called by This Function

None

Common Block

None

Declarations

None

Input

Name	Description
DB	Dry-bulb temperature (°F)
W	Humidity ratio of moist air (lb H_2O/lb dry air)
P	Barometric pressure (in. Hg)

Output

Name	Description
V	Specific volume of moist air (ft^3/lb dry air)

Calculation Procedure

Compute V using DB, W, and P.

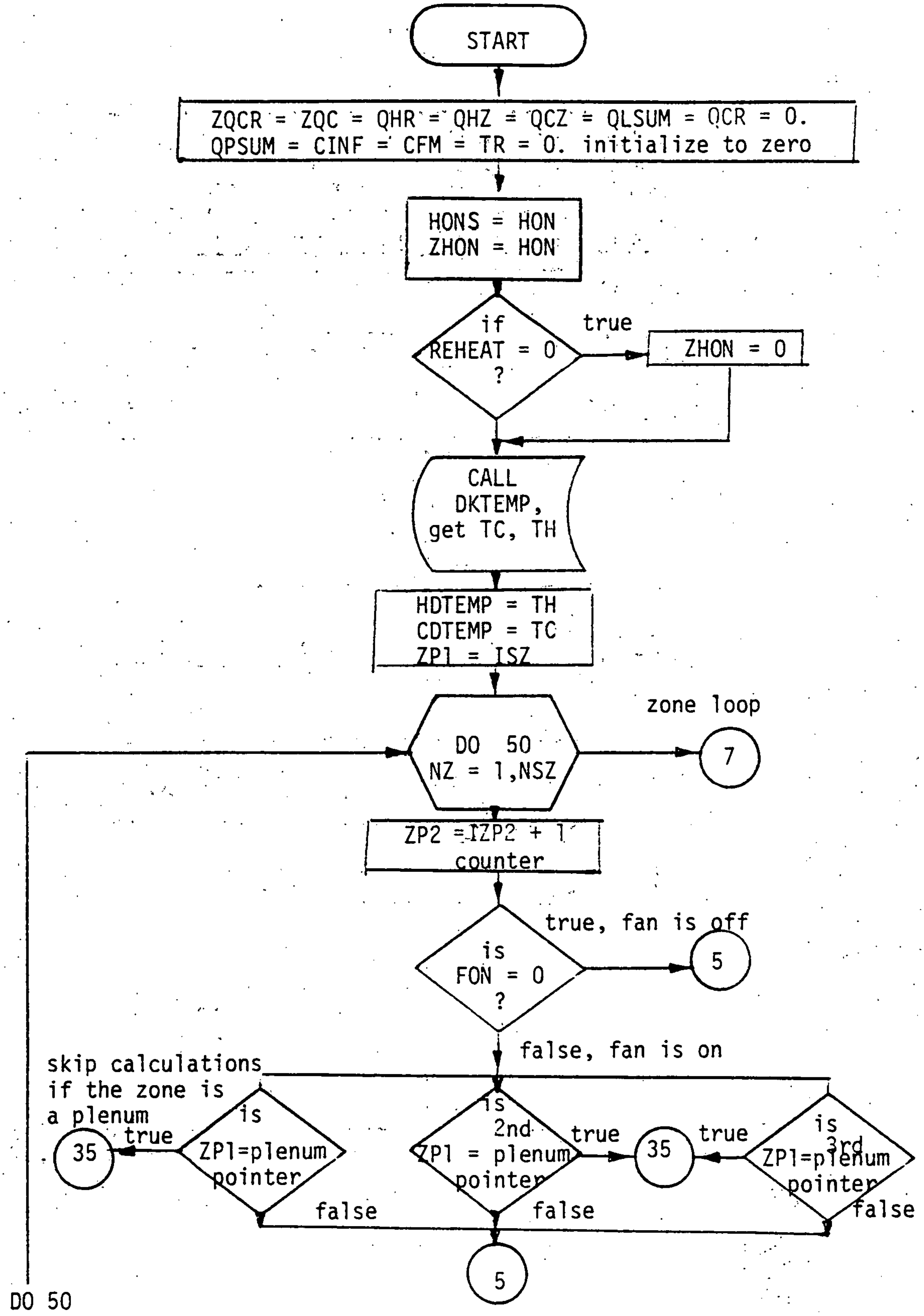
```
V = 0.754 * (DB+459.7) * (1.0+1.605+W) /P
```

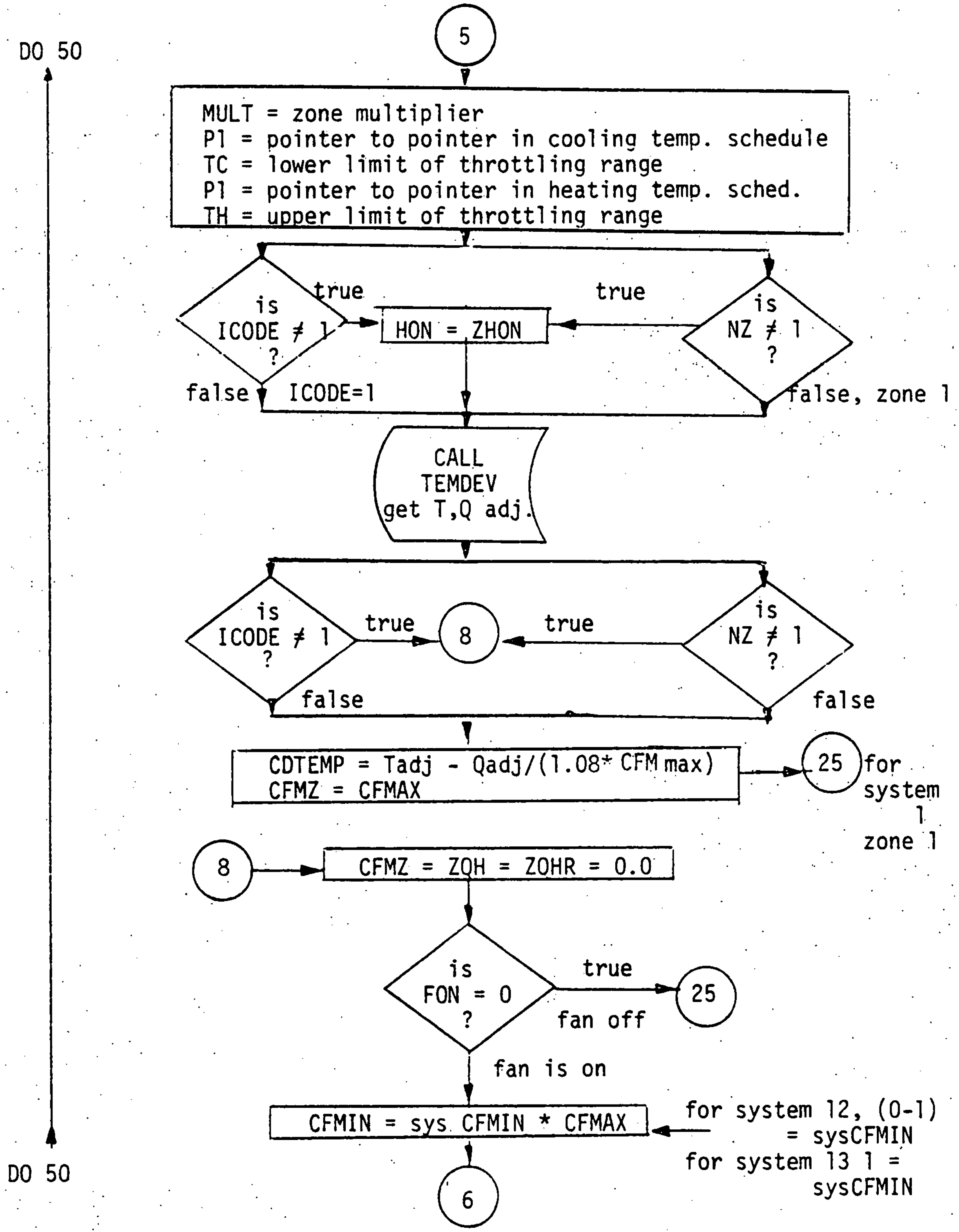
ASHRAE Verification

The equation used in the V function subprogram by the Cal-ERDA code is the same as the V equation in ASHRAE (Ref. 1).

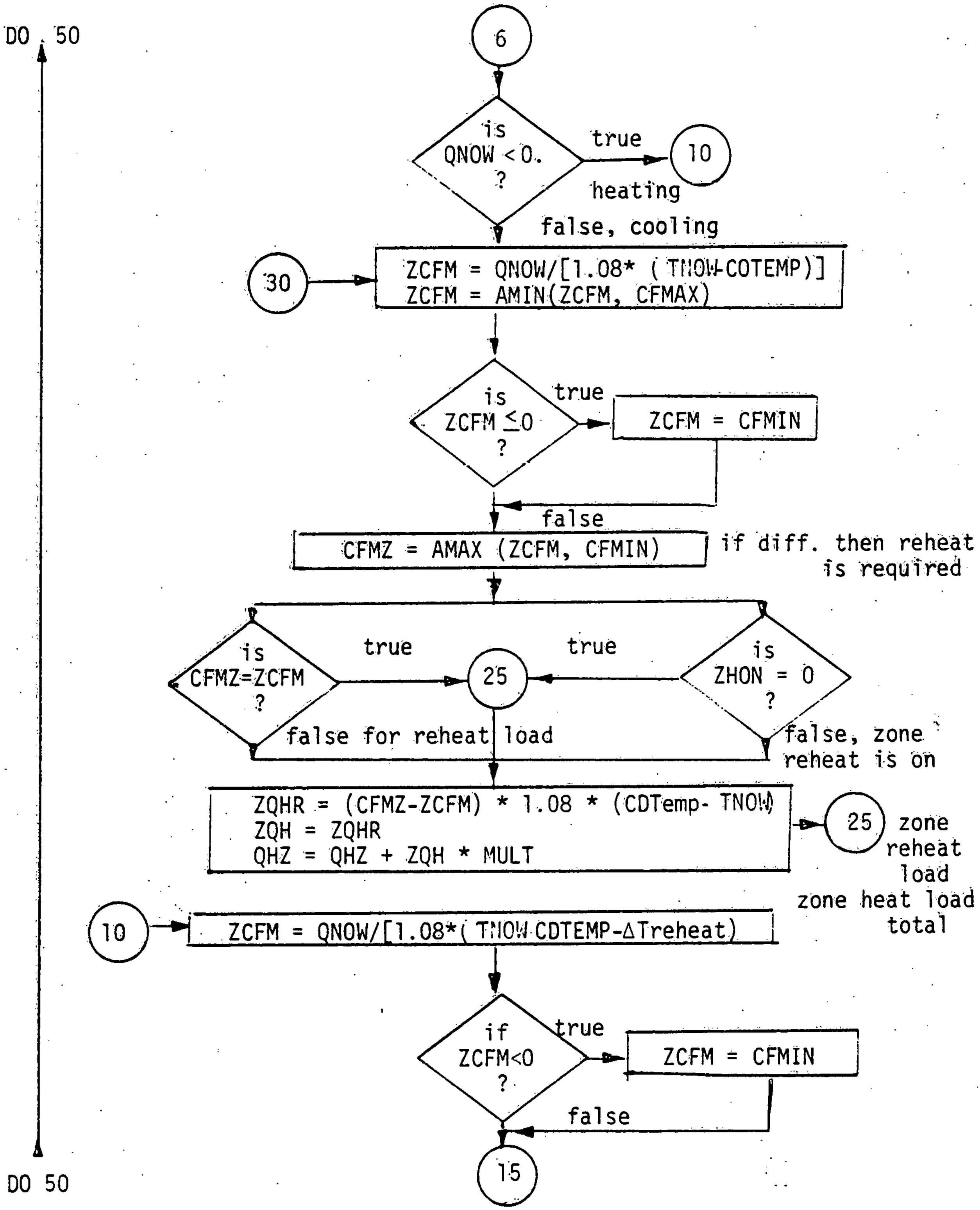
x. Subroutine VARVOL.

June 15, 1977

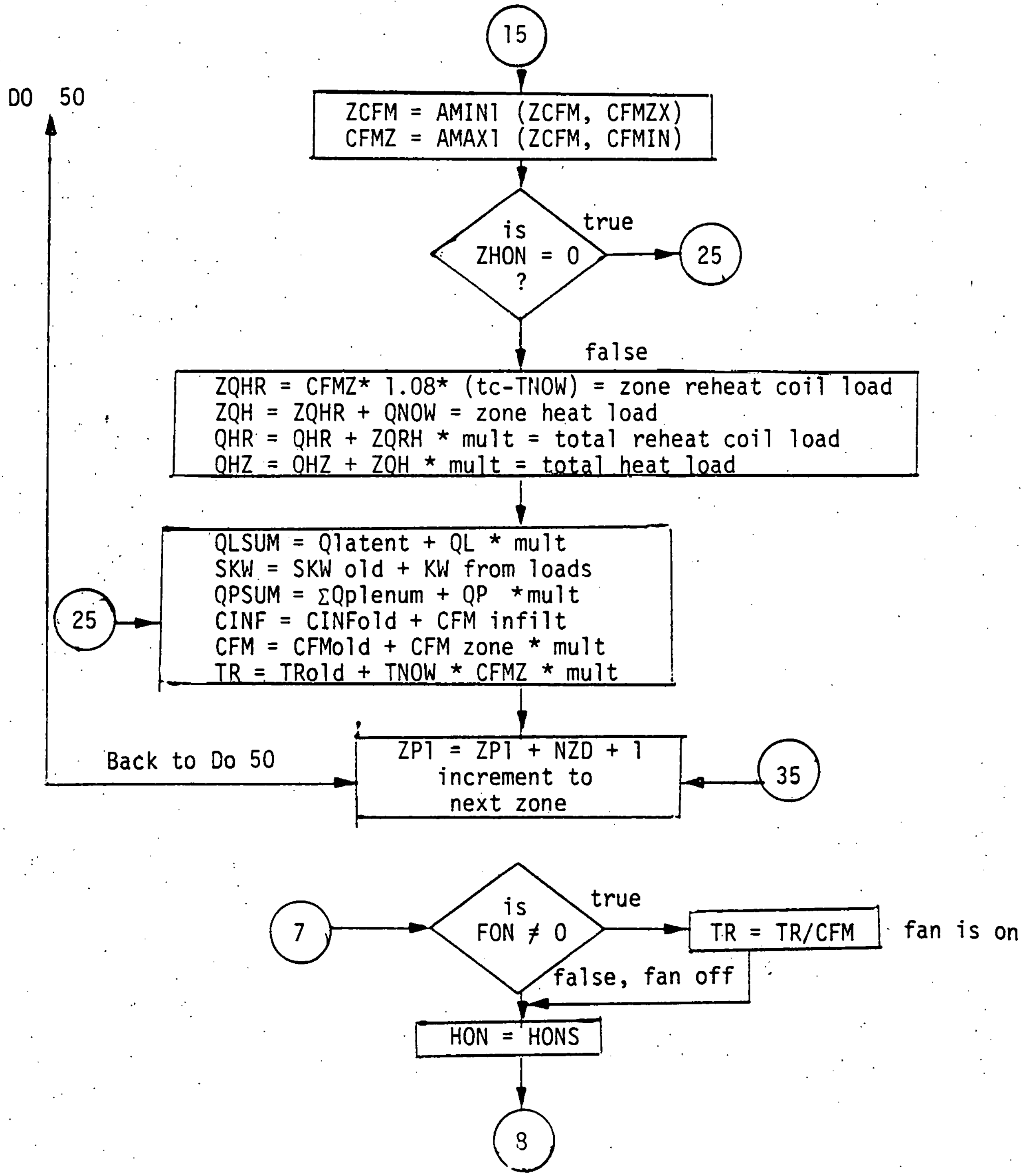



5
DO 50
MULT = zone multiplier
P1 = pointer to pointer in cooling temp. schedule
TC = lower limit of throttling range
P1 = pointer to pointer in heating temp. sched.
TH = upper limit of throttling range
is ICODE ≠ 1 ?
true
HON = ZHON
true
is NZ ≠ 1 ?
false
ICODE=1
false, zone 1
CALL TEMDEV get T,Q adj.
is ICODE ≠ 1 ?
true
8
true
is NZ ≠ 1 ?
false
false
CDTEMP = Tadj - Qadj/(1.08* CFM max)
CFMZ = CFMAX
25
for system 1 zone 1
8
CFMZ = ZOH = ZOHR = 0.0
is FON = 0 ?
true
fan off
25
fan is on
CFMIN = sys CFMIN * CFMAX
for system 12, (0-1) = sysCFMIN
for system 13 1 = sysCFMIN
6
DO 50

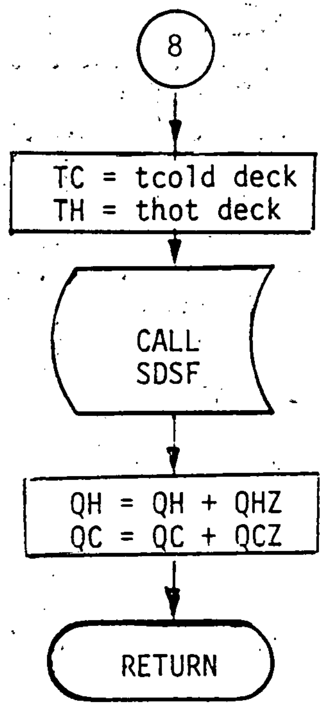
Subroutine VARVOL - Continued



Subroutine VARVOL - Continued


15
DO 50
ZCFM = AMIN1 (ZCFM, CFMZX)
CFMZ = AMAX1 (ZCFM, CFMIN)
is
ZHON = 0
?
true
25
false
ZQHR = CFMZ* 1.08* (tc-TNOW) = zone reheat coil load
ZQH = ZQHR + QNOW = zone heat load
QHR = QHR + ZQRH * mult = total reheat coil load
QHZ = QHZ + ZQH * mult = total heat load
25
QLSUM = Qlatent + QL * mult
SKW = SKW old + KW from loads
QPSUM = ΣQplenum + QP *mult
CINF = CINFold + CFM infilt
CFM = CFMold + CFM zone * mult
TR = TRold + TNOW * CFMZ * mult
Back to Do 50
ZP1 = ZP1 + NZD + 1
increment to
next zone
35
7
is
FON ≠ 0
true
TR = TR/CFM
fan is on
false, fan off
HON = HONS
8

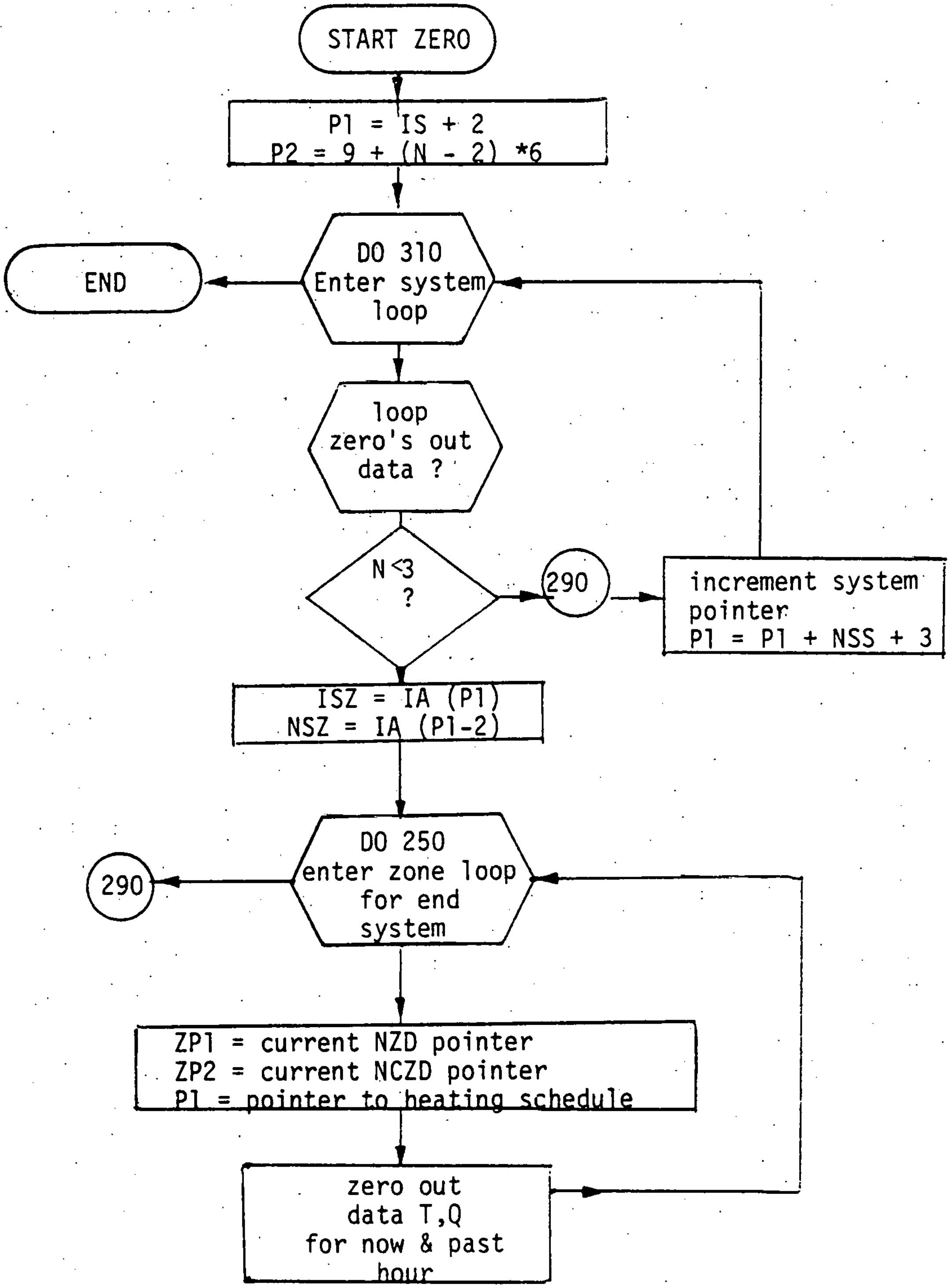
Subroutine VARVOL - Continued



reset

calculate heating and cooling coil loads

y. Subroutine ZERO. April 19, 1977



5. System Constants.

April 19, 1977

CONS

1	2	3	4	5
1.08	4790.	0.372	0.	0.

CRATF

light					medium					heavy
1	2	3	4	5	6	7	8	9	10	11
1.68	-1.73	.05	1.	-.82	1.81	-1.89	.08	1.0	-.87	1.85

g_1^* g_o^* g_2^* p_o p_1

12	13	14	15
-1.95	0.1	1.0	-.93

MO

1	2	3	4	5	6	7	8	9	10	11	12
31	28	31	30	31	30	31	31	30	31	30	31

1 code = IA(NSP + 1) = number of system type

V. PLANT PROGRAM

A. Objective and Description

The PLANT Program is used to simulate the equipment that uses fuel (e.g., oil, gas, or sunshine) to provide heating, cooling, process steam, and electricity. The program requires user input data to specify the type, size, control conditions, and operating parameters of the plant equipment, and it requires an input file of building thermal energy loads. This file is generated by the SYSTEMS Program. The plant equipment may include on-site electric generating and heat recovery capabilities that can satisfy coincident electrical, heating, and cooling demands of a building.

This program uses hourly demands for a typical year to calculate the energy consumption on an hourly basis and summarizes it by month and year. Conventional central plant equipment (e.g., boilers and chillers) or selective energy plants (a hybrid configuration having utility electricity and on-site electric generation capability) can be simulated using this program. Plant equipment having seasonal or temporal changes of components can also be simulated using this program. This is done by running the program multiple times and manually selecting the results to obtain a composite that represents a full year of operation.

Additional plant equipment models have been added to the program. A selection of mathematical models for the performance of several versions of flat-plate solar collectors (air and water) is included. A model to simulate a liquid or air collector, a controller, storage, auxiliary heating and cooling strategy, and insolation is also included. Table V.1 lists the types of equipment that can be simulated by the PLANT Program.

B. Program Notes

1. Optimization. Since the PLANT Program was developed as a design tool for total energy plants, it contains algorithms that allocate available equipment so that the total fuel input energy is minimized. This is subject to procedures that are discussed in the individual subroutine descriptions. All PLANT optimization routines use linear extrapolations and approximations. The user should be aware that these procedures make several key assumptions.

a. The rule employed to determine which equipment type within a generic class is operated is a precedence rule based purely on equipment

TABLE V.1
EQUIPMENT TYPE CODES

Index	Code	Equipment
1	GTURB	Gas turbine
2	DIESL	Diesel engine
4	STMB	Steam boiler
5	ABS1	One-stage absorption chiller
6	ABS2	Two-stage absorption chiller
7	ABS2E	Two-stage absorption chiller with economizer
9	COMPH	Hermetic compression chiller
10	COMPC	Open centrifugal compression chiller
11	COMPR	Reciprocating compression chiller
13	DBUN	Double-bundle chiller
14	CTOWR	Cooling tower
15	CTOWC	Ceramic cooling tower
17	STURB	Steam turbine
18	HTANK	Hot water tank
19	CTANK	Cold water tank

types. Thus, a two-stage absorption chiller with economizer always excludes all one-stage absorption chillers; a reciprocating compression chiller always excludes all open centrifugal and hermetic compression chillers; and a ceramic cooling tower always excludes all nonceramic cooling towers (see Table V.2). In this fashion, the program allows only one type of absorption chiller, one type of compression chiller (other than double-bundle chillers), and one type of cooling tower to be placed in operation during the simulation. If more than one type in a generic class is specified, certain types will not be operated.

b. A sophisticated heat recovery procedure that matches waste heat available from operating equipment (prime movers, steam generators, double-bundle chillers, etc.) to the demand for thermal energy at various temperature levels is employed. Five temperature ranges are used.

This procedure simulates a series of heat exchangers and the appropriate hardware and control equipment to accomplish this heat exchange is assumed.

TABLE V.2

EQUIPMENT PRECEDENCE RULES

Precedence Rule	Equipment Type		
	Absorption Chillers	Compression Chillers	Cooling Towers
Excludes all others	1) Two-stage absorption with economizer	1) Reciprocating compression	1) Ceramic
Excludes those below	2) Two-stage absorption 3) One-stage absorption	2) Open centrifugal or hermetic compression	2) Nonceramic

c. Allocation of cooling loads among the generic classes of chillers (absorption, compression, and double bundle) and electrical loads among the generic classes of prime movers (diesel, gas turbines, and steam turbines) is done so that waste heat from all operating equipment is used whenever possible. Thus, plant input energy is minimized. Some approximations are made in the chiller allocation to avoid iterative calculations, and average energy consumption figures are used for both cooling and electrical equipment.

These allocations schemes assume not only the existence of appropriate heat exchangers, piping, and valves, but also the control hardware required to accomplish this control logic.

d. The energy storage subroutine employs both a hot and chilled water storage tank with negligible heat losses. These tanks operate only over a specific temperature range defined by their capacities. Since no actual tank temperatures are calculated, it is assumed that the temperature of the incoming fluid is greater than the tank temperature and that the temperature of the outgoing fluid is greater than that required to meet the prescribed load. This is accomplished by restricting the tank supply and return temperatures to be above minimum levels.

Furthermore, it is assumed that the rate of heat exchange to the storage tanks is sufficient so that any amount of energy up to the tank capacity can be stored in an hour.

e. Finally, the default routine that distributes loads among equipment of the same type allocates the loads so that the part-load ratio of the operating unit is close to the optimal part-load ratio specified by the user.

This routine assumes that the units have identical part-load efficiency functions and that these can be approximated by quadratic equations. Note that this rule is assumed to be simple and practical enough to be accomplished by installed, noncomputerized, control equipment.

If these assumptions cannot be made, the user can override this allocation procedure by providing equipment assignment tables. If equipment is not available to meet the above optimization assumptions, as is usually the case in a retrofit analysis, the minimization routines can be bypassed by assigning appropriate values to the special parameter codes (see Sec. C.1.d).

2. Loads Not Met. The PLANT Program assumes that sufficient equipment capacity is available to meet all loads encountered. Thus, any loads not met because of plant deficiency are ignored. If the plant is undersized, the resulting energy consumption values reported will be low. Thus, if the maximum heating or cooling loads reported in the Systems Load Summary for the SYSTEMS Program exceed the plant heating or cooling capacity, the capacities should be increased and the PLANT Program rerun.

C. Input Data

1. User-Supplied Input. The PLANT Program can be run with a minimum specification of the following data:

(1) Plant equipment type and size of each,

(2) The total number installed, and

(3) The maximum number of installed units that are available.

If the user specifies no other input, the program will default values for each parameter in the program.

However, a user may also specify any parameter if he desires to override the default values. For example, a user may specify

(4) Equipment part-load performance parameters,

(5) Equipment performance characteristics,

(6) Special system and equipment parameters,

(7) Assignment statements that determine the sequence of allocation of equipment to meet energy demands, and

(8) Desired output for printing.

These parameters are all input using the BDL commands that are discussed in detail in the Users Manual. However, a discussion of the action initiated by each command, the consequences, and underlying assumptions is given below.

a. Equipment Type and Size. The parameters input by the PLANT-EQUIPMENT command are the type, size, number installed, and number available. These values specify the operating capacity (nominal rated load) of each generic equipment type available. The number available indicates how many of the installed units can be put on line simultaneously to meet loads. Solar equipment parameters are input using the COMPONENT command (See Sec. F).

Equipment types available, except solar, are listed in Table V.1.

b. Equipment Part-Load Performance. Part-load performance of each type of available nonsolar equipment is specified by the PART-LOAD-RATIO command. Data input here include the minimum fraction of the nominal rated load (capacity) at which the machine will come on line, the maximum fraction of loading or overloading allowed, and the optimal part-load ratio. This optimal part-load ratio is the point on the machine operating curve that represents the most efficient or best operating point. It is at this optimal point that the program will try to operate, through its optimization approximation technique. This point is generally the point of most efficient operation, but it may be set to any point that the user feels is best for operation. Details of this technique are discussed in the subroutine descriptions, and default values are listed in Table V.3.

TABLE V.3

EQUIPMENT PART-LOAD RATIO DEFAULT VALUES

		Part-Load Ratios			Electric Input/
Code	Equipment	Minimum	Maximum	Optimum	Nominal Capacity
GTURB	Gas turbine	0.02	1.05	0.6000	0.0000
DIESL	Diesel engine	0.02	1.05	0.6000	0.0000
STMB	Steam boiler	0.01	1.00	0.8700	0.0000
ABS1	One-stage absorption chiller	0.05	1.10	0.6500	0.0077
ABS2	Two-stage absorption chiller	0.05	1.10	0.6500	0.0077
ABS2E	Two-stage absorption chiller with economizer	0.05	1.10	0.6500	0.0077
COMPH	Hermetic compression chiller	0.10	1.05	0.6500	0.2275
COMPC	Open centrifugal chiller	0.10	1.05	0.6500	0.2275
COMPR	Reciprocating chiller	0.10	1.05	0.6500	0.2275
DBUN	Double-bundle chiller	0.10	1.05	0.6500	0.2275
CTOWR	Cooling tower	0.00	0.00	0.4365	0.0120
STURB	Steam turbine	0.02	1.10	0.9000	0.0000
HTANK	Hot water tank	0.00	0.00	0.0000	0.0000
CTANK	Cold water tank	0.00	0.00	0.0000	0.0000

Finally, the electric input required for operation at nominal capacity, divided by the nominal capacity, is specified. This ratio represents energy required to operate electrical auxiliaries, pumps, fans, etc.

Table V.4 lists the equipment performance coefficient default values. These values are chosen to be representative of commonly available equipment. However, the user should first verify that these values are appropriate, and, if they are not, he should determine proper values from manufacturer's part-load performance data.

TABLE V.4

EQUIPMENT PERFORMANCE COEFFICIENT DEFAULT VALUES

P-Code	Equipment Name	Coeff 0	Coeff 1	Coeff 2
One-Stage Absorption Chillers				
CAVL1A	Available capacity	1.00000	0.00000	0.000000
REN1A	Energy input/output coefficient	0.08375	0.63170	0.267050
TCON1A	Condensate temperature coefficient	1.00000	0.00000	0.000000
SR1DTA	Steam rate coefficient	1.00000	0.00000	0.000000
Two-Stage Absorption Chillers				
CAVL2A	Available capacity	1.00000	0.00000	0.000000
REN2A	Energy input/output coefficient	0.11467	0.67212	0.212120
SR2DTA	Steam rate coefficient	1.00000	0.00000	0.000000
TCON2A	Condensate temperature coefficient	1.00000	0.00000	0.000000
REN2AE	Energy input/output coefficient (with economizer)	0.12917	0.36902	0.511360
Chillers				
RPWR1C	Energy input/output coefficient (hermetic compression)	0.16017	0.31644	0.518940
RPWR2C	Energy input/output coefficient (open centrifugal)	0.04864	0.54542	0.388850
RPWR3C	Energy input/output coefficient (reciprocal compression)	-0.01200	0.44727	0.560610
Double Bundle				
RCAVDB	Available capacity ratio	1.00000	-0.03300	-0.005601
RPWRDB	Energy input/output coefficient	0.16017	0.31644	0.518940
ADJTBD	Condensate cooling water temperature adjustment	95.00000	2.50000	44.000000

Double Bundle - Continued

ADJEDB	Energy ratio adjustment factor	1.61000	-0.61000	0.000000

Diesel

RELD	Power out/fuel input coefficient	0.09755	0.63180	-0.416500
RJACD	Jacket heat/fuel input coefficient	0.39220	-0.43670	0.277960
RLUBD	Lube heat/fuel input coefficient	0.08830	-0.13710	0.080300
REXD	Exhaust heat/fuel input coefficient	0.31440	-0.13530	0.097260
TEXD	Exhaust temperature coefficient	720.00000	60.00000	0.000000

Steam Boiler

RFVELB	Energy input/output ratio	0.60000	0.88889	-0.493827

Steam Turbine

RFSTUR	Steam flow coefficient	1.0	0.0	0.0

Gas Turbine

FUEL1G	Fuel input/output coefficient 1-3	7.6830	-13.480	8.000
FUEL2G	Fuel input/output coefficient 4-6	1.8822	-0.00433	0.000014
FEXG	Exhaust flow coefficient	0.01823	0.00003	0.0
TEX1G	Exhaust temperature coefficient 1-3	1.0	0.38450	0.028150
TEX2G	Exhaust temperature coefficient 4-6	406.96	0.63170	0.000224
ELUBG	Lube oil coefficient	0.22300	-0.4000	0.228600
UACG	Stack U-factor x area coefficient	0.03805	0.9000	-0.0

Tower

RF1	Rating factor temperature coefficient 1-3	7.6680	-0.12796	0.000594
RF2	Rating factor temperature coefficient 4-6	7.47850	-0.14145	0.000749
RF3	Rating factor temperature coefficient 7-9	4.69600	-0.08080	0.000400
RF4	Rating factor temperature coefficient 10-12	4.20850	-0.07881	0.000432
RF5	Rating factor temperature coefficient 13-15	3.18760	-0.05461	0.000277
RF6	Rating factor temperature coefficient 16-18	2.63970	-0.04440	0.000224
RFR	Rating factor range coefficient	0.0	0.1	0.0

c. Equipment Performance Characteristics. Operation of each generic equipment type, except for exhaust waste heat exchangers (which characterize performance as a function of U-value and log-mean temperature difference), is expressed by a quadratic function or product of quadratic functions of single variables that represent equipment energy performance. More than one function may be required for a complete specification. These coefficients, representing the constant, linear, quadratic terms, are specified by the PERFORMANCE-COEFFICIENT command.

The performance equations are represented by equations of the general form

$$P = (CT) + (LT)x_1 + (QT)x_1^2,$$

or sometimes as a product of quadratics,

$$P = \{(CT)_1 + (LT)_1\ x_1 + (QT)_1\ x_1^2\} \cdot \{(CT)_2 + (LT)_2\ x_2 + (QT)_2\ x_2^2\},$$

where x_1 and x_2 represent any operating variables, e.g., load ratio or ambient air temperature. Both the dependent variable, P = performance, and the independent variables, x_i, are ratios between 0 and 1.

The coefficients required for each generic equipment type, the corresponding operating variables (x_i's), and graphs of the default curves are discussed in the individual equipment subroutine descriptions. Default values for these coefficients, and the performance characteristics corresponding to each set of coefficients, are listed in Table V.4.

d. Special System and Equipment Parameters. Any parameter not classified as a part-load ratio or a performance coefficient is included in the special parameters. The types of parameters involved and the subroutines in which they are used are listed in Table V.5; the default values are listed in Table V.6.

e. Equipment Assignment. Normally the program assigns equipment to meet loads so that energy input to the plant is minimized. However, the user may override the default order of equipment assignment, among equipment of the same type, with the EQUIPMENT-ASSIGNMENT command. Note that the assignment of equipment between different types that are available, e.g., absorption vs compression chillers, is not affected by the EQUIPMENT-ASSIGNMENT command, but is predetermined by the program algorithms.

TABLE V.5

SPECIAL PARAMETER CODES

Code	Name	Routine
CPTYPE	TEPS type (1 = utility only, 2 = mixed)	OPCOOL
DTCOOL	Chilled water temperature rise	ABSREF
HFUELB	Heat content of fuel	BOILER
HSTEAM	Steam enthalpy	ABSREF HEATREC STMUSE
PELCL	Electric input to circular pump/cooling load	CCBTEPS
PELHT	Electric input to circular pump/heating load	CCBTEPS
PELTWR	Electric input to tower/tower cooling load	TOWER
PEXSTUR	Nominal exhaust steam pressure (psig)	STMTUR HEATREC OPCOOL
PSTEAM	Steam pressure (psig)	SATUR
PSTMTUR	Entering steam pressure (psig)	STMTUR
RAVRHDB	Available to recoverable heat ratio	DBUNDLE
RFLASH	Boiler flash water/steam feed	HEATREC
RMXKWD	Maximum exhaust flow/kW output	DIESEL
RMXKWG	Maximum exhaust flow/kW output	GASTUR
RPMNOM	Nominal speed (rpm)	STMTUR HEATREC OPCOOL
RWCA	Tower water/absorption chiller capacity	TOWER
RWCC	Tower water/compression chiller capacity	TOWER
RWCDB	Tower water/double-bundle chiller capacity	TOWER
RWSTUR	Condensate/entering steam	STMTUR EFFIC HEATREC STMUSE
SRATB	Air, fuel stoichiometric ratio	BOILER
SR1A	Full-load steam rate, one-stage absorption chiller	ABSREF
SR2A	Full-load steam rate, two-stage absorption chiller	ABSREF
TCOOL	Chilled water temperature	ABSREF DBUNDLE
TCW	Leaving condenser water temperature	DBUNDLE
TLEAVE	Boiler stack leaving temperature	BOILER

TMINC	Minimum tank temperature for cooling	SOCOOL
TOTUEF	Total efficiency of utility electrical generation	EFFIC
TOWOPR	Tower operation type (1 = variable water rate, 2 = fixed water rate)	TOWER
TSATUR	Steam saturation temperature	ABSREF GASTUR HEATREC
TSTMTUR	Entering steam temperature	STMTUR
TTOWR	Entering tower water temperature	ABSREF
TWMAKE	Make-up water temperature	HEATREC

Equipment is assigned over discrete load ranges where the user specifies that up to a given load a unit of a given size is assigned, etc.

f. <u>Output Specification.</u> Any combination of the following reports may be specified.

Central plant energy utilization monthly and annual summary

Equipment use statistics

Equipment sizes and availability

Equipment load ratios

Equipment performance coefficients

Equipment assignments

Special parameters

2. <u>Energy Load File Input from SYSTEMS Program.</u> The data input file to the PLANT Program from the SYSTEMS Program, which is named SYSHRO, consists of eight quantities for each hour:

(1) Preheat coil, QH (MBtu),

(2) Heating coil load (MBtu),

(3) Total reheat coil load (MBtu),

(4) Preheat coil air inlet temperature (°F),

(5) Heating coil air inlet temperature (°F),

(6) Reheat coil air inlet temperature (average of all reheat coils)(°F),

(7) Cooling load, QC (MBtu), and

(8) Electrical load, kW (kWh).

The file consists of one record for each day, in groups of eight, each group containing the load data for one hour.

The data are unformatted, and the program takes the absolute value of Items (1), (7), and (8).

TABLE V.6

SPECIAL PARAMETER DEFAULT VALUES

Code	Name	Value (English Units)
HSTEAM	Steam enthalpy	1199.578
TSATUR	Steam saturation temperature	369.635
RFLASH	Boiler flash water/steam feed (HEATREC)	0.071
PELCL	Electrical input to circulating pump/cooling load	0.018
PELHT	Electrical input to circulating pump/heating load	0.006
PELTWR	Electrical input to tower/tower cooling load (TOWER)	0.013
TOWOPR	Tower operating type (TOWER)	2.000
TWMAKE	Make-up water temperature (HEATREC)	55.000
TCOOL	Chilled water temperature	44.000
DTCOOL	Chilled water temperature rise	15.000
TTOWR	Entering tower water temperature	60.000
TCW	Leaving condenser water temperature	110.000
TMINC	Minimum tank temperature for cooling (SOCOOL)	180.000
CPTYPE	TEPS type (1. = utility only, 2. = mixed total energy plant)	1.000
TLEAVE	Boiler stack leaving temperature (BOILER)	550.000
SR2A	Full-load steam rate (two-stage absorption chiller)	12.200
SR1A	Full-load steam rate (one-stage absorption chiller)	18.700
RAVRHDB	Available recoverable heat ratio (DBUNDLE)	0.950
RMXKWD	Maximum exhaust flow/kW output (DIESEL)	5.000
RMXKWG	Maximum exhaust flow/kW output (GASTURB)	40.000
RWCA	Tower water/absorption chiller capacity (TOWER)	3.000
RWCC	Tower water/compression chiller capacity (TOWER)	3.000
RWCDB	Tower water/double-bundle chiller capacity (TOWER)	3.000
SRATB	Air, fuel stoichiometric ratio (BOILER)	17.000
HFUELB	Heat content of fuel (BOILER)	20000.000
RHFLASH	Recovered heat/flash steam energy (HEATREC)	0.500
PSTEAM	Steam pressure (psig)	150.000
PSTMTUR	Entering steam pressure (psig) (steam turbine)	150.000
TSTMTUR	Entering steam temperature (steam turbine)	494.635
PEXSTUR	Nominal exhaust steam pressure (psig) (steam turbine)	-12.700
RPMNOM	Nominal speed (rpm) (steam turbine)	10000.000
RWSTUR	Condensate/entering steam (steam turbine)	0.970
TOTUEF	Total efficiency of utility electrical generation (EFFIC)	0.300

D. Program Structure and Operation

The structure and operation of the PLANT Program is best illustrated by both functional charts showing energy flows and equipment relationships and flow charts showing the program logic and the relationship between subroutines.

1. Program Functional Chart. Figure V.1 is an overall functional chart for the PLANT Program that shows the flow of energy and how various pieces of equipment relate to this energy flow. First, loads are input from the load file for a day, and the hourly simulation loop is begun. An attempt is first made to satisfy the heating load from solar energy, if solar equipment is available. Next an attempt is made to meet or partly meet the heating and cooling loads from either stored hot or chilled water or by direct heat exchange with waste heat streams, and the remaining loads are passed to the heating and cooling equipment.

Solar cooling is then attempted, and the remaining load is passed to absorption or compression chillers, or both. Finally, the hourly heating, cooling, and electrical loads remaining are converted to energy requirements at the primary energy equipment. Waste heat from the primary energy equipment is passed to the waste heat recovery heat exchangers where it is applied to the heating and cooling loads for the hour.

After the primary energy equipment is simulated, the source energy consumption at the building boundary is determined and stored for reporting.

The above procedure is repeated for each hour of the day, and a new day's set of loads is read from the energy load file. The whole loop is repeated for each month of the year, and the results are sent to the report file.

2. Flow Charts. The main program, CCBTEPS, directs the overall sequence of operation, as illustrated in the flow chart in Fig. V.1. The basic calculational sequence that is repeated is the hourly plant simulation loop illustrated in Fig. V.2. Each of the major calculational blocks shown allocates loads and simulates plant equipment. They are further described in Figs. V.3 through V.9.

A detailed description of each subroutine, including flow charts, is given in Sec. E.

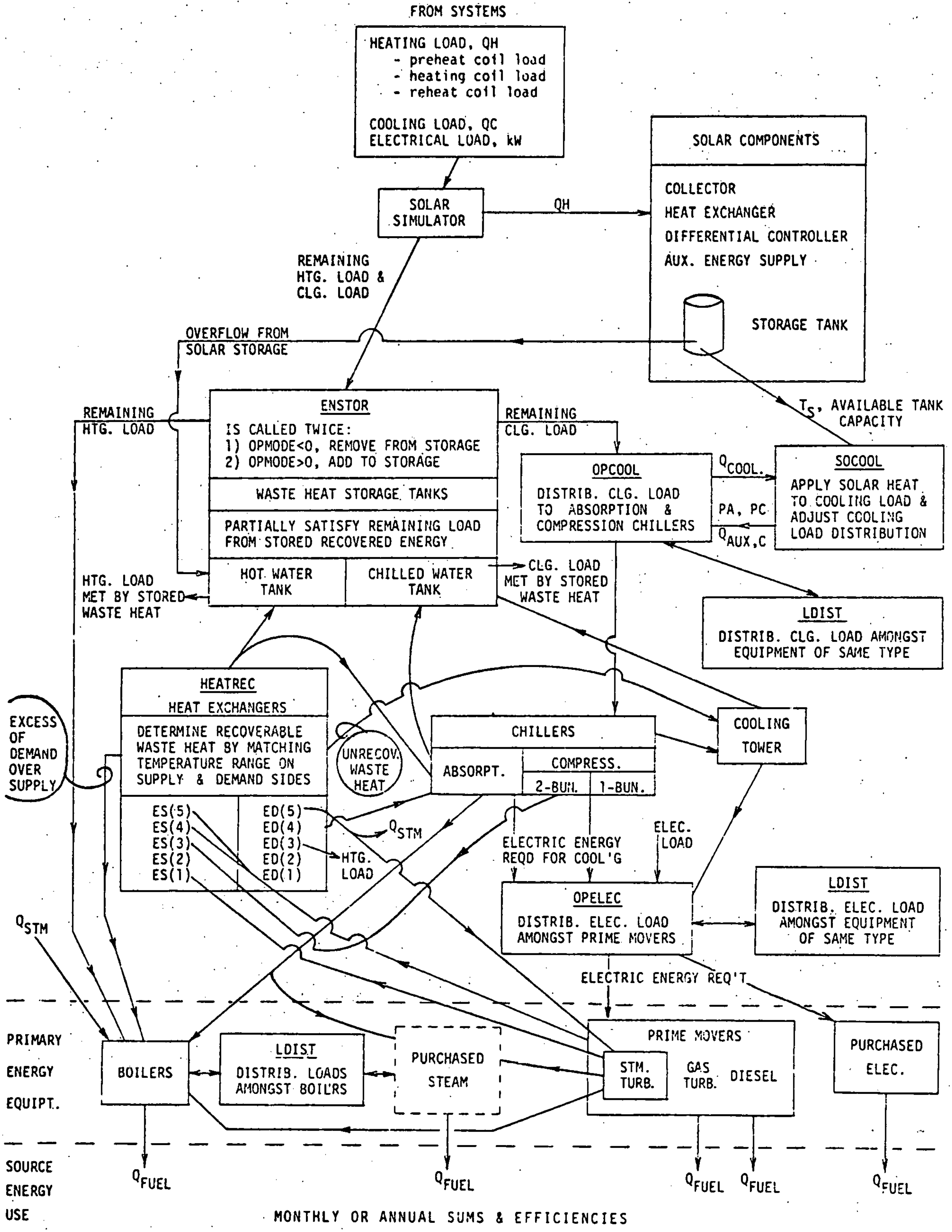


Fig. V.1 Cal-ERDA PLANT functional chart.

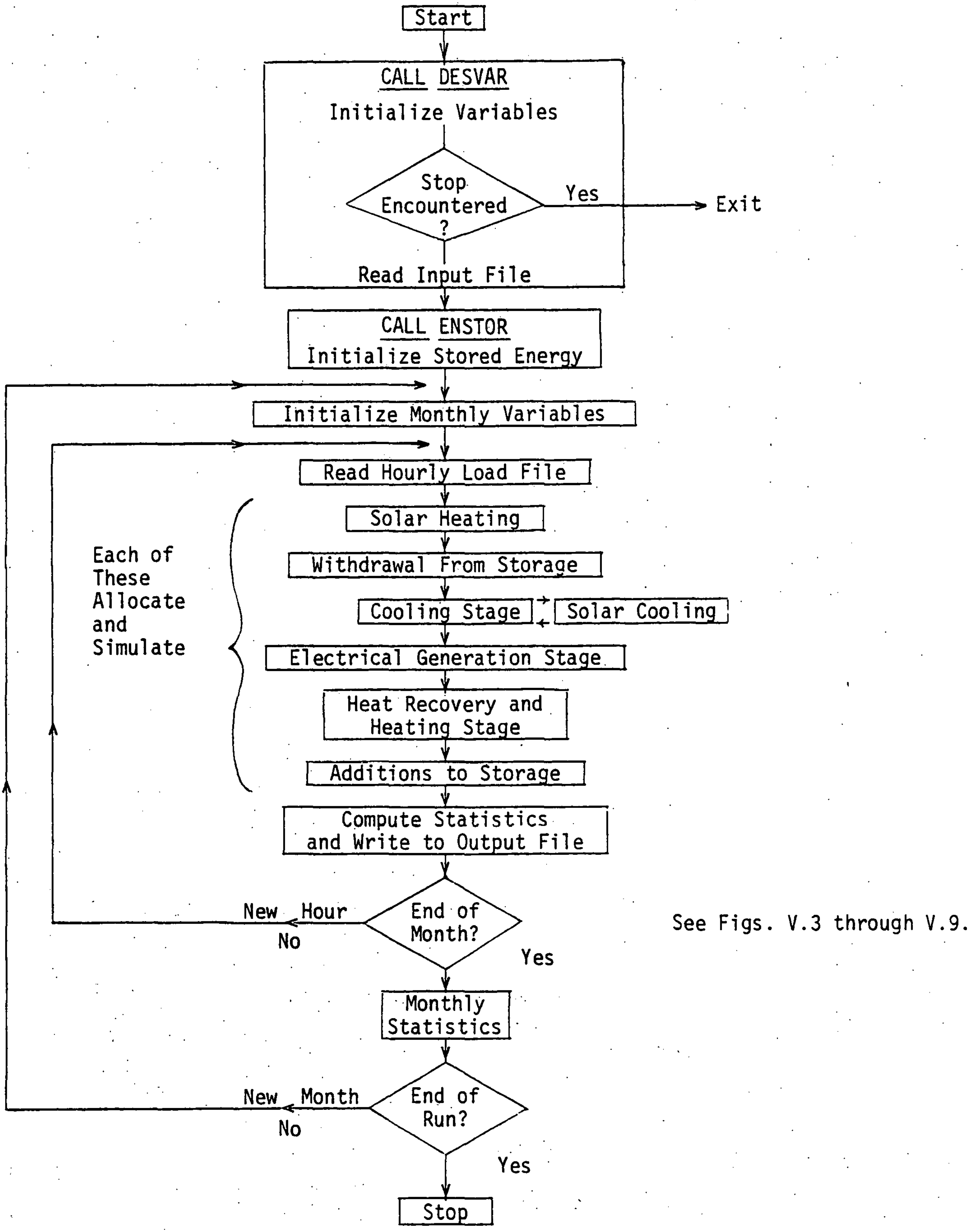
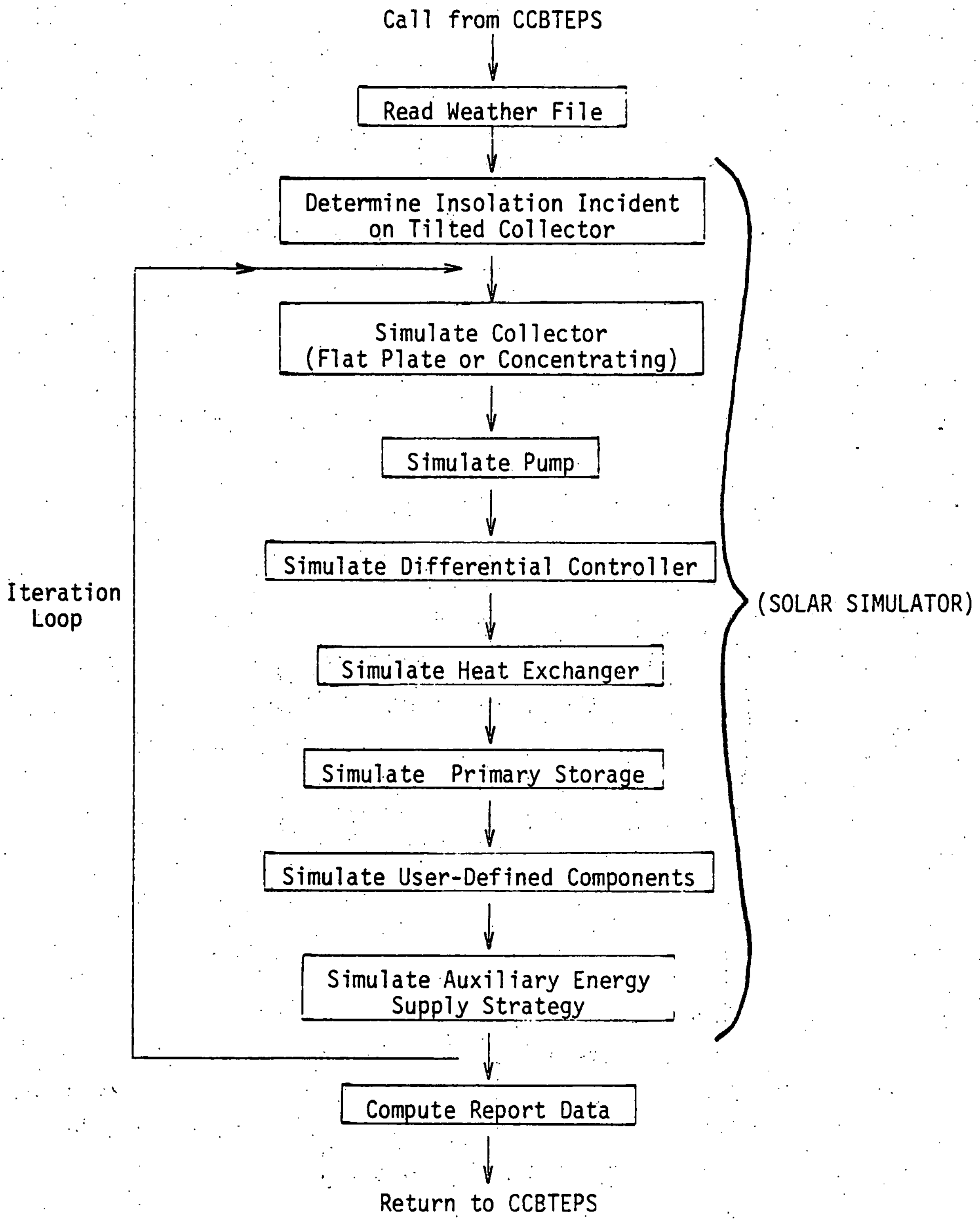
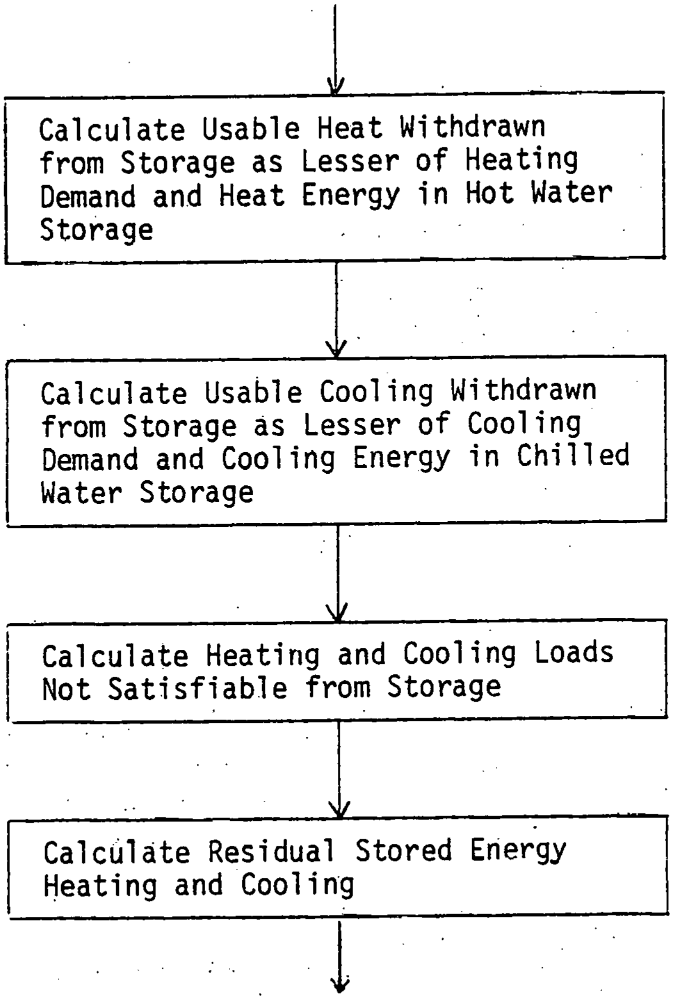


Fig. V.2. Hourly plant simulation loop illustration.



Call from CCBTEPS
Read Weather File
Determine Insolation Incident on Tilted Collector
Simulate Collector (Flat Plate or Concentrating)
Simulate Pump
Simulate Differential Controller
Simulate Heat Exchanger
Simulate Primary Storage
Simulate User-Defined Components
Simulate Auxiliary Energy Supply Strategy
Iteration Loop
(SOLAR SIMULATOR)
Compute Report Data
Return to CCBTEPS

Fig. V.3. Solar heating simulation.


Calculate Usable Heat Withdrawn from Storage as Lesser of Heating Demand and Heat Energy in Hot Water Storage
Calculate Usable Cooling Withdrawn from Storage as Lesser of Cooling Demand and Cooling Energy in Chilled Water Storage
Calculate Heating and Cooling Loads Not Satisfiable from Storage
Calculate Residual Stored Energy Heating and Cooling

(ENSTOR, called with first argument < 0)

Fig. V.4. Withdrawal from storage.

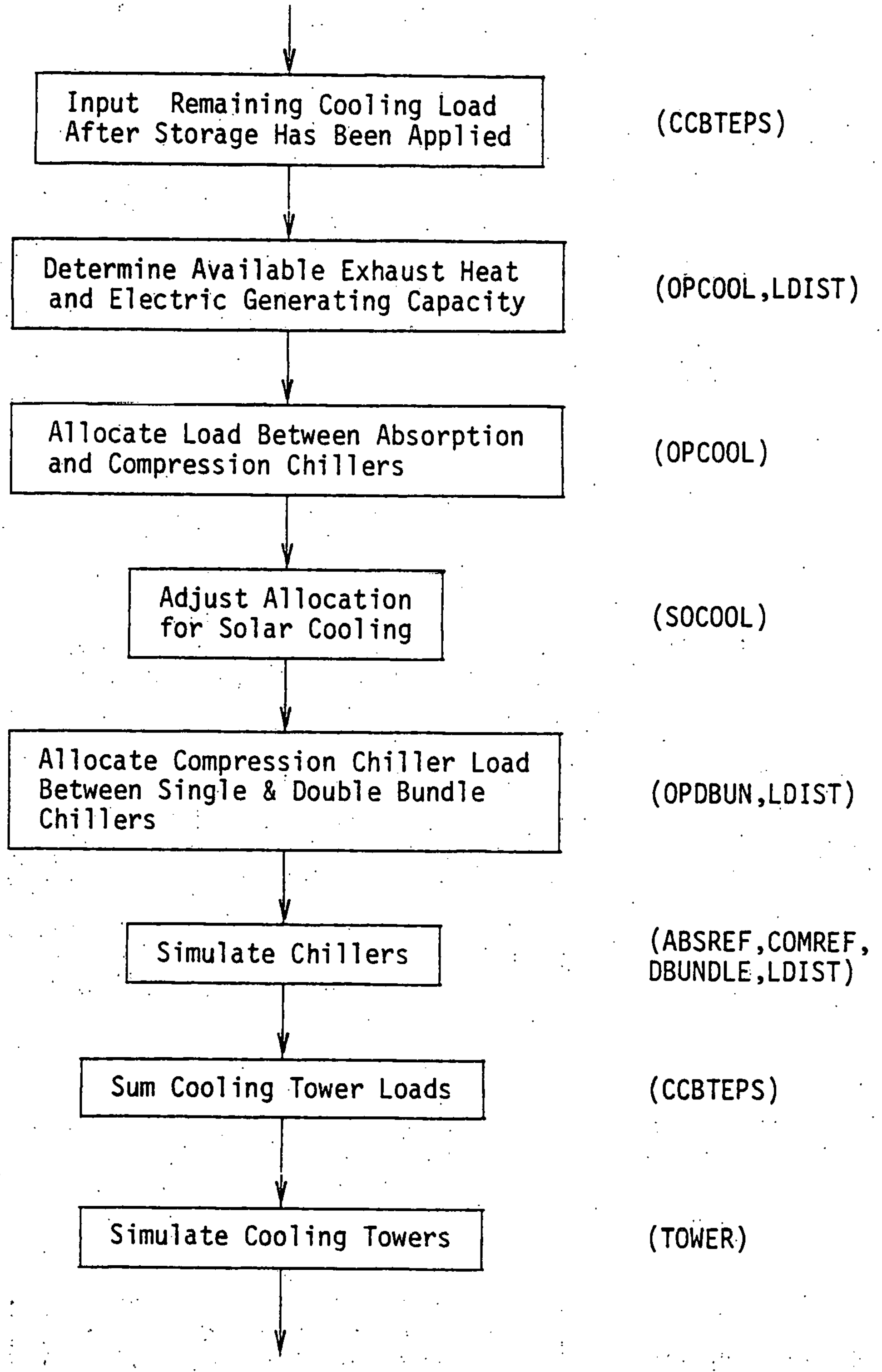

Input Remaining Cooling Load After Storage Has Been Applied
(CCBTEPS)
Determine Available Exhaust Heat and Electric Generating Capacity
(OPCOOL,LDIST)
Allocate Load Between Absorption and Compression Chillers
(OPCOOL)
Adjust Allocation for Solar Cooling
(SOCOOL)
Allocate Compression Chiller Load Between Single & Double Bundle Chillers
(OPDBUN,LDIST)
Simulate Chillers
(ABSREF,COMREF, DBUNDLE,LDIST)
Sum Cooling Tower Loads
(CCBTEPS)
Simulate Cooling Towers
(TOWER)

Fig. V.5. Cooling stage.

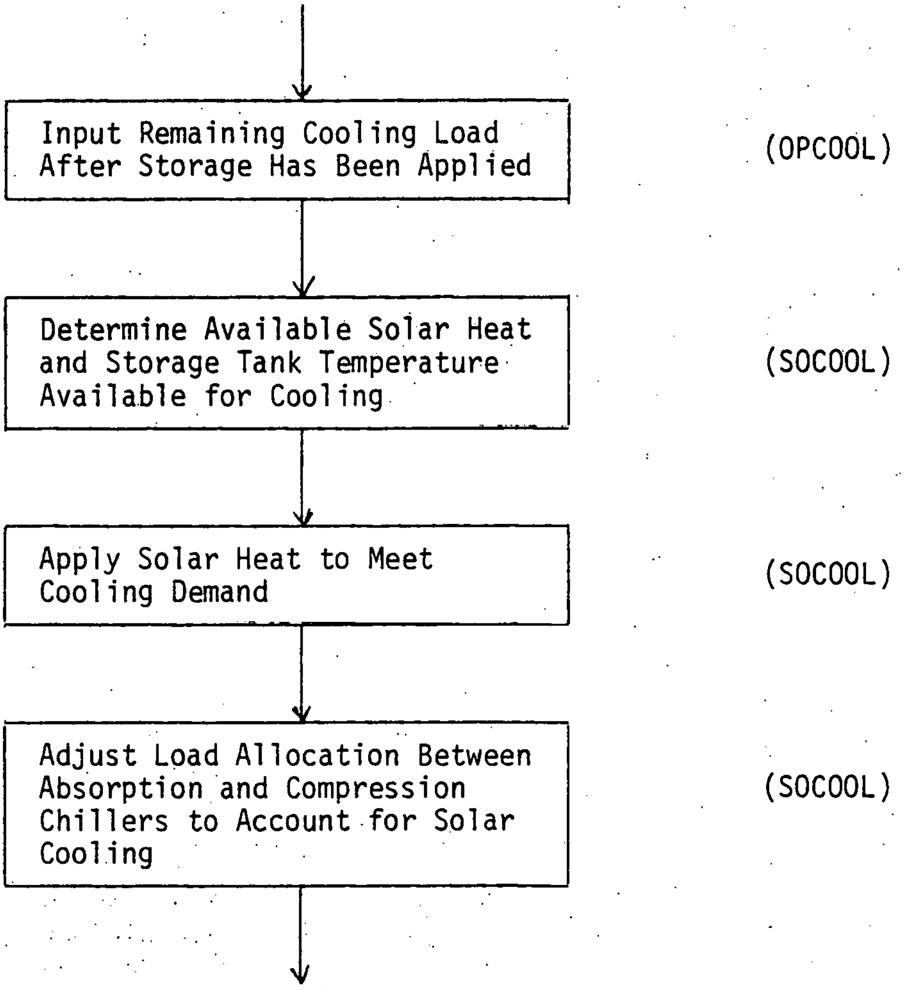

Input Remaining Cooling Load After Storage Has Been Applied
(OPCOOL)
Determine Available Solar Heat and Storage Tank Temperature Available for Cooling
(SOCOOL)
Apply Solar Heat to Meet Cooling Demand
(SOCOOL)
Adjust Load Allocation Between Absorption and Compression Chillers to Account for Solar Cooling
(SOCOOL)

Fig. V.6. Solar cooling.

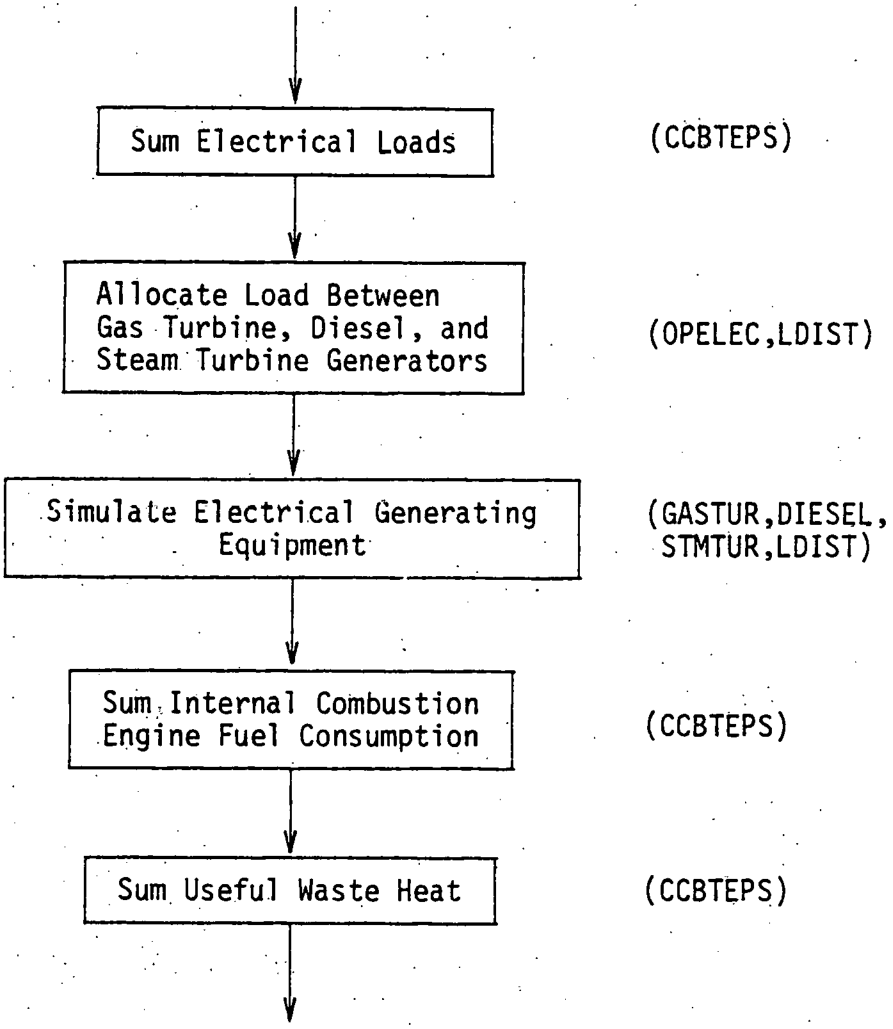

Sum Electrical Loads
(CCBTEPS)
Allocate Load Between Gas Turbine, Diesel, and Steam Turbine Generators
(OPELEC,LDIST)
Simulate Electrical Generating Equipment
(GASTUR,DIESEL, STMTUR,LDIST)
Sum Internal Combustion Engine Fuel Consumption
(CCBTEPS)
Sum Useful Waste Heat
(CCBTEPS)

Fig. V.7. Electrical power generation stage.

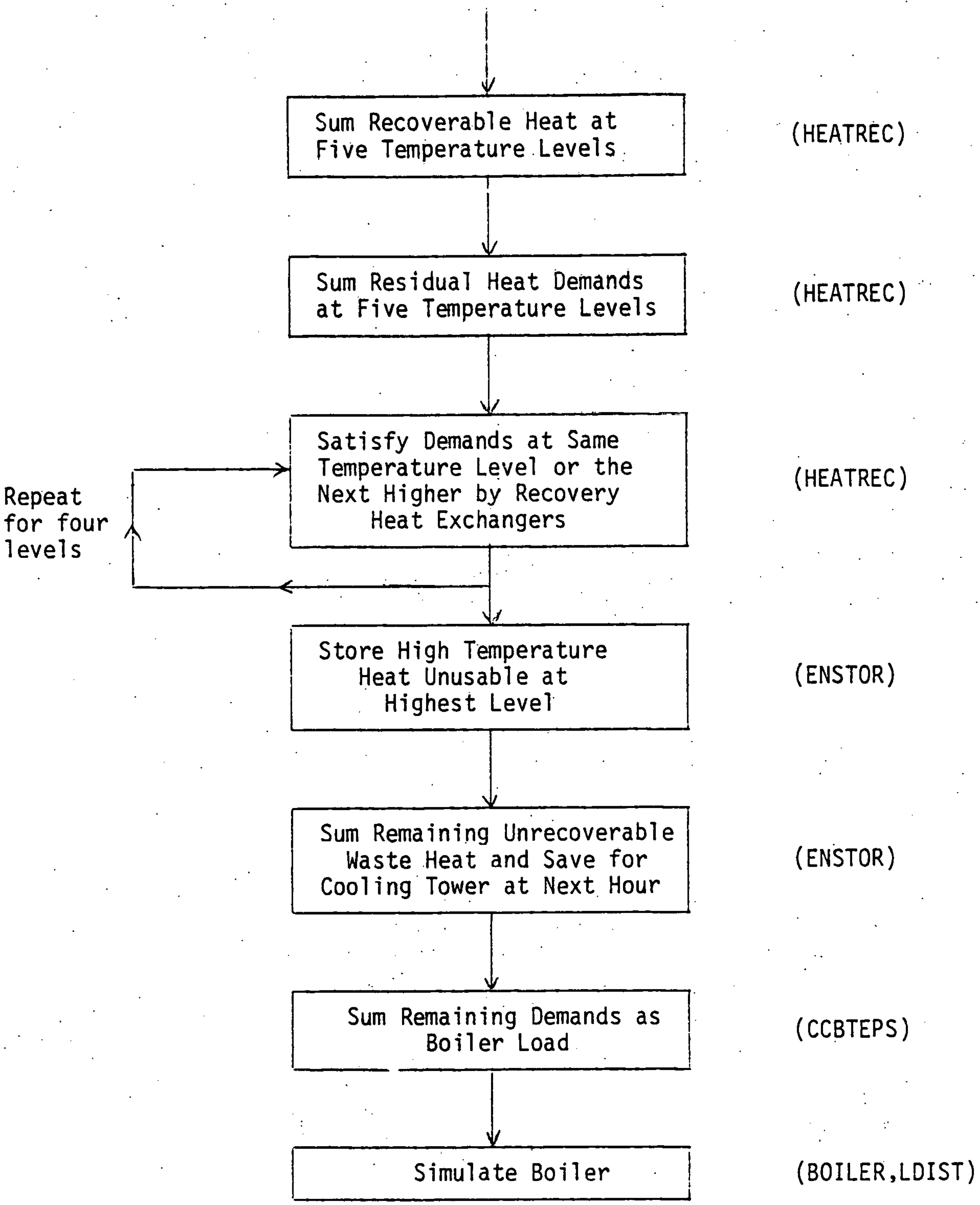

Sum Recoverable Heat at Five Temperature Levels
(HEATREC)
Sum Residual Heat Demands at Five Temperature Levels
(HEATREC)
Satisfy Demands at Same Temperature Level or the Next Higher by Recovery Heat Exchangers
(HEATREC)
Repeat for four levels
Store High Temperature Heat Unusable at Highest Level
(ENSTOR)
Sum Remaining Unrecoverable Waste Heat and Save for Cooling Tower at Next Hour
(ENSTOR)
Sum Remaining Demands as Boiler Load
(CCBTEPS)
Simulate Boiler
(BOILER,LDIST)

Fig. V.8. Heat recovery and heating stage.

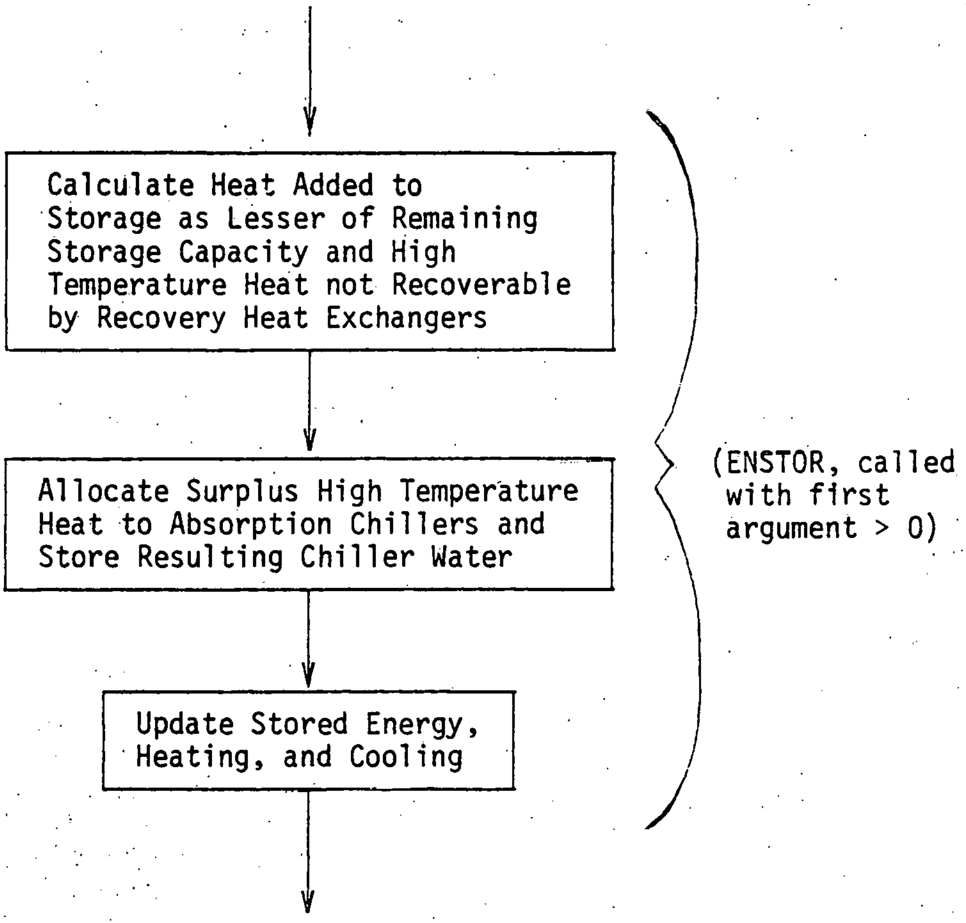

Calculate Heat Added to Storage as Lesser of Remaining Storage Capacity and High Temperature Heat not Recoverable by Recovery Heat Exchangers
Allocate Surplus High Temperature Heat to Absorption Chillers and Store Resulting Chiller Water
Update Stored Energy, Heating, and Cooling
(ENSTOR, called with first argument > 0)

Fig. V.9. Addition to storage.

E. Subroutine Descriptions

1. List of Subroutines. A description of all routines used to simulate central plant operation and components is given in this section. Structured flow charts are presented where appropriate.

Subroutines are of the types listed in Table V.7. A list of all program variations in common blocks, with brief descriptions, is given in Table V.8.

TABLE V.7

CLASSIFIED LIST OF SUBROUTINES

Class	Subroutine	Brief Description
Main Program	CCBTEPS	Inputs loads, controls monthly and hourly simulation
Equipment Simulation	ABSREF	Simulates three types of absorption chillers
	BOILER	Simulates steam boilers
	COMREF	Simulates three types of compression chillers
	DBUNDLE	Simulates double-bundle chillers
	DIESEL	Simulates diesel engine/generator sets
	ENSTOR	Calculates energy storage in hot and chilled water tanks
	GASTUR	Simulates gas turbine/generator sets
	SOCOOL	Simulates solar cooling
	STMTUR	Simulates steam turbines
	TOWER	Simulates cooling towers
Thermodynamic Functions	ENTHAL	Specific enthalpy of steam
	ENTROP	Specific entropy of steam
	RFACT	Rating factor of cooling towers
	SATUR	Saturation temperature of steam
	WETBULB	Wet-bulb temperature Fahrenheit
Load Allocation	HEATREC	Calculates heat recoveries and required steam boiler output
	LDIST	Distributes load to similar equipment units with identical efficiency functions
	OPCOOL	Distributes cooling load to chiller classes
	OPDBUN	Calculates double-bundle cooling load ratio to total cooling load
	OPELEC	Distributes electrical load to generator
	STMUSE	Calculates total steam consumption
Utility	DESVAR	Initializes and reads input data
	DFLTASG	Assigns default values and post processes input data
	DFLTP	Contains performance coefficient defaults
	DFLTS	Contains special parameter defaults
	EFFIC	Calculates total efficiencies and fuel consumption
	KARDRD	Reads a data card and echo prints
	LSEARC	Searches key lists to identify keyword codes
	REPORT	Calls specific report routines
	RTPRNT	Prints report titles
	R1PRNT	Prints monthly values

Utility - Continued	R3PRNT	Prints all internal parameters
	STATIS	Calculates monthly sums
	STATPR	Prints equipment use statistics
	STATSM	Stores monthly results
	WDREAD	Reads weather file
Solar Simulator	CBS	Executive
	CCM	Computes cloud-cover modifier
	ERR	Error processor
	INCRTM	Increments time step
	JEXEC	Junior executive
	RDCOMP	Reads data from input processor
	SOLCOM	Computes solar radiation components (direct, diffuse, reflected)
	SOLDAY	Computes daily solar values
	SOLPOS	Computes solar position
	TRACE	Debug routine
	TYPE1	Simulates solar collector
	TYPE2	Simulates differential controller
	TYPE3	Simulates variable flow rate pump
	TYPE4	Simulates hot water storage tank
	TYPE5	Simulates heat exchanger
	TYPE10	Simulates rock bed
	TYPE16	Simulates insolation
	TYPE21	Simulates preconnected liquid subsystem
	TYPE22	Simulates preconnected air subsystem
	TYPE25	Report routine
	TYPE29	Simulates auxiliary controller, liquid
	TYPE31	Simulates auxiliary controller, air
	CBSR	Main report routine
	RPT1	Insolation report routine
	RPT2	System performance report routine
	RPT3	Iteration report routine

TABLE V.8

LIST OF VARIABLES IN COMMON BLOCKS

Variable	Common Block	Description
AMAXLD(I)	/STATD/	Maximum part load on equipment index I (Btu)
CNOM(J,I)	/EDATA/	Nominal size of equipment type I, size index J (Btu)
CNT	/TOWERD/	Cooling capacity per unit cell at 90-80-70 point (Btu/hr)(=CNOM(1,ITOWR))
CNTH	/TOWERD/	Cooling capacity per unit cell at half speed (Btu/hr) (=CNOM(1,ITOWR)*ROPT(ITOWR))
CNTU	/TOWERD/	CNT in tower units
CNTUH	/TOWERD/	CNTH in tower units
DEMAND(I)	/EPARS/	Demand on equipment I for current hour (Btu)
DISTB(IL,I)	/DISTB/	Load range (Btu)
EBOILER	/HOURTOT/	Boiler net heat output (Btu/hr)
ECOOL	/HOURTOT/	Total cooling energy (Btu/hr)

ECOOLS	/MONTOT/	Monthly total cooling energy (Btu/month)
EELECCD	/HOURTOT/	Electric energy input to double-bundle chillers (Btu/hr)
EELECCO	/EFFICD/	Total electric energy input to cooling stage (Btu/hr)
EELECT	/HOURTOT/	Total electric energy including that used by plant (Btu/hr)
EELECTS	/MONTOT/	Monthly total electric energy including that used by plant (Btu/month)
EFIEC	/HOURTOT	Total fuel input for electric energy consumed by cooling stage (Btu/hr)
EFIECS	/MONTOT/	Monthly total fuel input for electric energy consumed by cooling stage (Btu/month)
EFIHC	/HOURTOT/	Total fuel input for heat energy consumed by cooling stage (Btu/hr)
EFIHCS	/MONTOT/	Monthly total fuel input for heat energy consumed by cooling stage (Btu/month)
EFUEL	/HOURTOT/	Total fuel energy input to plant (Btu/hr)
EFUELB	/HOURTOT/	Total boiler fuel energy (Btu/hr)
EFUELBS	/MONTOT/	Monthly total boiler fuel energy (Btu/month)
EFUELD	/HOURTOT/	Total fuel input to diesel engines (Btu/hr)
EFUELDS	/MONTOT/	Monthly total fuel input to diesel engines (Btu/month)
EFUELE	/HOURTOT/	Total fuel input for electric energy generation by the plant (Btu/hr)
EFUELES	/MONTOT/	Monthly total fuel input for electric energy generation (Btu/month)
EFUELG	/HOURTOT/	Total fuel input to gas turbines (Btu/hr)
EFUELGS	/MONTOT/	Monthly total fuel input to gas turbines (Btu/month)
EFUELHS	/MONTOT/	Monthly total fuel input for heat energy generation (Btu/month)
EFUELIE	/HOURTOT/	Total fuel input for electric energy generation by the plant and the utility (Btu/hr)
EFUELIH	/HOURTOT/	Total fuel input for heat energy generation (Btu/hr)
EFUELS	/MONTOT/	Monthly total fuel energy input (Btu/month)
EHEAT	/EFFICD/	Heat energy (building) load (Btu/hr)
EHEATT	/HOURTOT/	Total heat energy (Btu/hr)
EHEATTS	/MONTOT/	Monthly total heat energy (Btu/month)
ENPEAK(1-12,IU)	/STATD/	Monthly energy peak for utility index IU (Btu/hr)
ENPEAK(13,IU)	/STATD/	Yearly energy peak for utility index IU (Btu/hr)
ENTOT	/HOURTOT/	Total energy consumed by the plant and the utility company (Btu/hr)

ENTOTS	/MONTOT/	Monthly total energy consumed by the plant and the utility (Btu/month)
ENUSE(1-12,IU)	/STATD/	Monthly energy used for utility index IU (Btu)
ENUSE(13,IU)	/STATD/	Yearly energy used for utility index IU (Btu)
EPLANT	/EFFICD/	Electric energy generated in the plant (Btu/hr)
ERCVCD	/EFFICD/	Heat energy recovered from double-bundle chiller (Btu/hr)
ERECOVR	/HOURTOT/	Total recovered heat energy (Btu/hr)
ERECOVS	/MONTOT/	Monthly total recovered heat energy (Btu/month)
ESOLAR	/SOLARD/	Available solar energy (Btu/hr)
ESOLC	/SOLARD/	Solar heat energy used for cooling (Btu)
ESOLH	/SOLARD/	Solar heat energy used for heating (Btu)
ESTMAB	/EFFICD/	Steam energy input to absorption chiller (Btu/hr)
ESTMS	/STM/	Total steam energy consumed by steam users except absorption chillers (Btu/hr)
ESTORBL	/HOURTOT/	Total heat energy available for storing during hour (Btu/hr)
ESTRED	/HOURTOT/	Total heat energy stored during hour (Btu/hr)
EUT	/HOURTOT/	Total utility electricity (Btu/hr)
EUTS	/MONTOT/	Monthly total utility electricity (Btu/month)
EWASTCD	/EFFICD/	Wasted recoverable heat energy in double-bundle chiller (Btu/hr)
EWASTE	/EFFICD/	Waste heat energy from diesel engines and gas turbines (Btu/hr)
EWASTED	/HOURTOT/	Total wasted recoverable heat energy (Btu/hr)
EWASTES	/MONTOT/	Monthly total wasted recoverable heat energy (Btu/month)
EWTRM	/STM/	Energy of water mixture from steam users (Btu/hr)
FE	/EFFICD/	Inverse of plant electric energy generation efficiency
FH	/EFFICD/	Inverse of plant heat energy generation efficiency
FSTMS	/STM/	Total steam flow into steam users (lbs/hr)
FSTMTUR	/STMTUR/	Flow of steam entering steam turbine (lbs/hr)
FUE	/EFFICD/	Inverse of total electric energy generation efficiency
FWTRM	/STM/	Flow of return water mixture (lbs/hr)
HEXSTM	/SIMTUR/	Enthalpy of exhaust steam from steam turbine (Btu/lb)
HR	/WEATHR/	Humidity ratio
HSTMTUR	/STMTUR/	Enthalpy of superheated high pressure steam (Btu/lb)
IABSOR	/EPARS/	Absorption chiller type index

ICOMPR	/EPARS/	Compression chiller type index
ICONT	/CARD/	Column 80 of data card
IDAY	/DATE/	Day number
IDISTB(J,IL,I)	/DISTB/	Number of units in use of size index J, load range index IL
IENAME(1,I)	/EDATA/	Equipment code (6H...... format). If IENAME (1,I) = 0, equipment indexed by I is undefined
IENANE(2-4,I)	/EDATA/	Equipment name (30H...... format)
IHR	/DATE/	Hour number
IMON	/DATE/	Month number
IN(1-7)	/CARD/	Columns 10-79 of data card
IOPR(J,I)	/LDISTD/	Number of units operating at current time step for equipment index I, size index J
IOPRHR(J,I)	/STATD/	Number of operation hours of equipment indexed by I, size indexed by J
IPRT	/CDPR/	Logical unit number for print output
IREPOP(1,IR)	/REPOPT/	Report option of index IR (2H..... format)
IREPOP(2,IR)	/REPOPT/	Report option page control of index IR (3H...... format)
IREPOPD(1,IR)	/REPOPT/	Defaults of IREPOP(1,IR) for P all card
IREPOPD(2,IR)	/REPOPT/	Defaults of IREPOP(2,IR) for P all card
ISEQ	/CARD/	Column 3 of data card
ITIT(1-8,IT)	/TITLED/	Title data of index IT (80H..... format)
ITOWR	/EPARS/	Tower type index
KARD	/CDPR/	Logical unit number for card input
KAV(J,I)	/EDATA/	Number available type I, size index J
KEYLST(1,IK)	/KEYLST/	Data card type code (2H.. format). If KEYLST (1,IK) = 0, then type indexed by IK is undefined
KEYLST(2-4,IK)	/KEYLST/	Data card type name (30H..... format)
KEYMAX	/KEYLST/	Range of IK
KEY1	/CARD/	Columns 1-2 of data card
KINS(J,I)	/EDATA/	Number installed type I, size index J
KT	/TOWERD/	Number of cells in tower (=KAV(ITOWR))
MAXTIM(1,I)	/STATD/	Month of maximum part load
MAXTIM(2,I)	/STATD/	Day of maximum part load
MAXTIM(3,I)	/STATD/	Hour of maximum part load
MTIT	/TITLED/	Maximum number of titles available
NAM	/CARD/	Columns 4-9 of data card

NDISTB(I)	/DISTB/	Number of load ranges for load distribution by table (i.e., range of IL)
NEDATA	/EDATA/	Range of I
NEQSIZE(I)	/EDATA/	Number of different sizes of equipment type I (i.e., range of J)
NOPR(I)	/LDISTD/	Number of different sizes of equipment operating at current time step for equipment index I
NPDATA	/PDATA/	Range of IP
NREPOP	/REPOPT/	Range of IR
NSDATA	/SDATA/	Range of IS
NTIT	/TITLED/	Maximum number of titles entered
OPCAP(I)	/EPARS/	Operating capacity of equipment I (Btu)
OPCAPY(I)	/STATD/	Operating capacity of equipment I totaled over the year (Btu)
PD	/EFFICD/	Ratio of diesel-engine load to total electrical load
PDATA(1-3,IP)	/PDATA/	Equipment performance coefficients 1,2,3 of index IP
PDATA(4,IP)	/PDATA/	Equipment performance coefficient code of index IP (6H..... format); if PDATA(4,IP) = 0, those coefficients are unused
PDATA(5-8,IP)	/PDATA/	Equipment performance coefficient name of index IP (40H... format)
PEL(I)	/EDATA/	Electric input to nominal capacity ratio (Btu/Btu)
PG	/EFFICD/	Ratio of gas turbine load to total electrical load
PLOAD(I)	/EPARS/	Part load of equipment I for current hour (Btu)
PLOADY(I)	/STATD/	Part load of equipment I totaled over the year (Btu)
PNTK	/TOWERD/	Fan motor power for one cell (kW)
PRNTA1(IMON,1)	/PRNTA1/	Total heat energy (Btu/month)
PRNTA1(IMON,2)	/PRNTA1/	Total electric energy (Btu/month)
PRNTA1(IMON,3)	/PRNTA1/	Cooling electric energy (Btu/month)
PRNTA1(IMON,4)	/PRNTA1/	Recovered energy (Btu/month)
PRNTA1(IMON,5)	/PRNTA1/	Wasted recoverable energy (Btu/month)
PRNTA1(IMON,6)	/PRNTA1/	Heat energy input for cooling (Btu/month)
PRNTA1(IMON,7)	/PRNTA1/	Electric energy input for cooling (Btu/month)
PRNTA1(IMON,8)	/PRNTA1/	Energy input for heating (Btu/month)
PRNTA1(IMON,9)	/PRNTA1/	Energy input for electricity (Btu/month)
PRNTA1(IMON,10)	/PRNTA1/	Total fuel input (Btu/month)
PRNTA1(IMON,11)	/PRNTA1/	Total energy input (Btu/month)
PRNTA1(IMON,12)	/PRNTA1/	Average plant efficiency (fraction)
PRNTA1(IMON,13)	/PRNTA1/	Unused

PS	/EFFICD/	Ratio of steam turbine load to total electrical load
RMAX(I)	/EDATA/	Maximum part-load ratio of equipment type I
RMIN(I)	/EDATA/	Minimum part-load ratio of equipment type I
ROPT(I)	/EDATA/	Optimum part-load ratio of equipment type I
SDATA(1,IS)	/SDATA/	Special parameter value of index IS
SDATA(2,IS)	/SDATA/	Special parameter code of index IS (7H..... format); if SDATA(2,IS) = 0., that special parameter space IS is unused
SDATA(3-6,IS)	/SDATA/	Special parameter name of index IS (40H..... format)
SSTMTUR	/STMTUR/	Entropy of (superheated) high-pressure steam
TAIR	/WEATHR/	Air temperature (°F)
TEXSTM	/STMTUR/	Temperature of exhaust steam from steam turbine (°F)
TOTCAP(I)	/EPARS/	Total nominal capacity of equipment I (Btu)
TOTUEF	/EFFICD/	Electric energy efficiency of utility
TSOLAR	/SOLARD/	Solar energy storage tank temperature (°F)
TWET	/WEATHR/	Wet-bulb temperature (°F)

a. Main Program - CCBTEPS. CCBTEPS is the main program that directs the logical flow, inputs data, initializes variables, and controls the hourly, daily, and monthly simulation.

b. Equipment Simulation Subroutines. All equipment simulation is done in the same way (by a sequence of equations). There is no iteration in the computation except for the cooling tower where the rating factor is approximated by an iterative procedure. Calculations are in dimensionless variables wherever possible.

Performance equations are expressed as products of quadratics of single variables; therefore, the coefficients in performance equations can be assigned by a simple one-dimensional curve-fitting procedure. Note that approximating performance equations as products of single-variable quadratics is realistic, since most, if not all, manufacturers present component performance data as one-dimensional function curves where all variables except one are fixed.

The quadratics are sequenced in descending order of importance as follows:

$$y = f_1(x_1,c^1) * f_2\ (x_2,c^2) * \ldots * f_n(c_n,c^n),$$

in which the superscript notation is explained in Sec. C.1.c. In the present version of Cal-ERDA, the product of no more than two quadratics is used in any performance equation. The x's are the independent variables, with the first being the main variable; y is the dependent variable; and

$$f_i(x_i, c^i) = c_0{}^i + c_1{}^i * x_i + c_2{}^i * (x_i)^2.$$

The performance coefficients, c^i, are three-vectors and are listed in Table V.4. Users can either use the existing default values or enter their own data.

Example: In subroutine GASTUR, the ratio of input energy to output energy is given as follows:

Fuel energy input/electric energy output

= f_1 (part-load ratio, FUEL1G) * f_2 (ambient air temperature, FUEL2G),

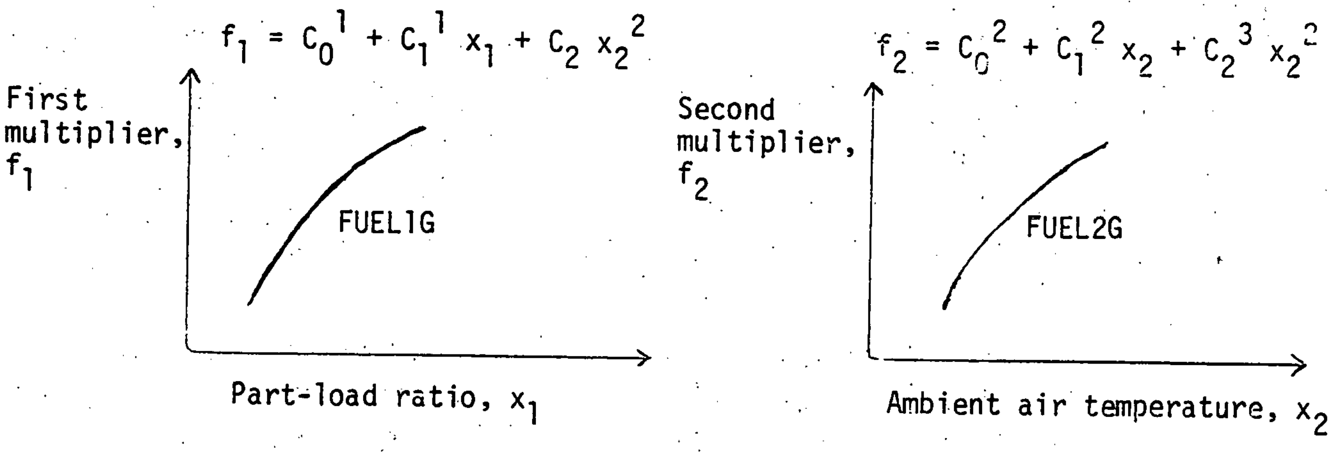
or

Fuel energy input = electric energy output

$$* [\text{FUEL1G}(1) + \text{FUEL1G}(2) * r + \text{FUELIG}(3) * (r)^2]$$

$$* [\text{FUEL2G}(1) + \text{FUEL2G}(2) * t + \text{FUEL2G}(3) * (t)^2],$$

in which r is the part-load ratio and t is the ambient air temperature. This is illustrated in the following diagram.



$f_1 = C_0{}^1 + C_1{}^1\ x_1 + C_2\ x_2{}^2$

$f_2 = C_0{}^2 + C_1{}^2\ x_2 + C_2{}^3\ x_2{}^2$

Fuel energy input = electric energy output * f_1 * f_2.

c. Thermodynamic Functions. Thermodynamic functions, such as SATUR and ENTROP, calculate thermodynamic properties needed by the other routines. These functions are obtained from either ASHRAE or NBS literature.

d. Load Allocation Subroutines. Load allocation routines, such as OPCOOL, OPELEC, OPDBUN, and LDIST, allocate loads among those types of equipment available to satisfy them. The optimization rules employed are described in Sec. V.B.

e. Utility Subroutines. Utility subroutines include those that read data, obtain default values, or print report data, such as DESVAR and REPORT. Others accumulate statistics, such as STATIS, or manipulate block data, or enter default or initial values or constants, such as DFLTE and DFLTP.

2. Description by Subroutine. Each PLANT subroutine is detailed below in common format: general description, data description, calculation procedure, ASHRAE verification, and functional and/or flow charts. Subroutines are discussed in alphabetical order.

a. CCBTEPS.

Description

CCBTEPS is the main program that directs the logical flow, inputs data, initializes variables, and controls the hourly, daily, and monthly simulation.

Subprograms Calling This Routine

SUBROUTINE EXEC in PROGRAM CALERDA*

Subprograms Called by This Routine

SUBROUTINES	ABSREF
	BOILER
	COMREF
	DBUNDLE
	DESVAR
	DIESEL
	EFFIC
	ENSTOR
	GASTUR
	HEATREC
	LDIST
	OPCOOL
	OPELEC
	REPORT
	SOLAR SIMULATOR
	STATIS
	STATSM
	STMTUR
	STMUSE
	TOWER
FUNCTION	WETBULB

Common Blocks

DATE, EDATA, EFFICD, EPARS, HOURTOT, MONTOT, SDATA, STM, STMTUR, TOWERD, WEATHR, STATD

All of these common blocks are used to pass through data to and from subroutines within this main routine.

* Not implemented as of August 1, 1977; CCBTEPS is called using a control card.

Declarations

```
EQUIVALENCE (OPRTOWR, SDATA(1,7)), (HSTEAM, SDATA(1,1)), (TSATUR,
SDATA(1,2)), (PELTWR, SDATA(1,6)), (PELCL, SDATA(1,4)), (PELHT,
SDATA(1,5)), (PEXSTUR, SDATA(1,34)), (RPMNOM, SDATA(1,35))
DATA MONLEN/31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31/
DATA NVAR/8/
```

Input

Energy load file from Tape 20 (generated by SYSTEMS Program and called SYSHRO), ENGYLD (I,J,K), where I = load or weather data type index, J = day index, k = hour index. This file is an unformatted binary file consisting of 365 records, each containing data for 24 hrs.

Name	Description
ENGYLD (I,J,K)	Preheat coil load (in MBtu) for I = 1; Heating coil load (MBtu) for I = 2; Total reheat coil load (MBtu) for I = 3; Preheat coil air inlet temperature (°F) for I = 4; Heating coil air inlet temperature (°F) for I = 5; Reheat coil air inlet temperature (average of all reheat coils) (°F) for I = 6; ECOOL, the cooling load (in MBtu) for I = 7; and EELEC, the electrical load (in kWh) for I = 8.

The total heating load, EHEAT, is the sum of ENGYLD (I,J,K) elements for I = 1, 2, and 3.

Plant Data Array from the plant input processor, named PDATA (I,J), where I = index of the coefficient in the quadratic polynominal function whose name appears in PDATA (4,N), and J = parameter type index.

Source of Data	Name	Description
PDATA(1,1)	CAVL1A	Available capacity (one-stage absorption chiller)
PDATA(1,2)	CAVL2A	Available capacity (two-stage absorption chiller)
PDATA(1,3)	REN1A	Energy input/output (I/O) coefficients (one-stage absorption chiller)
PDATA(1,4)	REN2A	Energy I/O coefficients (two-stage absorption chiller)
PDATA(1,5)	REN2AE	Energy I/O coefficients (two-stage absorption chiller with economizer)
PDATA(1,6)	TCON1A	Condensate temperature coefficient (one-stage absorption chiller)
PDATA(1,7)	RPWR1C	Energy I/O coefficients (hermetic compression chiller)
PDATA(1,8)	RPWR2C	Energy I/O coefficients (open centrifugal compression chiller)

PDATA(1,9)	RPWR3C	Energy I/O coefficients (reciprocating compression chiller)
PDATA(1,10)	RCAVDB	Available capacity ratio (double-bundle chiller)
PDATA(1,11)	RPWRDB	Energy I/O coefficients (double-bundle chiller)
PDATA(1,12)	ADJTDB	Condensate cooling water temperature adjustment factor (double-bundle)
PDATA(1,13)	ADJEDB	Energy ratio adjustment factor (double-bundle chiller)
PDATA(1,14)	RELD	Power out/fuel input coefficients (diesel)
PDATA(1,15)	RJACK	Jacket heat/fuel input coefficients (diesel)
PDATA(1,16)	RLUBD	Lube heat/fuel input coefficients (diesel)
PDATA(1,17)	REXD	Exhaust heat/fuel input coefficients (diesel)
PDATA(1,18)	TEXD	Exhaust temperature coefficients (diesel)
PDATA(1,19)	FUEL1G	Fuel I/O coefficients 1-3 (gas turbine)
PDATA(1,20)	FUEL2G	Fuel I/O coefficients 4-6 (gas turbine)
PDATA(1,22)	FEXG	Exhaust flow coefficients (gas turbine)
PDATA(1,23)	TEX1G	Exhaust temperature coefficients 1-3 (gas turbine)
PDATA(1,24)	TEX2G	Exhaust temperature coefficients 4-6 (gas turbine)
PDATA(1,25)	FLUBG	Lube oil coefficients (gas turbine)
PDATA(1,26)	RF1	Rating factor temperature coefficients 1-3 (cooling tower)
PDATA(1,27)	RF2	Rating factor temperature coefficients 4-6 (cooling tower)
PDATA(1,28)	RF3	Rating factor temperature coefficients 7-9 (cooling tower)
PDATA(1,29)	RF4	Rating factor temperature coefficients 10-12 (cooling tower)
PDATA(1,30)	RF5	Rating factor temperature coefficients 13-15 (cooling tower)
PDATA(1,31)	RF6	Rating factor temperature coefficients 16-18 (cooling tower)
PDATA(1,32)	RFUELB	Energy I/O coefficients (steam boiler)
PDATA(1,33)	SR1DTA	Steam rate coefficients (one-stage absorption chiller
PDATA(1,34)	SR2DTA	Steam rate coefficients (two-stage absorption chiller)
PDATA(1,35)	TCON2A	Condensate temperature coefficients (two-stage absorption chiller)
PDATA(1,36)	RFSTUR	Steam flow coefficients (steam turbine)

PDATA(1,37)	UACD	Stack U-factor x area coefficients (diesel)
PDATA(1,38)	UACG	Stack U-factor x area coefficients (gas turbine)
PDATA(1,39)	RFR	Rating factor range coefficients (cooling tower)

Special Parameter Data Array from the PLANT input processor named SDATA(1,J), where I = parameter value index (I = 1), code (I = 2), or name (I = 3-6), and J = parameter type index.

SDATA(1,1)	HSTEAM	Steam enthalpy (Btu/lb)
SDATA(1,2)	TSATUR	Saturation temperature (°F)
SDATA(1,3)	RFLASH	Boiler flash water/steam feed
SDATA(1,4)	PELCL	Electric input to circulation pump/cooling load
SDATA(1,5)	PELHT	Electric input to circulation pump/heating load
SDATA(1,6)	PELTWR	Electric input to cooling tower/tower cooling load
SDATA(1,7)	TOWOPR	Tower operation type
SDATA(1,9)	TWMAKE	Make-up water temperature (°F)
SDATA(1,10)	TCOOL	Chilled water temperature (°F)
SDATA(1,11)	DTCOOL	Chilled water temperature (°F)
SDATA(1,12)	TTOWR	Entering tower water temperature (°F)
SDATA(1,13)	TCW	Leaving condenser water temperature (°F)
SDATA(1,14)	TMINH	Minimum tank temperature for heating (°F)
SDATA(1,15)	TMINC	Minimum tank temperature for cooling (°F)
SDATA(1,16)	CPTYPE	Plant type (1 = utility only; 2 = mixed plant)
SDATA(1,17)	TLEAVE	Boiler stack leaving temperature (°F)
SDATA(1,18)	SR2A	Full-load steam rate (lb/hr) (two-stage absorption chiller)
SDATA(1,19)	SR1A	Full-load steam rate (lb/hr) (one-stage absorption chiller)
SDATA(1,20)	RAVRHDB	Available recoverable heat ratio
SDATA(1,22)	RMXKWD	Maximum exhaust flow/kW input (diesel)
SDATA(1,24)	RMXKWG	Maximum exhaust flow/kW input (gas)
SDATA(1,25)	RWCA	Tower water/absorption chiller capacity
SDATA(1,26)	RWCC	Tower water/compression chiller capacity
SDATA(1,27)	RWCDB	Tower water/double-bundle chiller capacity
SDATA(1,28)	SRATB	Air fuel stoichiometric ratio
SDATA(1,29)	HFUELB	Heat content of fuel (Btu/lb)
SDATA(1,30)	RHFLASH	Recovered heat/flash steam energy
SDATA(1,31)	PSTEAM	Steam pressure (psi)
SDATA(1,32)	PSTMTUR	Entering steam pressure (psi)

SDATA(1,33)	TSTMTUR	Entering steam temperature (°F)
SDATA(1,34)	PEXSTUR	Nominal exhaust steam pressure (psi)
SDATA(1,35)	RPMNOM	Nominal speed (rpm)
SDATA(1,36)	RWSTUR	Condensate/entering steam
SDATA(1,37)	TOTUEF	Total efficiency of utility electric generation

Output

Name	Description
PRNTA1(IMON,1)	Total heat energy (Btu/month)
PRNTA1(IMON,2)	Total electric energy (Btu/month)
PRNTA1(IMON,3)	Cooling electric energy (Btu/month)
PRNTA1(IMON,4)	Recovered energy (Btu/month)
PRNTA1(IMON,5)	Wasted recoverable energy (Btu/month)
PRNTA1(IMON,6)	Heat energy input for cooling (Btu/month)
PRNTA1(IMON,7)	Electric energy input for cooling (Btu/month)
PRNTA1(IMON,8)	Energy input for heating (Btu/month)
PRNTA1(IMON,9)	Energy input for electricity (Btu/month)
PRNTA1(IMON,10)	Total fuel input (Btu/month)
PRNTA1(IMON,11)	Total energy input (Btu/month)
PRNTA1(IMON,12)	Average plant efficiency (fraction)

(IMON = 13 represents total annual sum)

IENAME(2-4,I)	Equipment name of type I (30H Format)
AVGOPR	Average operation ratio
OPCAPY(I)	Operating capacity totaled over the year for equipment type I (Btu)
AMAXLD(I)	Maximum load of equipment type I (Btu)
MAXTIM(1,I)	Month of maximum part load for equipment type I
MAXTIM(3,I)	Hour of maximum part load for equipment type I
CNOM(J,I)	Nominal size of equipment type I, size index J (Btu/hr)
IOPRHR(J,I)	Number of operation hours for equipment type I, size index J
KINS(J,I)	Number of equipment of type I, size index J that are installe
KAV(J,I)	Number of equipment of type I, size index J that are available
IUNAM(IU)	Utility type IU, energy name
ENUSE(13,IU)	Annual energy used for utility type IU (MW)
ENPEAK(13,IU)	Annual energy peak for utility type IU (Btu)

Calculation Procedure

First, variables are initialized, and then the input data are read. The basic calculational sequence that is repeated is the hourly plant simulation loop illustrated in Fig. V.2. This hourly loop is repeated until the end of the month is encountered; then monthly energy consumption summaries and statistics are computed and written on the report file.

ASHRAE Verification

There is no ASHRAE algorithm applicable to the CCBTEPS routine; ASHRAE verification appears in the appropriate subroutine write-ups that follow.

b. Subroutine ABSREF.

SUBROUTINE ABSREF (ECOOL, FSTEAM, ESTEAM, EELEC, ETOWER, TCOND)

Description

ABSREF is a subroutine that simulates operation of a steam-fired absorption chiller. Three types of absorption chillers are included: a one-stage absorption chiller, a two-stage absorption chiller without economizer, and a two-stage absorption chiller with economizer.

Parameters relating to output variables are evaluated as follows:

(1) Ratio of (available capacity)/(nominal capacity) = $f_1(x_1,c^1)$,

where f_1 = a quadratic polynomial in x_1, and c_1 is a set of three polynomial coefficients;

x_1 = (condenser water inlet temperature - chilled water outlet temperature) (°F) = TTOWR - TCOOL;

c^1 = CAVL1A, for one-stage absorption, and
= CAVL2A, for two-stage absorption.

Note that each set of coefficients applies to a particular machine, particular cooling and chilled water flow rates, and a particular steam supply pressure.

(2) Ratio of (energy input)/(design input) = $f(x_1,c^1)$,

where f_1 = as defined in (1) above;

x_1 = part-load ratio
= (cooling load)/(available capacity);

c^1 = REN1A for one-stage absorption,
= REN2A for two-stage absorption without economizer,
= REN2AE for two-stage absorption with economizer.

(3) Part-load steam rate = $f_1(x_1,c^1) * f_2(x_2,c^2)$/(part-load ratio),

where f_1 = design-load steam rate (lbs/ton-hr), as defined in (1) above;

x_1 = chilled water temperature rise (°F),
= DTCOOL;

C^1 = SR1DTA for one-stage absorption,
= SR2DTA for two-stage absorption; and

f_2 = ratio of (energy input)/(design energy input), [(2) above] and part-load ratio = cooling load/available capacity.

Thus, the steam rate is the full-load steam rate adjusted for part-load conditions and for chilled water temperatures other than nominal. It is implicit that the coefficients SR1DTA and SR2DTA are for a particular cooling water temperature.

(4) Ratio of (condensate water temperature)/(steam saturation temperature)

$$= f_1(x_1, c^1),$$

where f_1 = as defined in (1) above;

x_1 = part-load ratio
= (cooling load)/(available capacity); and

c^1 = TCON1A for one-stage absorption
= TCON2A for two-stage absorption.

Note that any set of the above performance parameters is specific to a given supply steam pressure, chilled and condenser water flow rates, nominal fouling factors for all heat exchangers, and condenser water temperature rise.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

None

Common Blocks

EDATA, EPARS, PDATA, SDATA

Declarations

```
DIMENSION CAVL1A(8), CAVL2A(8), REN1A(8), REN2A(8), REN2AE(8), TCON1A(8),
          TCON2A(8), SR1DTA(8), SR2DTA(8)

EQUIVALENCE (CAVL1A, PDATA(1,1)), (CAVL2A, PDATA(1,2)), (REN1A, PDATA(1,3)),
            (REN2A, PDATA(1,4)), (REN2AE, PDATA(1,5)), (TCON1A, PDATA(1,6)),
            (TCON2A, PDATA(1,35)),
            (SR1DTA, PDATA(1,35)), (SR2DTA, PDATA(1,34))

EQUIVALENCE (HSTEAM, SDATA(1,1)), (TSATUR, SDATA(1,2)), (SR1A, SDATA(1,19))
            (SR2A, SDATA(1,18)), (DTCOOL, SDATA(1,11)), (TCOOL, SDATA
            (1,10)), (TTOWR, SDATA(1,12))
```

Input

Source of Data	Name	Description
CCBTEPS (PLOAD(IABSOR))	ECOOL	Total cooling energy (Btu/hr)

/EPARS/	IABSOR	Types of absorption chillers: 5 for one-stage absorption 6 for two-stage absorption without economizer 7 for two-stage absorption with economizer
SDATA(1,1)	HSTEAM	Steam enthalpy (Btu/lb)
SDATA(1,2)	TSATUR	Steam saturation temperature (°F)
SDATA(1,10)	TCOOL	Chilled water temperature (°F)
SDATA(1,11)	DTCOOL	Chilled water temperature rise, (°F)
SDATA(1,12)	TTOWR	Temperature of water leaving cooling tower and entering condenser (°F)
SDATA(1,18)	SR2A	Full-load steam rate for two-stage absorption chiller (lbs/ton-hr)
SDATA(1,19)	SR1A	Full-load steam rate for one-stage absorption chiller (lbs/ton-hr)
PDATA(1,1)	CAVL1A	Quadratic polynomial coefficients for one-stage absorption as described in previous section
PDATA(1,2)	CAVL2A	Quadratic polynomial coefficients for two-stage absorption as described in previous section
PDATA(1,3)	REN1A	Quadratic polynomial coefficients for one-stage absorption as described in previous section
PDATA(1,4)	REN2A	Quadratic polynomial coefficients for two-stage absorption as described in previous section
PDATA(1,5)	REN2AE	Quadratic polynomial coefficients for two-stage absorption as described in previous section
PDATA(1,6)	TCON1A	Quadratic polynomial coefficients for one-stage absorption as described in previous section
PDATA(1,33)	SR1DTA	Quadratic polynomial coefficients for one-stage absorption as described in previous section
PDATA(1,34)	SR2DTA	Quadratic polynomial coefficients for two-stage absorption as described in previous section
PDATA(1,35)	TCON2A	Quadratic polynomial coefficients for two-stage absorption as described in previous section
/EDATA/	RMIN(IABSOR)	Minimum part-load ratio for absorption chiller type IABSOR

/EDATA/	PEL(IABSOR)	Electrical input to nominal capacity ratio for absorption chiller type IABSOR

Output

Name	Description
EELEC	Electrical energy input (Btu/hr)
ESTEAM	Steam energy input (Btu/hr)
ETOWER	Tower cooling load (Btu/hr)
FSTEAM	Steam flow rate (lbs/hr)
TCOND	Condensate water temperature (°F)

Calculation Procedure

1. Convert data.

   Nominal cooling capacity = CNA = OPCAP(IABSOR = 5, 6, or 7),

   where OPCAP = operating capacity of equipment,
   5 = one-stage absorption chiller,
   6 = two-stage absorption chiller, and
   7 = two-stage absorption chiller with economizer.

   Electrical power to pump = PNA = PEL(IABSOR) * nominal capacity.

2. Set initial conditions.

   Steam flow and energy, electrical input, and tower load = 0
   condensate temperature = steam saturation temperature

3. Available cooling capacity = CAVAIL = Nominal cooling capacity * [C(1) +C(2)*X+C(3)*X*X],

   where X = TTOWR - TCOOL,
   C = CAVL1A for one-stage, and
   C = CAVL2A for two-stage.

   Note that the default values for these coefficients assume that the available capacity equals the nominal capacity under all conditions.

4. Set chiller load to cooling load subject to lower and upper bound.

   Minimum chiller load = RMIN(IABSOR) * available cooling capacity
   Maximum chiller load = available cooling capacity = CAVAIL

5. Chiller design load = CDLOAD = available cooling capacity = CAVAIL.

6. Set electrical load.

   Electrical input = EELEC = electrical power to pump = PNA
   (assumes electrical consumption of pump is independent of load)

7. Design-load ratio = RLOAD = design load/nominal capacity = CDLOAD/CNA.

8. Part-load ratio = RPL = chiller load/chiller design load, where the chiller design load is assumed to be equal to the available capacity at the hour.

   Note that the conventional definition of the part-load ratio is taken as the fraction of the nominal capacity. Here the design load is used.

9. Design-load steam rate = SR = full-load steam rate
* [C(1) + C(2) * X + C(3) * X * X],

where x full-load steam rate = nominal steam rate
= SR1A or SR2A
= DTCOOL,
C = SR1DTA for one-stage, and
C = SR2DTA for two-stage.

10. Calculate energy input ratio.

a. One-stage:

Input energy/full-load input energy = REN = [C(1) + C(2) * X
+ C(3) * X * X],

where X = RPL and
C = REN1A.

b. Two-stage:

If nominal capacity (=CNA) > 1400 tons, an economizer is automatically put into operation whenever the part-load ratio < 0.5. If the nominal capacity is < 1400 tons, an economizer is in operation only if the chiller specified has an economizer available. Thus, input energy/full-load input energy
= REN = [C(1) + C(2) * X + C(3) * X * X],

where X = RPL,
C = REN2AE, with economizer, and
C = REN2AE, without economizer.

11. Condensate temperature = TCOND = steam saturation temperature *[C(1)
+ C(2) * X + C(3) * X * X],

where X = RPL,
C = TCON1A for one-stage, and
C = TCON2A for two-stage

12. Part-load steam rate = SRPL = full-load steam rate * (energy input ratio/
part-load ratio)
= SR * REN/RPL.

13. Steam flow rate = FSTEAM = chiller load * part-load steam rate
= CLOAD * SRPL/12000.

14. Steam energy input = ESTEAM = Steam flow rate * (steam enthalpy condensate
temp + 32°F)
= FSTEAM * (HSTEAM-TCOND+32.)

15. Cooling tower load = ETOWER
= cooling load + electrical input + steam energy input
= ECOOL + EELEC + ESTEAM.

ASHRAE Verification

ASHRAE documentation (Ref. 2) expresses absorption chiller cooling capacity as a quadratic equation in two variables: condenser water inlet temperature and chilled water outlet temperature. However, Cal-ERDA expresses the capacity relationship as a quadratic in the difference between these same two temperatures. The difference between these two expressions is that the Cal-ERDA equation neglects higher order (≥ 3) terms.

The ASHRAE document recommends an equation to correct for chilled water flow rates differing from the reference (default) value. Cal-ERDA includes no such correction; the user must take this into account when using manufacturer's flow rate correction charts. A different set of coefficients must be computed for other than nominal flow rates.

Reference 2 gives the part-load steam rate as the product of quadratic equations in the condenser water inlet temperature and the chilled-water outlet temperature and in the ratio of actual-to-nominal capacity. The coefficients are determined from test data that are appropriate to this formulation.

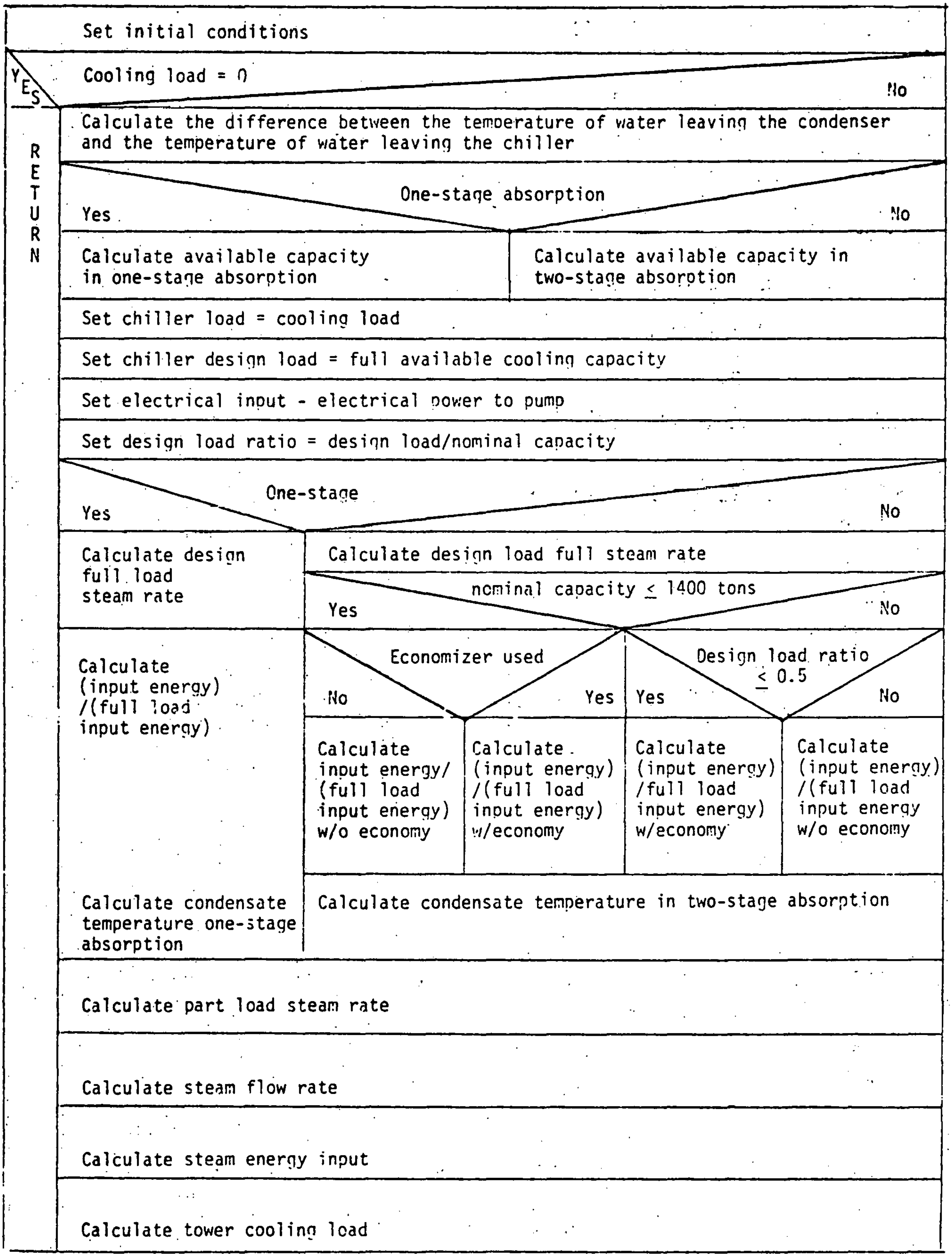
On the other hand, in Cal-ERDA the part-load steam rate is computed by first adjusting the full-load (nominal) steam rate to design-load (off-nominal) conditions. Next the design-load steam rate is adjusted for part-load conditions using the part-load ratio.

In this part-load steam rate calculation, Cal-ERDA assumes that the part-to-design load steam rate ratio is equal to the part-to-design load energy input ratio. The latter ratio is computed as a quadratic equation in the part-load ratio.

Note that Cal-ERDA expresses the full-load (nominal) steam rate adjustment for off-nominal operation as a quadratic in the chilled water temperature rise. Thus Cal-ERDA makes no steam rate adjustment for off-nominal condenser water inlet temperatures. It is implicit that the quadratic coefficients SR1DTA and SR2DTA are for a particular cooling water temperature.

Finally, Cal-ERDA expresses the ratio of condensate water temperature to steam saturation temperature as a quadratic equation in the part-load ratio. The ASHRAE documentation does not discuss this point.

ABSREF Flow Chart


Set initial conditions
Cooling load = 0
YES
No
RETURN
Calculate the difference between the temperature of water leaving the condenser and the temperature of water leaving the chiller
One-stage absorption
Yes
No
Calculate available capacity in one-stage absorption
Calculate available capacity in two-stage absorption
Set chiller load = cooling load
Set chiller design load = full available cooling capacity
Set electrical input - electrical power to pump
Set design load ratio = design load/nominal capacity
One-stage
Yes
No
Calculate design full load steam rate
Calculate design load full steam rate
nominal capacity ≤ 1400 tons
Yes
No
Calculate (input energy) /(full load input energy)
Economizer used
No
Yes
Design load ratio ≤ 0.5
Yes
No
Calculate input energy/ (full load input energy) w/o economy
Calculate (input energy) /(full load input energy) w/economy
Calculate (input energy) /full load input energy) w/economy
Calculate (input energy) /(full load input energy w/o economy
Calculate condensate temperature one-stage absorption
Calculate condensate temperature in two-stage absorption
Calculate part load steam rate
Calculate steam flow rate
Calculate steam energy input
Calculate tower cooling load

c. Subroutine BOILER.

SUBROUTINE BOILER (EBLNET, EFUELB)

Description

Subroutine BOILER simulates a fuel-fired steam boiler. The type of boiler fuel is implied by the values of the special parameters HFUELB and SRATB. The parameter TLEAVE is also a special parameter.

The boiler fuel energy input rate = (full-load boiler fuel energy input rate)/$f_1(x_1,c^1)$,

where f_1 = a quadratic polynomial in x_1, and c^1 is a set of three polynomial coefficients;

x_1 = part-load ratio = (net energy output)/(operating capacity), and

c^1 = RFUELB.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

None

Common Blocks

EDATA, EPARS, PDATA, SDATA, WEATHR

Declarations

```
DIMENSION RFUELB (8)
EQUIVALENCE (RFUELB, PDATA(1,32))
EQUIVALENCE (TLEAVE, SDATA(1,17)), (SRATB, SDATA(1,28)), (HFUELB, SDATA
  (1,29))
```

Input

Source of Data	Name	Description
CCBTEPS (PLOAD(4))	EBLNET	Boiler net energy output rate (Btu/hr)
/WEATHR/	TAIR	Ambient air temperature (°F)
/WEATHR/	HR	Humidity ratio
SDATA(1,17)	TLEAVE	Boiler leaving stack temperature (°F)
SDATA(1,28)	SRATB	Air-to-fuel stoichiometric ratio
SDATA(1,29)	HFUELB	Heat content of fuel (Btu/lb)
PDATA(1,32)	RFUELB	Quadratic polynomial coefficients for part-load ratio as described in previous section
/EPARS/	OPCAP(4)	Operating capacity of boiler
/EDATA/	RMIN(4)	Minimum part-load ratio for boiler

Output

Name	Description
EFUELB	Boiler fuel energy input rate (Btu/hr)

Calculation Procedure

1. Initialize output variable: EFUELB = 0.
2. If boiler net energy output requirement = EBLNET = 0, skip calculation and return; otherwise,
3. Calculate the full-load boiler energy input rate using the net energy output, the fuel heat content, the temperature rise of the combustion air (taking account of humidity), and the stoichiometric ratio.

   EFLB = EBLNET/[0.87-(1.25*SRATB/HFUELB) * ((TLEAVE-TAIR) * (0.25+HR) + 1000. *HR)]
4. Calculate part-load ratio as (net energy output)/(operating capacity) subject to a lower bound.

   RB = AMAX1(EBLNET/OPCAP(4), RMIN(4))
5. Calculate boiler fuel input rate using quadratic equation coefficients,

   EFUELB = EFLB/[C(1)+C(2) * RB + C(3) * RB * RB],

   where C = RFUELB.

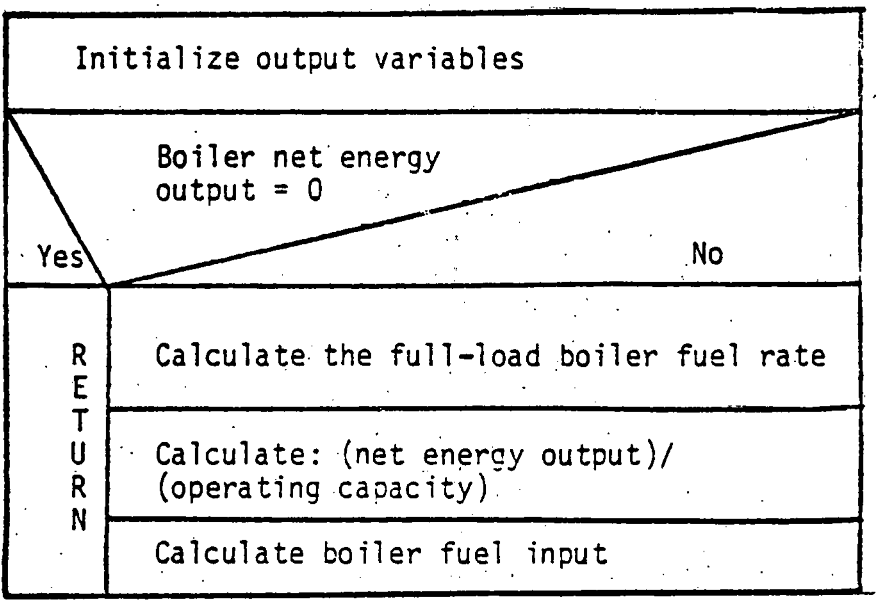
ASHRAE Verification

The ASHRAE documentation (Ref. 2) expresses boiler efficiency, defined as the energy transferred to the heated water divided by the heating capacity of the fuel, as a quadratic polynomial in the part-load ratio. The part-load ratio is defined as the fraction of full load.

Cal-ERDA uses a different formulation. First, the full-load energy input rate is calculated wherein the net energy output rate is divided by an empirical expression that adjusts for the operating conditions of fuel heat content, temperature rise of the combustion air, and the air-to-fuel stoichiometric ratio. The expression used is an empirical correction factor developed by the CCB.

Then the boiler fuel input rate is computed by dividing the full-load energy input rate by a quadratic polynomial in the part-load ratio. The part-load ratio is defined as in Ref. 2. This quadratic is not an efficiency defined in the conventional manner. However, it relates the full-load theoretical energy input to the actual energy input.

BOILER Flow Chart


Initialize output variables
Boiler net energy output = 0
Yes
No
RETURN
Calculate the full-load boiler fuel rate
Calculate: (net energy output)/ (operating capacity)
Calculate boiler fuel input

d. Subroutine COMREF.

SUBROUTINE COMREF (ECOOL, EELEC, ETOWER)

Description

Subroutine COMREF is used to simulate operation of a compression chiller. Three types of compression chillers are included: the hermetic centrifugal chiller, the open centrifugal chiller, and the reciprocating chiller.

The chiller input/output relationship is given as:

Fraction of nominal chiller electrical energy input

= (energy input)/(nominal energy input) = $f_1(x_1;c^1)$,

where f_1 = a quadratic polynomial in x_1, and c^1 is a set of polynomial coefficients;

x_1 = part-load ratio
= (required cooling)/(nominal cooling capacity); and

c^1 = RPWR1C, for hermetic centrifugal chiller
= RPWR2C, for open centrifugal chiller
= RPWR3C, for reciprocating chiller.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

None

Common Blocks

EDATA, EPARS, PDATA

Declarations

```
DIMENSION RPWR1C(8), RPWR2C(8), RPWR3C(8)
EQUIVALENCE (RPWR1C, PDATA(1,7)), (RPWR2C, PDATA(1,8)), (RPWR3C, PDATA
  (1,9))
```

Input

Source of Data	Name	Description
CCBTEPS (PLOAD(ICOMPR))	ECOOL	Required cooling for compression type ICOMPR
/EPARS/	ICOMPR	Compression chiller type: 9, for hermetic centrifugal chiller 10, for open centrifugal chiller 11, for reciprocating centrifugal chiller
PDATA(1,7)	RPWR1C	Quadratic polynomial coefficients for hermetic centrifugal chiller as described in previous section
PDATA(1,8)	RPWR2C	Quadratic polynomial coefficients for open centrifugal chiller as described in previous section

PDATA(1,9)	RPWR3C	Quadratic polynomial coefficients for reciprocating chiller as described in previous section
/EDATA/	RMIN(ICOMPR)	Minimum part-load ratio of compression chiller type ICOMPR
/EDATA/	RMAX(ICOMPR)	Maximum part-load ratio of compression chiller type ICOMPR

Output

Name	Description
EELEC	Electrical energy input (Btu/hr)
ETOWER	Cooling tower load (Btu/hr)

Calculation Procedure

1. Convert data.

   Nominal cooling capacity = CNC = OPCAP(ICOMPR = 9, 10, or 11)
   where OPCAP = nominal operating capacity of equipment (Btu/hr)

   Nominal electrical power input = PNC = PEL(ICOMPR) * CNC (Btu/hr)
   where PEL(ICOMPR = electrical input to nominal capacity ratio

2. Initialize output variables.

   Electrical input and tower load = 0.
   EELEC = ETOWER = 0.

3. If no cooling is required, i.e., ECOOL $\leq$ 0, skip the following calculation and return; otherwise,

4. Set chiller load to required cooling load subject to a lower bound.

   Chiller load $\geq$ RMIN(ICOMPR) * CNC
   CLOUD2 = AMAX1(ECOOL,RMIN(ICOMPR*CNC)
   CLOUD = AMIN1(CLOUD2, RMAX(ICOMPR*CNC)

5. Calculate part-load ratio as the fraction of nominal capacity.

   RLOAD = CLOAD/CNC

6. Calculate fraction of nominal power input as a quadratic polynomial in the part-load ratio. Use the energy input/output coefficients that are appropriate to the type of chiller being used.

   RPWER = C(1) + C(2) * RLOAD + C(3) * RLOAD * RLOAD,

   where C = RPWR1C for a hermetic centrifugal chiller,
   C = RPWR2C for an open centrifugal chiller, and
   C = RPWR3C for a reciprocating chiller.

7. Electric energy input is calculated.

   EELEC = RPOWER * PNC

8. Calculate the cooling tower load resulting from an energy balance on the chiller.

   ETOWER = EELEC + ECOOL

ASHRAE Verification

Centrifugal Chillers

ASHRAE documentation (Ref. 2) expresses centrifugal compression chiller cooling capacity as a quadratic equation in two variables: condenser water inlet temperature and chilled water outlet temperature. However, Cal-ERDA makes no adjustment of nominal capacity for either of these temperatures. Therefore, the user must enter the nominal chiller capacity appropriate to specific condenser water inlet and chilled water outlet temperatures. If these conditions change significantly, a different nominal capacity must be used.

The ASHRAE document recommends correction factors for heat exchanger fouling factors and the chiller water temperature rise (or flow rate). Cal-ERDA includes no such correction. A different nominal capacity must be used for other than nominal values of these variables.

Reference 2 gives the part-load input power requirement as the product of quadratic equations in the condenser water inlet temperature and the part-load ratio and a linear equation in the chilled water outlet temperature. The coefficients are determined from test data that are appropriate to this formulation.

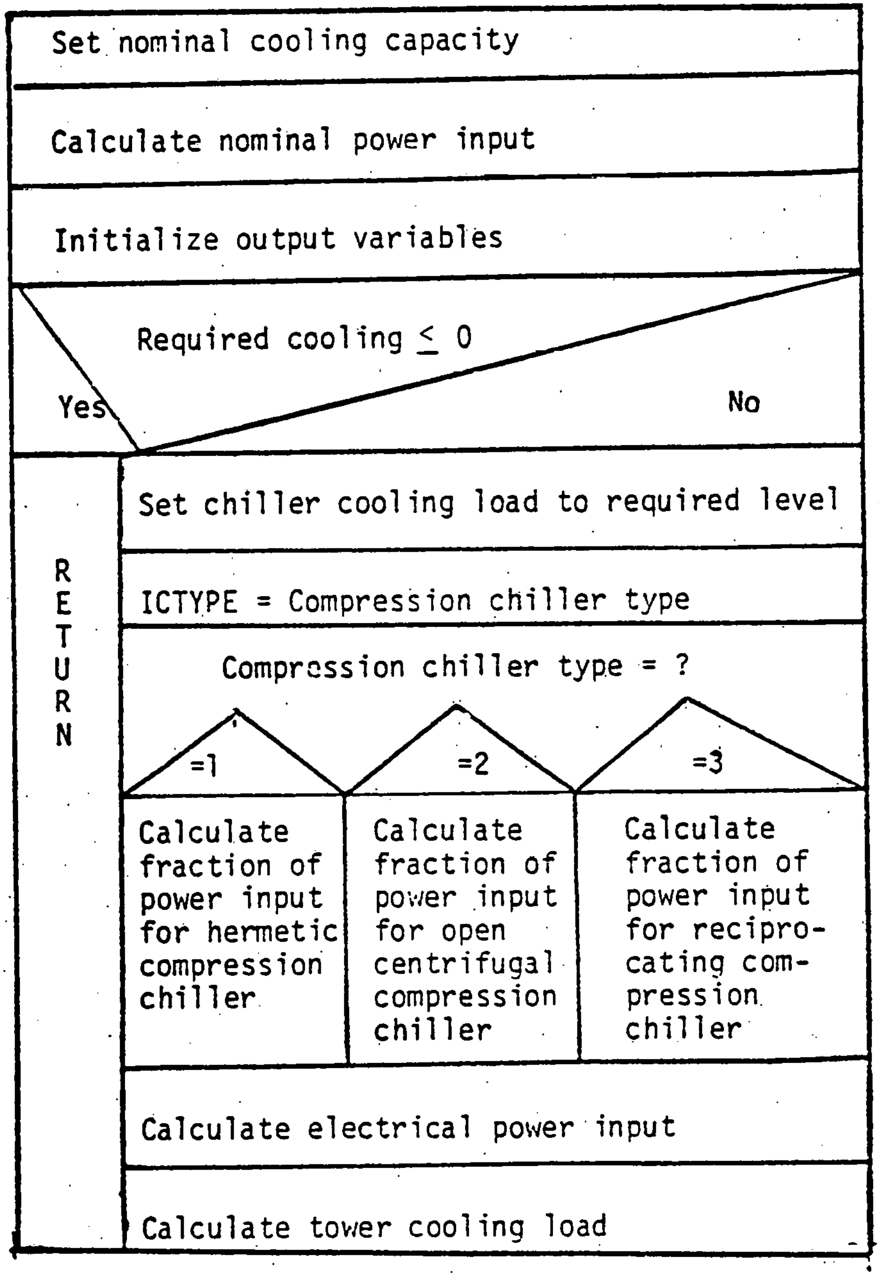
On the other hand, in Cal-ERDA no adjustment of input/output characteristics is made for either condenser-water inlet temperature or chilled-water outlet temperature. Therefore, the user must enter part-load performance coefficients appropriate to specific condenser-water inlet and chilled-water outlet temperatures. The Cal-ERDA formulation of input/output characteristics is in terms of a quadratic equation in the part-load ratio (defined as in Ref. 2); this is identical to the ASHRAE approach.

Reciprocating Chillers

Reference 2 expresses reciprocating compression chiller cooling capacity as a quadratic equation in evaporating temperature and condensing temperature. Since these temperatures are characterized by the condenser-water inlet and chilled-water outlet temperatures, respectively, and the Cal-ERDA performance formulation is the same here as for centrifugal chillers, the same comments as appear in the centrifugal chiller section above apply.

Likewise, Ref. 2 gives the part-load input power requirements as a quadratic equation in the same two temperatures. Since Cal-ERDA makes no adjustment for these temperature conditions, the same comments as appear in the centrifugal chiller section above apply.

COMREF Flow Chart


Set nominal cooling capacity
Calculate nominal power input
Initialize output variables
Required cooling ≤ 0
Yes
No
RETURN
Set chiller cooling load to required level
ICTYPE = Compression chiller type
Compression chiller type = ?
=1
=2
=3
Calculate fraction of power input for hermetic compression chiller
Calculate fraction of power input for open centrifugal compression chiller
Calculate fraction of power input for reciprocating compression chiller
Calculate electrical power input
Calculate tower cooling load

e. Subroutine DBUNDLE.

SUBROUTINE DBUNDLE (ECOOL, EHEAT, EELEC, ETOWER, ERCVCD, EWASTCD, RCAV)

Description

DBUNDLE simulates the operation of a double-bundle compression chiller. Parameters relating to output variables are evaluated as follows:

(1) Available cooling capacity ratio, RCAV

= ratio of (available cooling capacity)/(nominal capacity)

= $f_1(x_1, c^1)$,

where f_1 = a quadratic polynomial in x_1, and c^1 is a set of three polynomial coefficients;

x_1 = Z, as defined below;

c^1 = RCAVDB; and

Z = (condenser water outlet temperature-C(1))/C(2)-(chilled water outlet temperature-C(3));

where C = ADJTDB.

The variable Z is a condenser water temperature adjustment factor that is nonzero only when there is a heating requirement. This reflects the fact that the available capacity depends on whether there is a heating requirement because the condenser water outlet temperature may need to be increased to meet this heating requirement.

(2) Energy ratio adjustment factor, G

= (design-load input energy)/(nominal [full]-load input energy)

= $f_2(x_2,c^2)$;

where f_2 = a quadratic polynominal in x_2, and c^2 is a set of three polynomial coefficients;

x_2 = RCAV = f_1, as defined in (1) above; and

c^2 = ADJEDB.

(3) Ratio of (input energy)/(design-load input energy) = $f_1(x_1,c^1)$,

where f_1 = as defined in (1) above,

x_1 = (cooling load)/(nominal capacity); and

c^1 = RPWRDB.

(4) Available recoverable heat

= (cooling load + electrical energy input) * RAVRHDB.

where RAVRHDB = available recoverable heat ratio, a special parameter.

Subprograms Calling This Routine

SUBROUTINE OPDBUN

Subprograms Called by This Routine

None

Common Blocks

EDATA, EPARS, PDATA, SDATA

Declarations

```
DIMENSION RCAVDB(8), RPWRDB(8), ADJTDB(8), ADJEDB(8)
EQUIVALENCE (RCAVDB, PDATA(1,10)), (RPWRDB, PDATA(1,11)), (ADJTDB,
            PDATA(1,12)), (ADJEDB, PDATA(1,13))
EQUIVALENCE (CNCD, OPCAP(13)), (TCOOL, SDATA(1,10)), (TCW, SDATA(1,13)),
            (RAVRHDB, SDATA(1,20))
```

Input

Source of Data	Name	Description
CCBTEPS (PLOAD(13))	ECOOL	Cooling load for double-bundle chiller (Btu/hr)
CCBTEPS (ABS(ENGYLDC1, IHR, IDAY))	EHEAT	Heating load (Btu/hr)
/EPARS/ (OPCAP(13))	CNCD	Nominal cooling capacity for double-bundle chiller (Btu/hr)
/EDATA/	PEL(13)	Electrical input-to-nominal capacity ratio for double-bundle chiller
/EDATA/	RMIN(13)	Minimum part-load ratio for double-bundle chiller
SDATA(1,10)	TCOOL	Chilled water outlet temperature (°F)
SDATA(1,13)	TCW	Condenser water outlet temperature (°F)
SDATA(1,20)	RAVRHDB	Available recoverable heat ratio
PDATA(1,10)	RCAVDB	Quadratic polynomial coefficients for available cooling capacity ratio as described in previous section
PDATA(1,11)	RPWRDB	Quadratic polynomial coefficients for energy I/O ratio as described in previous section
PDATA(1,12)	ADJTDB	Constants for the evaluation of variable Z as described in previous section
PDATA(1,13)	ADJEDB	Quadratic polynomial coefficients for energy ratio adjustment factor as described in previous section

Output

Name	Description
EELEC	Electrical energy input (Btu/hr)
ERCVCD	Recovered heat (Btu/hr)
ETOWER	Cooling tower load (Btu/hr)
EWASTCD	Wasted recoverable heat from double-bundle chiller (Btu/hr)
RCAV	Available cooling capacity ratio = (available cooling capacity)/(nominal capacity)

Calculation Procedure

1. Initialize output variables.

   EELEC = ETOWER = ERCVCD = EWASTCD = 0.
   RCAV = available cooling capacity ratio = 1.

2. If no cooling required, i.e., ECOOL $\leq$ 0, skip calculation and return; otherwise,

3. Initialize intermediate variable.

   Z = 0.

4. If heating is required, i.e., EHEAT > 0, calculate condenser water temperature adjustment factor.

   Z = [RCW-C(1)]/[C(2)-(TCOOL-C(3))],

   where C = ADJTDB.

5. Calculate available cooling capacity ratio.

   RCAV = C(1) + C(2) * Z + C(3) * Z * Z,

   where C = RCAVDB and

   Z = defined above.

6. Set the available cooling capacity ratio, RLOAD, subject to an upper bound, RCAV, and a lower bound, minimum fractional capacity = RMIN(13).

   RLOAD = AMIN1(ECOOL/CNCD, RCAV)
   RLOAD = AMAX1(RLOAD, RMIN(13))

7. Calculate energy ratio adjustment factor.

   G = [C(1)+C(2)*RCAV+C(3)*RCAV*RCAV],

   where C = ADJEDB.

8. Calculate energy I/o relationship.

   RPOWER = [C(1)+C(2)*RLOAD+C(3)*RLOAD*RLOAD],

   where C = RPWRDB.

9. Calculate electrical energy input to chiller.

   EELEC = RPOWER * G * PEL(13) * OPCAP(13),

   where OPCAP(13) = CNCD = nominal capacity for double-bundle chiller.

10. Calculate available recoverable heat as the total energy consumption of chiller * available recoverable heat ratio.

    EHAV = (RLOAD*CNCD+EELEC) * RAVRHDB.

11. Set recovered heat, EHRCV, to available recoverable heat subject to an upper bound; heat recovered $\leq$ heating load.

    EHRCV = AMIN1 (EHAV, EHEAT)

    Note that this waste heat is assumed usable only to meet the space heating load (see HEATREC).

12. Calculate cooling tower load resulting from an energy balance on the chiller.

    ETOWER = EELEC + ECOOL - EHRCV

13. Determine portion of recovered heat needed elsewhere at this hour, ERCVCD; that not needed is classified as wasted recoverable waste heat, EWASTCD.

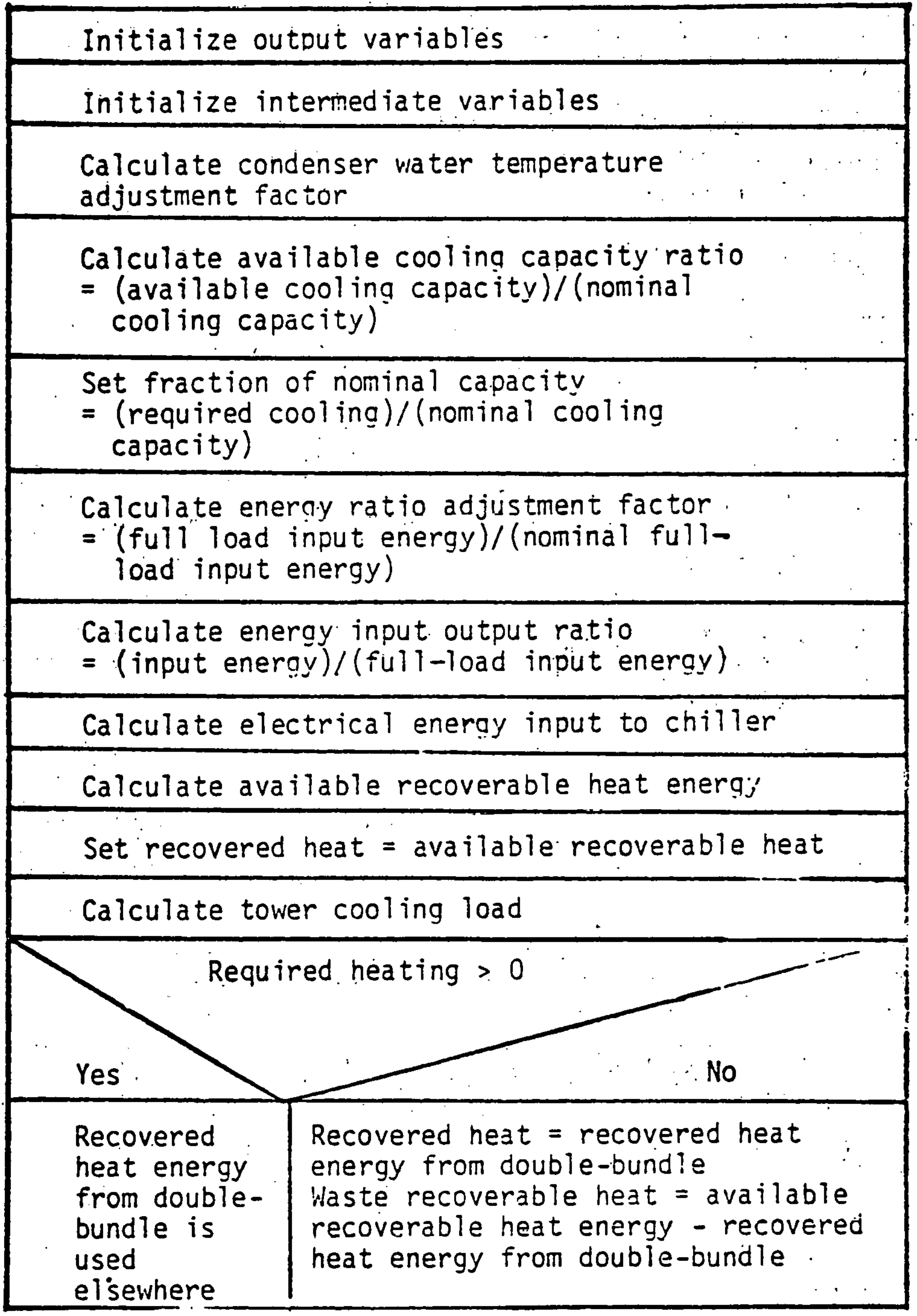
    IF (EHEAT. GT. 0.) ERCVCD = EHRCV
    IF (EHEAT. GT. 0.) EWASTCD = EHAV - EHRCV

    EWASTCD is assumed not storable.

ASHRAE Verification

The ASHRAE documentation (Ref. 2) does not discuss the simulation of double-bundle chillers. The Cal-ERDA routine is based on an empirical model developed by CCB.

DBUNDLE Flow Chart



Initialize output variables

Initialize intermediate variables

Calculate condenser water temperature adjustment factor

Calculate available cooling capacity ratio = (available cooling capacity)/(nominal cooling capacity)

Set fraction of nominal capacity = (required cooling)/(nominal cooling capacity)

Calculate energy ratio adjustment factor = (full load input energy)/(nominal full-load input energy)

Calculate energy input output ratio = (input energy)/(full-load input energy)

Calculate electrical energy input to chiller

Calculate available recoverable heat energy

Set recovered heat = available recoverable heat

Calculate tower cooling load

Required heating > 0

Yes

Recovered heat energy from double-bundle is used elsewhere

No

Recovered heat = recovered heat energy from double-bundle
Waste recoverable heat = available recoverable heat energy - recovered heat energy from double-bundle

f. Subroutine DESVAR.

Description

This subroutine initializes and reads input data.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

SUBROUTINE DFLTASG
SUBROUTINE KARDRD
SUBROUTINE LSEARC
SUBROUTINE R3PRNT

Common Blocks

CARD, CDPR, DISTB, ECARD, EDATA, EPARS, KEYLST, LDISTD, PDATA, PRNTA1, REPOPT, SDATA, STATD, TITLED, UCOSTD

Declarations

```
DIMENSION DISTBD(7,10), CD(35), IVNM(3), VVL(3)
EQUIVALENCE (HSTEAM, SDATA(1,1)), (TSATUR, SDATA(1,2)), (TLEAVE, SDATA
            (1,17)), (PSTEAM,SDATA(1,31)), (TSTMTUR, SDATA(1,33))

DATA IBLNK/10H/
DATA ISTOPF/0/
DATA KARD, IPRT, ICDS/5,9,8/
DATA NREPOP/14/
DATA IREPOPD/..../†
DATA KEYLIST/..../†
```

Input

Subroutine DESVAR reads or initializes all the data required for running CCBTEPS. Rather than including the description of the data here, see the data description in Sec. V.C, Input Data; Sec. V.E.1, CCBTEPS; and in each individual subroutine.

Output

DESVAR prints error messages for incorrect or unrecognizable input data and calls R3PRNT to print internal parameters.

Calculation Procedure

None

ASHRAE Verification

None

[†] These arrays are filled with alphabetic data for titles and labels for report printing.

g. Subroutine DFLTASG.

Description

DFLTASG is called by the master data input subroutine, DESVAR, to assign default values and postprocess input data.

Subprograms Calling This Routine

SUBROUTINE DESVAR

Subprograms Called by This Routine

FUNCTION ENTHAL
FUNCTION ENTROP
FUNCTION SATUR

Common Blocks

EDATA, EPARS, REPOPT, SDATA, STMTUR, TITLED, TOWERD

Declarations

```
EQUIVALENCE (TLEAVE, SDATA(1,17)), (HSTEAM, SDATA(1,1)), (TSATUR, SDATA
            (1,2)), (PSTEAM, SDATA(1,31)), (TSTMTUR, SDATA(1,33))
DIMENSION EQCPWR(10)
DATA EQCPWR/-0., .67, 0., .40, .20, .10, .20, .67, .20, .67/
DATA NEDATA/20/
DATA(IENAME(I),I=1,20)
```

The IENAME array is filled with alphabetic energy-type labels.

```
DATA RMIN(1), RMAX(1), ROPT(1), PEL(1)/.02, 1.05, .60, 0.1/
DATA RMIN(2), RMAX(2), ROPT(2), PEL(2)/.02, 1.05, .60, 0./
DATA RMIN(4), RMAX(4), ROPT(4), PEL(4)/.01, 1.0, .87, 0./
DATA RMIN(5), RMAX(5), ROPT(5), PEL(5)/.05, 1.10, .65, 7.677E-3/
DATA RMIN(6), RMAX(6), ROPT(6), PEL(6)/.05, 1.10, .65, 7.677E-3/
DATA RMIN(7), RMAX(7), ROPT(7), PEL(7)/.05, 1.10, .65. 7.677E-3/
DATA RMIN(9), RMAX(9), ROPT(9), PEL(9)/.10, 1.05, .65, .227457/
DATA RMIN(10), RMAX(10), ROPT(10), PEL(10)/.10, 1.05, .65. .227467/
DATA RMIN(11), RMAX(11), ROPT(11), PEL(11)/.10, 1.05, .65. .227467/
DATA RMIN(13), RMAX(13), ROPT(13), PEL(13)/.10, 1.05, .65. .227467/
DATA RMIN(14), RMAX(14), ROPT(14), PEL(14)/0., 0., .43652, .012/
DATA RMIN(15), RMAX(15), ROPT(15), PEL(15)/0., 0., .43652, .012/
DATA RMIN(16), RMAX(16), ROPT(16), PEL(16)/4*0./
DATA RMIN(17), RMAX(17), ROPT(17), PEL(17)/.02, 1.1, .9, 0./
DATA RMIN(18), RMAX(18), ROPT(18), PEL(18)/4*0./
DATA RMIN(19), RMAX(19), ROPT(19), PEL(19)/4*0./
DATA MTIT/4/, NTIT/0/, ITIT/32*10+1          /
```

Input

Source of Data	Name	Description
/EDATA/	NEQSIZE(I)	Same as described in CCBTEPS
/EDATA/	KAV(J,I)	Same as described in CCBTEPS
/EDATA/	CNOM(J,I)	Same as described in CCBTEPS
/EDATA/	ROPT(I)	Same as described in CCBTEPS
SDATA(1,1)	HSTEAM	Same as described in CCBTEPS

SDATA(1,2)	TSATUR	Same as described in CCBTEPS
SDATA(1,17)	TLEAVE	Same as described in CCBTEPS
SDATA(1,31)	PSTEAM	Same as described in CCBTEPS
SDATA(1,33)	TSTMTUR	Same as described in CCBTEPS

Output

Name	Description
RMIN(I)	Same as described in CCBTEPS
RMAX(I)	Same as described in CCBTEPS
ROPT(I)	Same as described in CCBTEPS
PEL(I)	Same as described in CCBTEPS
NEDATA	Same as described in CCBTEPS
IENAME(J,I)	Same as described in CCBTEPS
TOTCAP(I)	Total nominal capacity of equipment type I (Btu/hr)
IABSOR	Same as described in TOWER
ICOMPR	Same as described in TOWER
ITOWR	Same as described in TOWER
KT	Same as described in TOWER
PNTK	Same as described in TOWER
CNTU	Same as described in TOWER
CNT	Same as described in TOWER
CNTUH	Same as described in TOWER
CNTH	Same as described in TOWER
TLEAVE	Same as described in CCBTEPS
PSTEAM	Same as described in CCBTEPS
TSATUR	Same as described in CCBTEPS
HSTEAM	Same as described in CCBTEPS
TSTMTUR	Same as described in CCBTEPS
HSTMTUR	Same as described in CCBTEPS
SSTMTUR	Same as described in CCBTEPS

Calculation Procedure

None

ASHRAE Verification

Inapplicable

DFLTASG Flow Chart

Assign default values for reference cost to actual cost conversion parameters
Define variable names
Assign values for minimum part load ratio, RMIN(I), maximum part load ratio, RMAX(I), optimum part load ratio, ROPT(I), and electrical input to nominal capacity ratio, PEL(I)
Default SOLAR data
Calculate absorption chiller index, IABSOR
Calculate compression chiller index, ICOMPR
Calculate cooling tower index, ITOWR
Fill total capacity array, TOTCAP
Set up variables for cooling tower
Assign default values for TLEAVE, PSTEAM, TSATUR, HSTEAM, and TSTMTUR
Calculate specific enthalpy HSTMTUR, and specific entropy SSTMTUR for superheated high pressure steam
Fill equipment cost array EQCOSD from reference cost array EQCOSR

h. Subroutine DIESEL.

SUBROUTINE DIESEL (EELECD, EJACKD, ELUBED, EEXD, EFUELD)

Description

This subroutine simulates a super-charged diesel engine generator set. Parameters relating to output variables are evaluated as follows:

(1) Ratio of (electrical energy output)/(fuel energy input) = $f_1(x_1,c^1)$,

where f_1 = a quadratic polynomial in x_1, and c^1 is a set of polynomial coefficients;
x_1 = part-load ratio,
= (electrical power output)/(nominal electrical power capacity); and
c^1 = RELD.

(2) Ratio of (available jacket heat)/(fuel energy input) = $f_1(x_1,c^1)$,

where f_1 = as defined above;
x_1 = part-load ratio, as above; and
c^1 = RJACD.

(3) Ratio of (available lube oil heat)/(fuel energy input) = $f_1(x_1;c^1)$,

where f_1 = as defined above;
x_1 = part-load ratio, as above; and
c^1 = RLUBD.

(4) Ratio of (thermal energy of exhaust gases)/(fuel energy input) = $f_1(x_1,c^1)$,

where f_1 = as defined above;
x_1 = part-load ratio, as above; and
c^1 = REXD.

(5) Exhaust gas temperature (°F) = $f_1(x_1;c^1)$,

where f_1 = as defined above;
x_1 = part-load ratio, as above; and
c^1 = TEXD.

The exhaust gas stack U-factor-area product, stack gas temperature, and available recoverable exhaust heat are calculated as discussed in the calculation procedure below.

Subprograms Calling This Routine

SUBROUTINE OPCOOL

Subprograms Called by This Routine

None

Common Blocks

EDATA, EPARS, PDATA, SDATA

Declarations

```
DIMENSION RELD(8), RJACD(8), RLUBD(8), REXD(8), TEXD(8), UACD(8)
EQUIVALENCE (RELD, PDATA(1,14)), (RJACD, PDATA(1,15)), (RLUBD, PDATA(1,16)),
            (REXD, PDATA(1,17)), (TEXD, PDATA(1,18)), (UACD, PDATA(1,37))
EQUIVALENCE (TSATUR, SDATA(1,2)), (RMXKWD, SDATA(1,22))
EQUIVALENCE (PND, OPCAP(2))
```

Input

Source of Data	Name	Description
CCBTEPS(PLOAD(2))	EELECD	Electrical energy output rate for diesel engine (Btu/hr)
/EPARS/(OPCAP(2))	PND	Nominal electrical power output (Btu/hr)
SDATA(1,2)	TSATUR	Steam saturation temperature (°F)
SDATA(1,22)	RMXKWD	(Maximum exhaust flow rate)/(kW output)
PDATA(1,14)	RELD	Quadratic polynomial coefficients related to electrical energy output
PDATA(1,15)	RJACD	Quadratic polynomial coefficients related to jacket heat
PDATA(1,16)	RLUBD	Quadratic polynomial coefficients related to lube oil heat
PDATA(1,17)	REXD	Quadratic polynomial coefficients related to exhaust heat energy
PDATA(1,18)	TEXD	Quadratic polynomial coefficients related to exhaust gas temperature
PDATA(1,37)	UACD	(Stack U-factor) * area coefficients for diesel engine
/EDATA/	RMIN(2)	Minimum part-load ratio for diesel engine

Output

Name	Description
EEXD	Available exhaust heat rate (Btu/hr)
EFUELD	Fuel energy input rate (Btu/hr)
EJACKD	Available jacket heat rate (Btu/hr)
ELUBED	Available lube oil heat (Btu/hr)

Calculation Procedure

1. Initialize output variables.

   EJACKD = ELUBED = EEXD = EFUELD = 0.

2. If no electrical energy is required, i.e., EELECD $\leq$ 0, skip the following calculations and return; otherwise,

3. Convert nominal electrical power output from Btu/hr to kW.

   PNDK = PND/3412

4. Determine part-load ratio, RL, as (electrical power output)/(nominal electrical power capacity), subject to a lower bound $\geq$ RMIN(2).

   RL = AMAX1 (EELECD/PND, RMIN(2))

5. Calculate ratio of (electrical power output)/(fuel input).

   RELEC = [C(1)+C(2)*RL+C(3)*RL*RL],

   where C = RELD; then calculate the fuel input.

   EFUELD = EELECD/RELEC

6. Calculate ratio of available jacket heat to fuel input.

   RJACK = [C(1)+C(2)*RL+C(3)*RL*RL],

   where C = RJACD; then calculate available jacket heat.

   EJACKD = EFUELD * RJACK

7. Calculate ratio of available lube oil heat to fuel input.

   RLUBE = [C(1)+C(2)*RL+C(3)*RL*RL],

   where C = RLUBD; and calculate available lube oil heat.

   ELUBED = EFUELD * RLUBE

8. Calculate ratio of thermal energy of exhaust gases to fuel input.

   REXH = [C(1)+C(2)*RL+C(3)*RL*RL],

   where C = REXD; then calculate thermal energy of exhaust gases.

   EEXDT = EFUELD * REXH

9. Calculate exhaust gas temperature, (°F).

   TEX = [C(1)+C(2)*RL+C(3)*RL*RL]

   where C = TEXD.

10. Calculate exhaust gas flow rate, FEXD (lb/hr), from the rate at which energy leaves the diesel engine by the exhaust gas stream.

    FEXD = EEXDT/(0.25*(TEX-32.)),

    where 0.25 is the specific heat, c_p, and (TEX-32.) is the enthalpy of the gas at the exhaust temperature.

11. Calculate the stack heat loss coefficient, UA (Btu/hr-°F), as a function of the nominal capacity of the engine and a set of two coefficients.

    UAD = [C(1)*OPCAP(2)**C(2)],

    where C = UACD.

    This is an empirical relationship developed from manufacturers' data.

12. Calculate the temperature, TSTACK(°F), of the exhaust gas leaving the stack as a function of the steam saturation temperature, the stack UA factor, and exhaust gas heat capacity rate, subject to an upper bound on flow rate.

    Maximum exhaust gas flow rate = (Maximum exhaust gas flow rate/kW output) * (nominal kW output),

    TSTACK = TSATUR + (TEX-TSATUR)/(exp(UAD/AMIN1(FEXD,RMXKWD*PNDK)*0.25)),

    where 0.25 is the specific heat of the exhaust gas. This equation expresses the effectiveness of the stack as a heat exchanger, with (UA/heat capacity rate) = the number of transfer units and the limiting stack temperature being the steam saturation (constant) temperature. This assumes that the stack is jacketed with a heat exchanger through which water at the saturation temperature is flowing. This waste heat is then usable directly from steam generation (see subroutine HEATREC).

13. Calculate available exhaust heat (always ≥ 0) from an energy balance on the stack.

    EEXD = AMAX1(FEXD*0.25*(TEX-TSTACK),0.).

    where 0.25 is the specific heat of the exhaust gas.

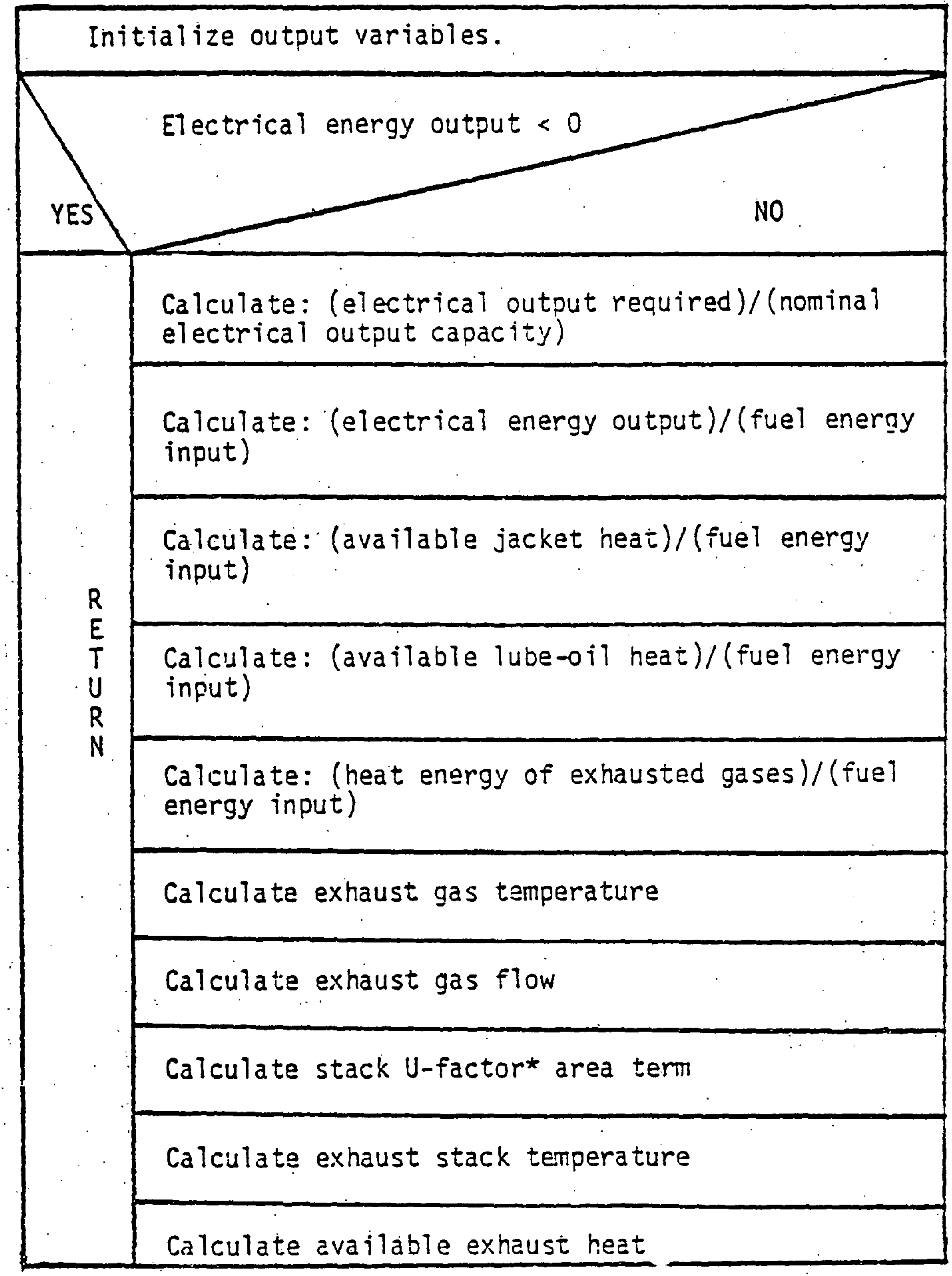
ASHRAE Verification

The part-load fuel consumption rate is expressed by ASHRAE (Ref. 2) as a linear function of part-load ratio. Cal-ERDA expresses the energy input/output relationship, rather then the fuel consumption directly, as a quadratic equation in the part-load ratio (defined with respect to nominal capacity). The approach is the same except that the Cal-ERDA expression allows for higher order variations, and this would be more accurate.

The recoverable heat rate, which applies to jacket, lube oil, and exhaust gas waste heat, is expressed by ASHRAE (Ref. 2) as a quadratic equation in the part-load ratio. The form of the Cal-ERDA expressions is identical. Note, however, that the thermal energy of the exhaust gases, calculated using a quadratic equation in the part-load ratio, is used only to calculate the exhaust gas flow rate. The available exhaust (waste) heat is then calculated from an energy balance on the stack and not directly, as in ASHRAE, as a quadratic equation in the part-load ratio.

Relationships for exhaust gas temperature, stack U-factor-area product and stack gas temperature are not presented by ASHRAE. The Cal-ERDA relationships for these variables were derived from manufacturers' data.

DIESEL Flow Chart


Initialize output variables.
Electrical energy output < 0
YES
NO
RETURN
Calculate: (electrical output required)/(nominal electrical output capacity)
Calculate: (electrical energy output)/(fuel energy input)
Calculate: (available jacket heat)/(fuel energy input)
Calculate: (available lube-oil heat)/(fuel energy input)
Calculate: (heat energy of exhausted gases)/(fuel energy input)
Calculate exhaust gas temperature
Calculate exhaust gas flow
Calculate stack U-factor* area term
Calculate exhaust stack temperature
Calculate available exhaust heat

i. Subroutine EFFIC[†]

Description

Subroutine EFFIC calculates efficiency factors for several classes of energy production. It also calculates total fuel energy input to the system.

Calculations are straightforward applications of definitions:

1/FE = plant electric energy generation efficiency where
FE = (fuel input energy to the prime movers)/(plant electric energy generation plus recovered heat)

1/FH = heat energy generation efficiency where
FH = (plant fuel input-plant electric generation*FE)/(total heat generation-steam energy used for electric generation)

1/FUE = Total electric energy generation efficiency where
FUE = (plant electric generation* FE + utility electricity/ TOTUEF)/(total electrical energy generation), and where
TOTUEF = efficiency of utility electric generation (a special parameter)

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

None

Common Blocks

EFFICD, EPARS, HOURTOT, SDATA, STMTUR

Declarations

```
EQUIVALENCE (RWSTUR, SDATA(1,36)), (TOTUEF, SDATA(1,37))
```

Input

Source of Data	Name	Description
SDATA(1,36)	RWSTUR	Exhaust steam/entering steam flow ratio
SDATA(1,37)	TOTUEF	Total efficiency of utility electric generation
/EPARS/	PLOAD(4)	Part load of boiler at current hour (Btu/hr)
/EFFICD/	EPLANT	Electric energy generated in the plant (Btu/hr)
/EFFICD/	PG	Ratio of gas turbine load to total electrical load
/EFFICD/	PD	Ratio of diesel engine load to total electrical load
/EFFICD/	PS	Ratio of steam turbine load to total electrical load

[†] This routine contains some errors that are being reviewed by the programmers. Corrections being made in the program will be reflected in later editions of this manual.

/EFFICD/	FE	Inverse of plant electrical energy generation efficiency
/EFFICD/	FH	Inverse of plant heat energy generation efficiency
/EFFICD/	FUE	Inverse of total electric energy generation efficiency
/EFFICD/	ESTMAB	Steam energy input to absorption chiller (Btu/hr)
/EFFICD/	EELECCO	Total electrical energy input to cooling stage (Btu/hr)
/EFFICD/	EHEAT	Heat energy (building load)(Btu/hr)
/EFFICD/	EWASTE	Waste heat energy from diesel engines and gas turbines (Btu/hr)
/EFFICD/	ERCVCD	Heat energy recovered from double-bundle chiller (Btu/hr)
/STMTUR/	FSTMTUR	Flow of steam entering steam turbine (Btu/hr)
/STMTUR/	HSTMTUR	Enthalpy of superheated high-pressure steam (Btu/lb)
/STMTUR/	TEXSTM	Temperature of exhaust turbine (°F)
/HOURTOT/	EUT	Total utility electricity (Btu/hr)
/HOURTOT/	EFUELB	Total boiler fuel energy (Btu/hr)
/HOURTOT/	EFUELE	Total fuel input energy to diesel and gas turbine for electric energy generation (Btu/hr)
/HOURTOT/	EFUEL	Total fuel input (Btu/hr)
/HOURTOT/	EBOILER	Boiler net heat output (Btu/hr)
/HOURTOT/	EELECT	Total electrical energy requirement (Btu/hr)
/HOURTOT/	EHEATT	Total heating load (Btu/hr)
/HOURTOT/	EELECCD	Total electric input to D-bundle chiller (Btu/hr)
/HOURTOT/	ERECOVR	Total recovered heat energy (Btu/hr)

Output

Name	Description
ENTOT	Total energy consumed by the plant and the utility company (Btu/hr)
EFIHC	Total fuel input for heat energy consumed by cooling stage (Btu/hr)
EFIEC	Total fuel input for electric energy consumed by cooling stage (Btu/hr)

EFUELITH Total fuel input for heat energy generation (Btu/hr)

EFUELIE Total fuel input for electrical energy generation (Btu/hr)

Calculation Procedure

1. Calculate total fuel input/hr = total fuel input for electrical generation + boiler fuel energy.

   EFUEL = EFLUELE + EFUELB

2. Calculate total plant electrical generation/hr = total electrical energy requirement of site (including plant) * (sum of the ratios of prime mover electric loads to total electrical load).

   EPLANT = EELECT * (PG+PD+PS),

   where PG, PD, and PS were calculated in OPELEC.

3. Calculate total electricity supplied by utility/hr = total electrical energy requirement of site (including plant) - total plant electrical generation/hr.

   EUT = EELECT - EPLANT

4. Calculate intermediate variable, boiler efficiency, BOILEF. BOILEF = 1 unless total fuel input to boiler > 0., when BOILEF = Boiler part-load ratio/total fuel input to boiler.

   BOILEF = 1.
   IF (EFUELB.GT.0) BOILEF = PLOAD(4)/EFUELB

5. Calculate waste heat of steam turbine = entering steam flow * (entering steam enthalpy - (exhaust steam temperature-32.) * (exhaust steam to entering steam flow).

   ESTMST = FSTMTUR * (HSTMTUR-(TEXSTM-32.) * RWSTUR)

6. Set heat recovered from steam turbines to waste heat of steam turbine, subject to an upper bound, waste heat of diesel engine and gas turbine.

   ERECST = AMIN1(ESTMST, EWASTE)

7. Calculate total heat recovered from electrical generation equipment (either directly or through storage) = total recovered heat, as calculated in subroutines HEATREC and ENSTOR) - recovered heat from double-bundle chiller, which could neither be used nor stored.

   ERECTOT = ERECOVR - ERCVCD

8. Initialize inverse of plant electrical generation efficiency.

   FE = 0:
   If EPLANT > 0., FE = (total fuel input energy for electrical generation) + (waste heat turbine - heat recovered from steam turbine)/(boiler efficiency)/(EPLANT + total heat recovered in plant - heat recovered from steam turbine).

   FE = (EFUELE + (ESTMST-ERECST)/ BOILEF)/(EPLANT + ERECTOT-ERECST)

9. Initialize inverse of total electric energy generation efficiency,

   FUE = 0.;

If EELECT > 0, FUE = [(total fuel input for electrical generation) - (total heat recovered in plant-heat recovered from steam turbine) * FE + (waste heat of steam turbine/boiler efficiency) + (total utility electricity/total efficiency of utility generation)]/total electrical energy requirement.

FUE = (EFUELE-(ERECTOT-ERECST) * FE + ESTMST/BOILEF + EUT/TOTEUF)/
EELECT

10. Initialize inverse of plant heat energy generation efficiency,

FH = 0.;

If total hourly heat energy >0, FH = [(Total boiler fuel energy) - steam turbine waste heat/boiler efficiency + (total plant heat recovered-heat recovered from steam turbine) *FE + (total electric input to double-bundle chiller*FUE)]/(Boiler net heat output-waste heat of steam turbine + total heat recovered in plant + recovered heat from D-bundle chiller).

FH = (EFUELB-ESTMST/BOILEF + (ERECTOT-ERECST)*FE
+ EELECCD * FUE)
/(EBOILER-ESTMST + ERECTOT + ERCVCD)

11. Total fuel input for heat energy consumed by cooling stage = steam energy input to absorption chiller * FH.

EFIHC = ESTMAB * FH

12. Total fuel input for electric energy consumed by cooling stage = total electric energy input to cooling stage * FUE.

EFIEC = EELECCO * FUE

13. Total fuel input for heat energy generation = total plant heat generation * FH.

EFUELIH = EHEATT * FH

14. Total fuel input for electric energy generation (plant and utility) = total electrical energy requirement * FUE.

EFUELIE = EELECT * FUE

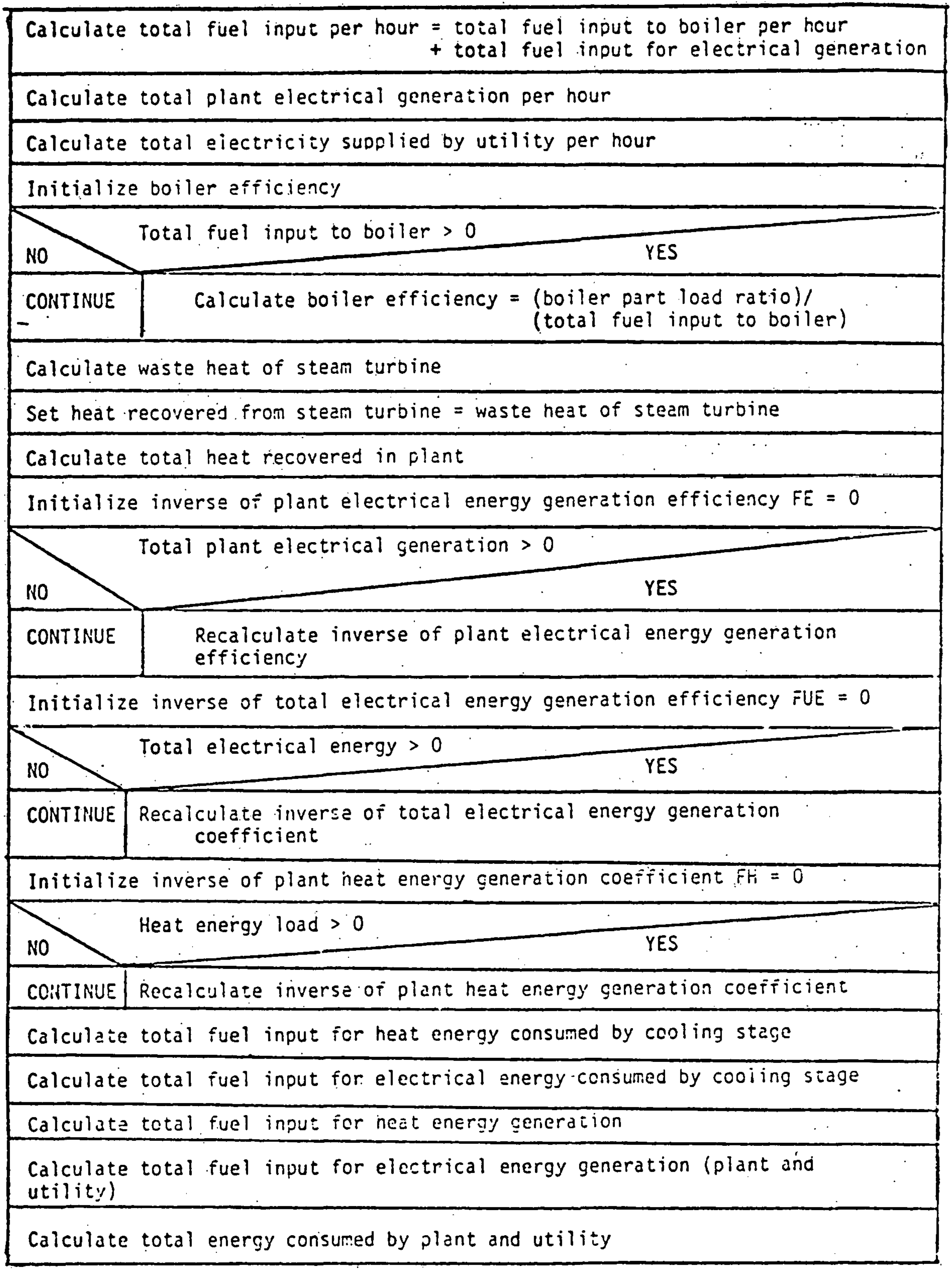
15. Total energy consumed by plant and utility = total fuel input + (total utility electricity)/(total efficiency of utility electrical generation).

ENTOT = EFUEL + EUT/TOTUEF

ASHRAE Verification

There is no applicable ASHRAE efficiency algorithm.

EFFIC Flow Chart



```
Calculate total fuel input per hour = total fuel input to boiler per hour
                                    + total fuel input for electrical generation
Calculate total plant electrical generation per hour
Calculate total electricity supplied by utility per hour
Initialize boiler efficiency
Total fuel input to boiler > 0
    NO:  CONTINUE
    YES: Calculate boiler efficiency = (boiler part load ratio)/
                                       (total fuel input to boiler)
Calculate waste heat of steam turbine
Set heat recovered from steam turbine = waste heat of steam turbine
Calculate total heat recovered in plant
Initialize inverse of plant electrical energy generation efficiency FE = 0
Total plant electrical generation > 0
    NO:  CONTINUE
    YES: Recalculate inverse of plant electrical energy generation
         efficiency
Initialize inverse of total electrical energy generation efficiency FUE = 0
Total electrical energy > 0
    NO:  CONTINUE
    YES: Recalculate inverse of total electrical energy generation
             coefficient
Initialize inverse of plant heat energy generation coefficient FH = 0
Heat energy load > 0
    NO:  CONTINUE
    YES: Recalculate inverse of plant heat energy generation coefficient
Calculate total fuel input for heat energy consumed by cooling stage
Calculate total fuel input for electrical energy consumed by cooling stage
Calculate total fuel input for heat energy generation
Calculate total fuel input for electrical energy generation (plant and
utility)
Calculate total energy consumed by plant and utility
```

j. Subroutine ENSTOR.

SUBROUTINE ENSTOR (OPMODE, EHEAT, ECOOLD, EHEATR, ECOOLR)

Description

This subroutine calculates heating and cooling energy storage. Heat energy is stored in hot-water tanks, while cooling energy is stored in chilled-water tanks. It is assumed that the speed of energy storage is sufficiently high so that any amount of energy up to the storage tank capacity can be stored in an hour.

At the beginning of each hourly simulation, it is called with OPMODE > 0. It first tries to store all waste heat as heat energy. If the capacity of the hot water tank is exceeded, the remainder of the waste heat is used to generate chilled water. The absorption chiller capacity assigned for this purpose is assumed to be the capacity remaining after absorption cooling has been serviced. The initialization of the subroutine is done at the beginning of the year by calling it with OPMODE = 0.

Temperatures are not explicitly calculated for either tank; a usable temperature range, dictated by the tank capacity, is assumed for each. The hot-water tank receives waste heat energy that cannot be used directly in the heat exchanger processes modeled by the heat recovery subroutine HEATREC. Thus, only the excess of water heat supply over demand at temperature level 5 is added to the tank. Any that exceeds the tank capacity is used to drive an absorption chiller. Similarly, cooling energy "added" to the chilled-water tank comes from absorption chillers.

Likewise, energy removed from each tank is assumed to be above or below the specific temperature level characteristics of the loads to which this energy is applied

Now no more than one hot and one cold storage tank are allowed.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

None

Common Blocks

EPARS, HOURTOT, LDISTD

Declarations

None

Input

Source of Data	Name	Description
CCBTEPS	ECOOLD	Cooling load (Btu/hr)
CCBTEPS	EHEAT	Heating load (Btu/hr)
CCBTEPS	OPMODE	Variable representing the operational mode; it is less than 0 when energy from storage is used, equal to 0 for initialization, and greater than 0 when energy is stored
/EPARS/	IABSOR	Absorption chiller type, same as in ABSREF
/EPARS/	TOTCAP(18)	Total heat storage capacity (Btu)
/EPARS/	TOTCAP(19)	Total cooling storage capacity (Btu)
/EPARS/	TOTCAP (IABSOR)	Total cooling capacity of absorption chiller type IABSOR (Btu/hr)
/LDISTD/	NOPR(18)	Number of heat storage tanks in operation
/LDISTD/	NOPR(19)	Number of cooling storage tanks in operation
/HOURTOT/	ESTORBL	Storable heat energy available from heat recovery (Btu/hr)
/HOURTOT/	EWASTED	Total wasted recoverable heat at end of hour (Btu/hr)
/HOURTOT/	ESTRED	Total energy stored (Btu/hr)

Output

Name	Description
EHSTOR	Stored heat energy at end of hour (Btu/hr)
ECSTOR	Stored cooling energy at end of hour (Btu/hr)
EHEATR	Remaining heating load after adjustment for stored energy (Btu/hr)
ECOOLR	Remaining cooling load after adjustment for stored energy (Btu/hr)

Calculation Procedure

1. Initialize energy storage variables.

   If OPMODE ≠ 0, skip to 2; otherwise,
   EHSTOR = ECSTOR = 0, and return.

2. Initialize remaining loads, i.e., those heating or cooling loads remaining after adjustment for storage.

   EHEATR = EHEAT
   ECOOLR = ECOOLD

3. Begin use of stored energy, i.e., removal of energy (heating or cooling) from storage.

   If OPMODE > 0, skip to 9; otherwise,

   set stored cooling energy used this hour, ECUSED, equal to the cooling load, subject to an upper bound = ECSTOR.

   ECUSED = AMIN1 (ECOOLD, ECSTOR),

   where ECSTOR = stored cooling energy at end of previous hour (Btu/hr).

4. Adjust cooling load for use of stored cooling energy to get remaining cooling load.

   ECOOLR = ECOOLD - ECUSED

5. Calculate stored cooling energy at end of hour.

   ECSTOR = ECSTOR - ECUSED
   = (stored cooling energy at beginning of hour) - (stored cooling energy used).

   Note that treatment of storage as an energy pool; no temperatures are calculated.

6. Repeat procedure for stored heating energy.

   EHUSED = AMIN1 (EHEAT, EHSTOR),

   where EHSTOR = stored heating energy at end of previous hour (Btu/hr),

   EHEATR = EHEAT - EHUSED, and
   EHSTOR = EHSTOR - EHUSED.

7. Initialize equipment operation indexes.

   IOPR(1,18) = IOPR(1,19) = 0.
   If EHUSED > 1, IOPR(1,18) = 1,
   If ECUSED > 1, IOPR(1,19) = 1,

   where IOPR(1,18) = equipment operation index (18 = heating energy storage)
   = 1 if heat either added or removed from storage during hour
   = 0, otherwise.

   IOPR(1,19) = equipment operation index (19 = cooling energy storage)
   = 1 if cooling energy either added or removed from storage during hour
   = 0, otherwise.

8. Return. Completes removal of energy from storage.

9. Begin storage of energy. Set heating energy stored this hour, EHSTRED, equal to the storable heat energy available from heat recovery (subroutine HEATREC), subject to an upper bound = (TOTCAP(18)-EHSTOR).

   EHSTRED = AMIN1 (TOTCAP(1,8)-EHSTOR, ESTORBL),

   where (TOTCAP(18)-EHSTOR) = remaining capacity of hot storage tank at beginning of hour
   and ESTORBL = high quality energy available from heat recovery, i.e., that overflowed from the highest heat recovery stage, ES(5), and could not be used directly at this hour.

Note the treatment of storage as an energy pool. Although no temperature is calculated, the energy stored here is $\geq$ 180 °F because of its origin in HEATREC.

10. Adjust heating energy stored.

    EHSTOR = EHSTOR + EHSTRED

11. Calculate recoverable waste heat that could not be stored because of insufficient capacity, EWREM.

    EWREM = ESTORBL - EHSTRED $\geq$ 0

12. Set ratio of heat energy in to cooling energy out, SA, for absorption chiller (to be driven by excess waste heat).

    If 1-stage chiller, SA = 1.5 (or COP = 0.67).
    If 2-stage chiller, SA = 0.9 (or COP = 1.11).

13. Initialize cooling energy stored.

    ECSTRED = 0.

14. If an absorption chiller is not available and/or the recoverable waste heat that could not be stored = 0, skip to 17; otherwise continue.

15. Waste heat that could not be stored is used to drive an absorption chiller. The cooling energy so produced is stored. Set cooling energy stored equal to the excess recoverable waste heat divided by the input/output energy ratio, subject to two sequential upper bounds.

    Maximum cooling energy produced = absorption chiller capacity remaining after absorption cooling has been serviced

    = OPCAP(IABSOR) - PLOAD(IABSOR),

    where PLOAD(IABSOR) = part-load of equipment IABSOR.
    Maximum cooling energy stored = (cooling storage capacity) - (cooling previously stored)
    = TOTCAP(19) - ECSTOR

    Thus ECSTRED = AMIN1 (TOTCAP(19)-ECSTOR, AMIN1 (EWREM/SA, OPCAP (IABSOR)-PLOAD(IABSOR))).

16. Update cooling energy stored.

    ECSTOR = ECSTOR + ECSTRED

17. Calculate the total heat energy that was recovered by storage.

    ESTRED = EHSTRED + ECSTRED * SA,
    where EHSTRED = heat stored directly
    ECSTRED * SA = heat to stored cooling energy

18. Update (accumulate) total heat recovered.

    ERECOVR = ERECOVR + ESTRED

19. Update (accumulate) total wasted recoverable heat, EWASTED, i.e., waste heat not used because of lack of sufficient demand and <u>not</u> sent to either hot or cold storage.

    EWASTED = EWASTED + (ESTORBL-ESTRED)

20. Update (accumulate) total heat energy used.

    EHEATT = EHEATT + ESTRED

21. Set equipment operation indexes to indicate whether energy is placed in or removed from storage.

    If EHSTRED > 1, IOPR(1,18) = 1.
    If ECSTRED > 1, IOPR(1,19) = 1.

22. Calculate present demand on heat storage tank.

    DEMAND(18) = ABS(EHEAT-EWASTED)

23. Calculate present part-load on heat storage tank.

    PLOAD(18) = ABS(EHUSED-EHSTRED)

24. Calculate present demand on cooling storage tank.

    DEMAND(19) = ABS(ECOOLD-EWREM/SA)

25. Calculate present part-load on cooling storage tank.

    PLOAD(19) = ABS(ECUSED-ECSTORED)

26. Calculate the number of different sizes of storage tanks in operation.

    Heat: NOPR(18) = IOPR(1,18)
    Cool: NOPR(19) = IOPR(1,19)

    At present NOPR can be only 0 or 1, i.e., only one size of each tank is possible.

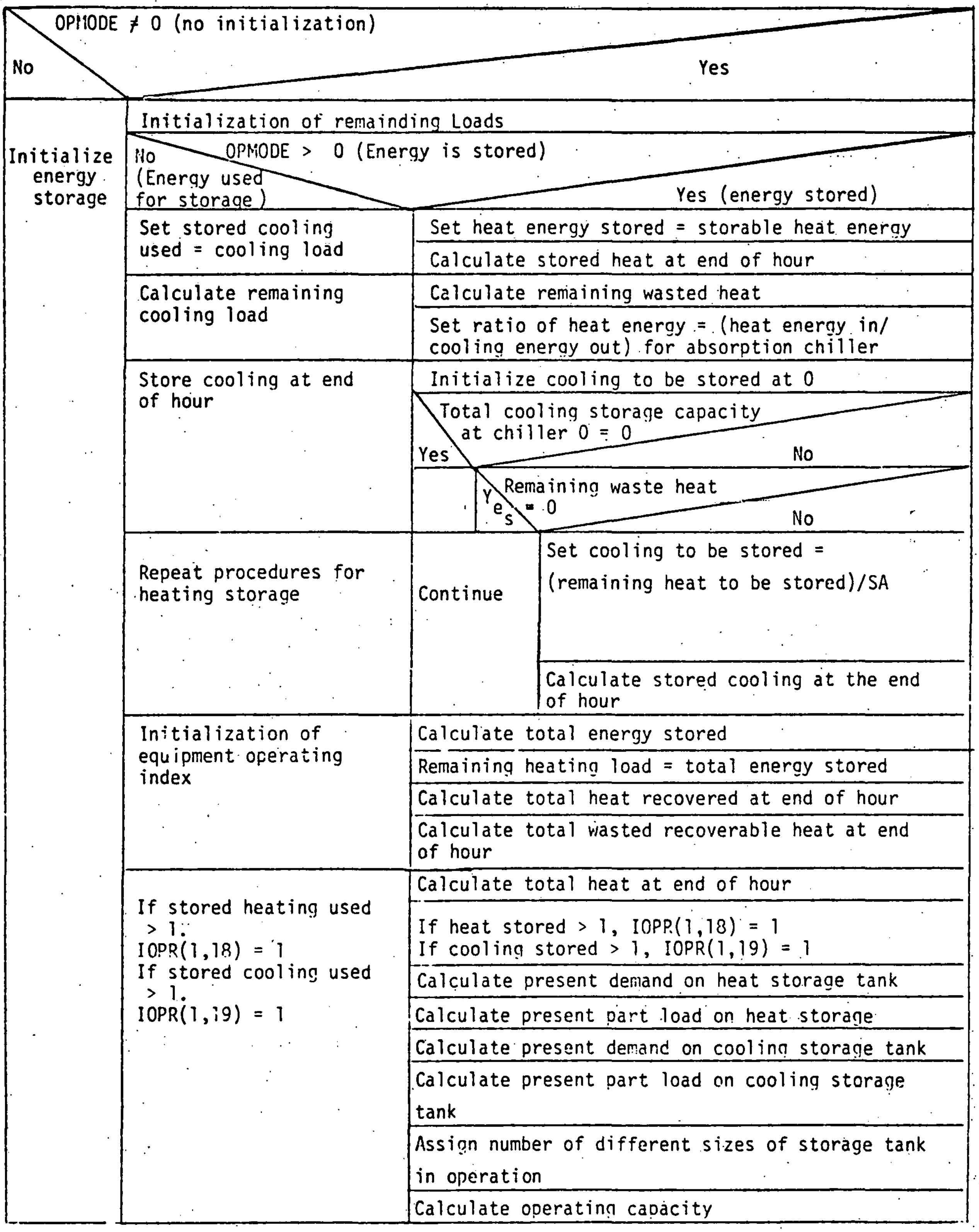
27. Calculate operating capacity of storage tanks.

    Heat: OPCAP(18) = NOPR(18) * TOTCAP(18)
    Cool: OPCAP(19) = NOPR(19) * TOTCAP(19)

ASHRAE Verification

There is no applicable ASHRAE storage model.

ENSTOR Flow Chart



```
OPMODE ≠ 0 (no initialization)
  No
    Initialize energy storage
  Yes
    Initialization of remainding Loads
    OPMODE > 0 (Energy is stored)
      No (Energy used for storage )
        Set stored cooling used = cooling load
        Calculate remaining cooling load
        Store cooling at end of hour
        Repeat procedures for heating storage
        Initialization of equipment operating index
        If stored heating used > 1.
        IOPR(1,18) = 1
        If stored cooling used > 1.
        IOPR(1,19) = 1
      Yes (energy stored)
        Set heat energy stored = storable heat energy
        Calculate stored heat at end of hour
        Calculate remaining wasted heat
        Set ratio of heat energy = (heat energy in/ cooling energy out) for absorption chiller
        Initialize cooling to be stored at 0
        Total cooling storage capacity at chiller 0 = 0
          Yes
            Continue
          No
            Remaining waste heat = 0
              Yes
              No
                Set cooling to be stored = (remaining heat to be stored)/SA
                Calculate stored cooling at the end of hour
        Calculate total energy stored
        Remaining heating load = total energy stored
        Calculate total heat recovered at end of hour
        Calculate total wasted recoverable heat at end of hour
        Calculate total heat at end of hour
        If heat stored > 1, IOPR(1,18) = 1
        If cooling stored > 1, IOPR(1,19) = 1
        Calculate present demand on heat storage tank
        Calculate present part load on heat storage
        Calculate present demand on cooling storage tank
        Calculate present part load on cooling storage tank
        Assign number of different sizes of storage tank in operation
        Calculate operating capacity
```

k. Function ENTHAL.

FUNCTION ENTHAL(PSTEAM, TSTEAM)

Description

ENTHAL is a function subprogram that calculates enthalpy of pure steam, in Btu/lb. Enthalpy is calculated as a quadratic polynomial of steam temperature in which the polynomial coefficients are related to the absolute steam pressure. This function applies for steam temperatures up to 1032°F (556°C) and steam pressures up to 1000 psig (6.9x10^6 Pa).

Subprograms Calling This Routine

SUBROUTINE DFLTASG

Subprograms Called by This Routine

None

Common Blocks

None

Declarations

None

Input

Source of Data	Name	Description
DFLTASG	PSTEAM	Steam pressure (psig)
DFLTASG	TSTEAM	Steam temperature (°F)

Output

Name	Description
ENTHAL	Enthalpy of pure steam (Btu/lb)

Calculation Procedure

1. Calculate absolute steam pressure.

   PSTMAB = PSTEAM + 14.7

   Note that sea-level atmospheric pressure is assumed.

2. Calculate enthalpy as a quadratic function of steam temperature.

   ENTHAL = A + B * TSTEAM + C * TSTEAM^2,

   where A = 1068. - 0.485 * PSTMAB,
   B = 0.432 + 0.000953 * PSTMAB, and
   C = 0.000036 - 0.000000496 * PSTMAB.

ASHRAE Verification

There is no applicable ASHRAE algorithm. The form of this function was taken directly from the NECAP program (Ref. 3, Vol. II, pp. 5-123).

ENTHAL Flow Chart

Calculate absolute steam pressure: PSTMAB = PSTEAM + 1.013 x 10^5 (Pa)
Determine quadratic coefficients A, B, and C using absolute steam pressure PSTMAB
Calculate enthalpy using steam temperature ENTHAL = A + B * TSTEAM + C * TSTEAM**2 (kJ/kg)

1. Function ENTROP.

FUNCTION ENTROP(PSTEAM, TSTEAM)

Description

ENTROP is a function subprogram used to calculate entropy of steam. Entropy is calculated in terms of a function involving saturation temperature and steam temperature, while the coefficients in the function are related to absolute steam pressure.

Subprograms Calling This Routine

SUBROUTINE DFTASG

Subprograms Called by This Routine

FUNCTION SATUR

Common Blocks

None

Declarations

None

Input

Source of Data	Name	Description
DFLTASG	PSTEAM	Steam pressure (psig)
DFLTASG	TSTEAM	Steam temperature (°F)

Output

Name	Description
ENTROP	Entropy of steam (Btu/lb)

Calculation Procedure

1. Call function SATUR to get saturation temperature as a function of steam pressure.

   TSATUR = SATUR(PSTEAM)

2. Entropy is calculated as a cubic function of the saturation temperature and steam temperature.

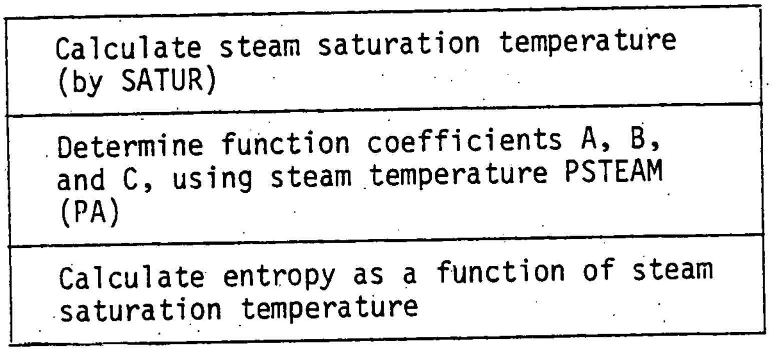
   $$\mathrm{ENTROP} = 2.385 - 0.004398 * \mathrm{TSATUR} + 0.000008146 * \mathrm{TSATUR}^2 - 0.626\times10^{-8} * \mathrm{TSATUR}^3 + 2. * C * (\mathrm{TSTEAM}-\mathrm{TSATUR}) + (B-920.*C) * \mathrm{ALOG}[(\mathrm{TSTEAM} + 460.)/(\mathrm{TSATUR} + 460.)]$$

   where A = 1068. - .485 * PSTEAM,
   B = 0.432 + 0.000953 * PSTEAM, and
   C = 0.000036 - 0.000000496 * PSTEAM.

ASHRAE Verification

There is no applicable ASHRAE algorithm. The form of this function was taken directly from the NECAP program.

ENTROP Flow Chart



Calculate steam saturation temperature (by SATUR)

Determine function coefficients A, B, and C, using steam temperature PSTEAM (PA)

Calculate entropy as a function of steam saturation temperature

m. Subroutine GASTUR

SUBROUTINE GASTUR (EELEC, EFUEL, ELUBE, EEX, TEX)

Description

This subroutine simulates a gas turbine generator set.

Parameters relating to output variables are evaluated as follows:

(1) Ratio of (gas turbine fuel energy input)/(electrical energy output)

$= f_1(x_1,c^1) * f_2(x_2,c^2)$,

where f_1 and f_2 = quadratic polynomials in x_1; c^1 and c^2 are sets of polynomial coefficients for the two polynomials, respectively;

x_1 = part-load ratio
= (electrical power output)/(nominal electrical power capacity);
c^1 = FUEL1G;
x_2 = ambient air temperature(°F); and
c^2 = FUEL2G.

(2) Ratio of (exhaust gas flow rate)/(nominal electrical power capacity
$= f_1(x_1,c^1)$,
where f_1 = as defined above,
x_1 = ambient air temperature (°F), and
c^1 = FEXG.

(3) Exhaust gas temperature (°F) = $f_1(x_1,c^1) * f_2(x_2,c^2)$,
where f_1 and f_2 = quadratic polynomials in x_1 and x_2; c^1 and c^2 are sets of polynomial coefficients for the two polynomials, respectively;
x_1 = part-load ratio, as above;
c^1 = TEX1G;
x_2 = ambient air temperature (°F); and
c^2 = TEX2G.

(4) Ratio of (available lube oil heat)/(nominal electrical power capacity)
$= f_1(x_1,c^1)$,
where f_1 = as defined above;
x_1 = part-load ratio, as above; and
c^1 = ELUBG.

The exhaust gas stack U-factor-area product and the stack gas temperature are calculated as discussed in the calculation procedure below.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

SUBROUTINE OPCOOL

Common Blocks

EDATA, EPARS, PDATA, SDATA, WEATHR

Declarations

```
DIMENSION FUEL1G(8), FUEL2G(8), FEXG(8), TEX1G(8), TEX2G(8), ELUBG(8),
          UACG(8)
EQUIVALENCE (FUEL1G, PDATA(1,19)), (FUEL2G, PDATA(1,20)), (FEXG, PDATA
            (1,22)),(TEX1G, PDATA(1,23)),(TEX2G, PDATA(1,24)),(ELUBG,
            PDATA(1,25)), (UACG, PDATA(1,38))
EQUIVALENCE (TSATUR, SDATA(1,2)), (RMXKWG, SDATA(1,24))
EQUIVALENCE (PNG, OPCAP(1))
```

Input

Source of Data	Name	Description
CCBTEPS(PLOAD(1))	EELEC	Electrical energy output rate for gas turbine (Btu/hr)
/WEATHR/	TAIR	Ambient air temperature (°F)
/EPARS/(OPCAP(1))	PNG	Nominal electrical power output (Btu/hr)
/EDATA/	RMIN(1)	Minimum part-load ratio for gas turbine-generator set
/SDATA/	TSATUR	Steam saturation temperature (°F)
/SDATA/	RMXKWG	(Maximum exhaust gas flow rate)/ (kW output)
PDATA(1,19)	FUEL1G	A set of quadratic polynomial coefficients as described in previous section
PDATA(1,20)	FUEL2G	A set of quadratic polynomial coefficients as described in previous section
PDATA(1,22)	FEXG	A set of quadratic polynomial coefficients as described in previous section
PDATA(1,23)	TEX1G	A set of quadratic polynomial coefficients as described in previous section
PDATA(1,24)	TEX2G	A set of quadratic polynomial coefficients as described in previous section
PDATA(1,25)	ELUBG	A set of quadratic polynomial coefficients as described in previous section
PDATA(1,28)	UACG	A set of quadratic polynmial coefficients for (stack U-factor) * area description

Output

Name	Description
EEX	Available exhaust heat rate (Btu/hr)
EFUEL	Fuel energy input rate (Btu/hr)
ELUBE	Available lube oil heat rate (Btu/hr)
TEX	Exhaust gas temperature (°F)

Calculation Procedure

1. Initialize output variables and exhaust gas flow rate.

```
EFUEL = ELUBE = EEX = FEX = 0.
TEX = 50.
```

2. If no electrical energy is required, i.e., EELEC $\leq$ 0, skip the following calculations and return; otherwise,

3. Convert nominal electrical power output from Btu/hr to kW.

   PNGK = PNG/3412

4. Set the part-load ratio RLOAD, to (electrical power output)/(nominal electrical power capacity), subject to a lower bound, RL $\geq$ RMIN(1).

   RLOAD = AMAX1 (EELEC/PNG, RMIN(1))

5. Calculate gas turbine fuel input rate using two sets of quadratic I/O coefficients.

   $$\text{EFUEL} = \text{EELEC} * [C_1(1)+C_1(2)*\text{RLOAD}+C_1(3)*\text{RLOAD}*\text{RLOAD}] * [C_2(1)+C_2(2)*\text{TAIR}+C_2(3)*\text{TAIR}*\text{TAIR}],$$

   where C_1 = FUEL1G and

   C_2 = FUEL2G.

6. Calculate exhaust gas flow rate (lb/hr).

   FEX = PNG * [C(1)+C(2)*TAIR+C(3)*TAIR*TAIR],

   where C = FEXG.

7. Calculate exhaust gas temperature (°F) using two sets of quadratic coefficients.

   $$\text{TEX} = [C_1(1)+C_1(2)*\text{RLOAD}+C_1(3)*\text{RLOAD}*\text{RLOAD}] * [C_2(1)+C_2(2)*\text{TAIR}+C_2(3)*\text{TAIR}*\text{TAIR}],$$

   where C_1 = TEX1G and
   C_2 = TEX2G.

8. Calculate stack heat loss coefficient, UA (Btu/hr-°F), as a function of the nominal capacity of the gas turbine and a set of two coefficients.

   UAG + [C(1)*OPCAP(1)**C(2)],

   where C = UACG.

   This is an empirical relationship developed from manufacturers' data.

9. Calculate the temperature of the exhaust gas leaving the stack, TSTACK (°F), as a function of the steam saturation temperature, the stack UA factor, and exhaust gas heat capacity rate, subject to an upper bound on flow rate.

   Maximum exhaust gas flow rate = (Maximum exhaust gas flow rate/kW output) * (nominal kW output)

   TSTACK = TSATUR + (TEX-TSATUR)/(exp(UAG/(AMIN1(FEX, RMXKWG*PNGK)*0.25)),

   where 0.25 is the specific heat of the exhaust gas.

   This equation expresses the effectiveness of the stack as a heat exchanger, with (UA/heat capacity rate) = the number of transfer units and the limiting stack temperature being the steam saturation (constant) temperature. This assumes that the stack is jacketed with a heat-exchanger through which water is flowing at the saturation temperature. This waste heat is then usable directly for steam generation (see subroutine HEATREC).

10. Calculate available exhaust heat (always ≥ 0) from an energy balance on the stack.

    EEX = AMAX1(FEX*0.25*(TEX-TSTACK),0.)

11. Calculate available lube oil heat.

    ELUBE = EELEC *[C(1)+C(2)*RLOAD+C(3)*RLOAD*RLOAD],

    where C = ELUBG.

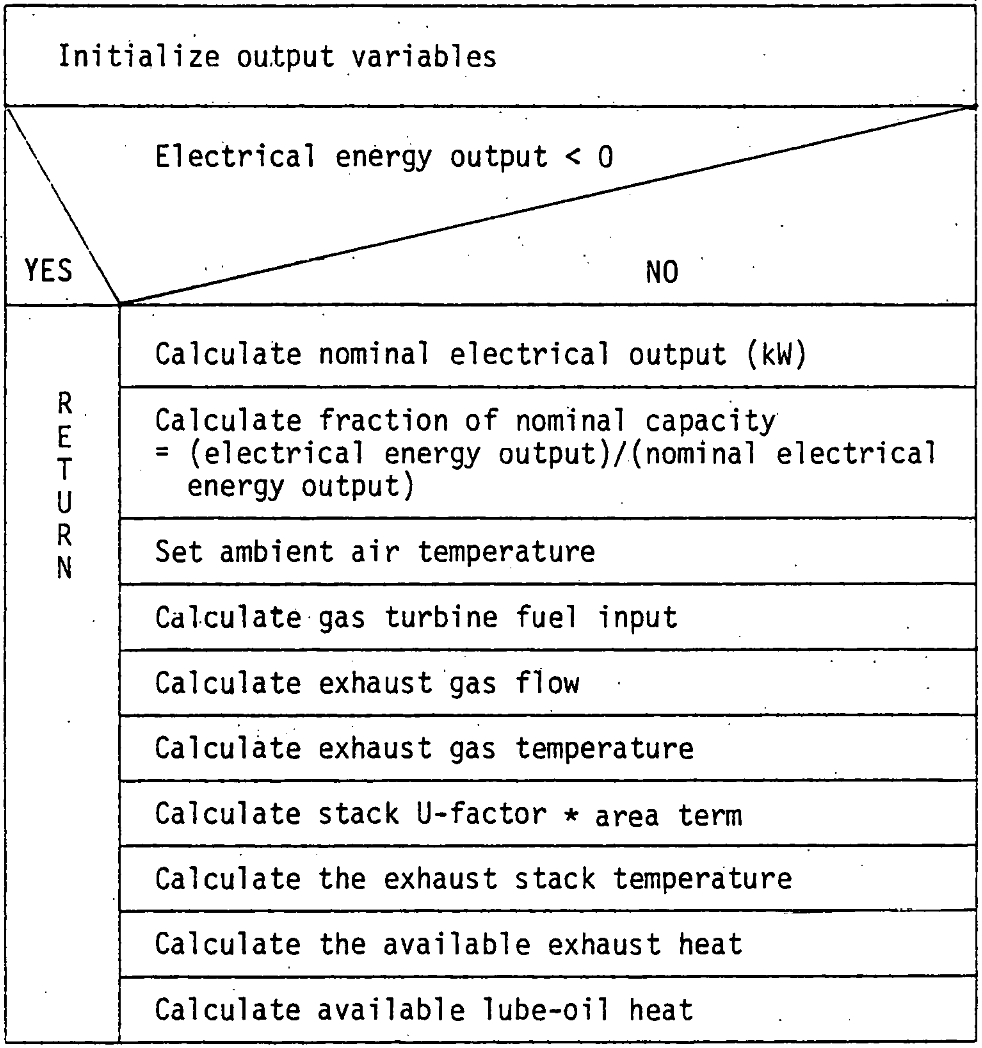
ASHRAE Verification

The part-load fuel consumption is expressed by ASHRAE (Ref. 2) as a linear function of part-load ratio. On the other hand, Cal-ERDA expresses the energy input/output relationship, rather than the fuel consumption directly, as a product of two quadratic equations: one in the part-load ratio (defined with respect to nominal capacity) and one in the ambient air temperature. Thus, the Cal-ERDA approach includes higher order variations in the part-load ratio expression and includes the dependence on inlet air (ambient) temperature.

The recoverable heat rate, which applies to lube oil and exhaust gas waste heat, is expressed by ASHRAE (Ref. 2) directly as a quadratic equation in the part-load ratio. The same expression is used in Cal-ERDA for the lube oil waste heat. Note, however, that in Cal-ERDA the exhaust gas flow rate and temperature are calculated using quadratic equations in the ambient air temperature and the part-load ratio and ambient air temperature, respectively. The available exhaust (waste) heat is then calculated from an energy balance on the stack and not directly, as in ASHRAE, as a quadratic equation in the part-load ratio.

Relationships for exhaust gas temperature, stack U-factor-area product, and stack gas temperature are not presented by ASHRAE. The Cal-ERDA relationships for these variables were derived from manufacturers' data.

GASTUR Flow Chart



Initialize output variables

Electrical energy output < 0

YES

NO

RETURN

Calculate nominal electrical output (kW)

Calculate fraction of nominal capacity
= (electrical energy output)/(nominal electrical energy output)

Set ambient air temperature

Calculate gas turbine fuel input

Calculate exhaust gas flow

Calculate exhaust gas temperature

Calculate stack U-factor * area term

Calculate the exhaust stack temperature

Calculate the available exhaust heat

Calculate available lube-oil heat

h. Subroutine HEATREC.

SUBROUTINE HEATREC (EHWDOM)

Description

This subroutine applies available waste heat from plant equipment to meet heat demands at five temperature levels and calculates total boiler demand. The temperature levels are:

Excess supply to hot water storage tank

	Demand ED(I)	Supply ES(I)	
Steam boiler heat + 2-stage absorption chiller heat	5	5	Waste heat from diesel and gas turbine exhaust
Single-stage absorption chiller heat	4	4	Diesel jacket heat + solar energy used for cooling
Space heating load	3	3	Recovered heat from double-bundle chiller
BSHW heating load	2	2	Lube oil rejected heat (diesel and gas turbine) + recovered heat from steam flash heating
Boiler feed water makeup	1	1	Steam turbine exhaust waste heat

Unrecoverable rejected heat to cooling tower

Thus, a series of heat exchangers is simulated. If the supply exceeds demand at a given level, the demand at the temperature stratum is satisfied and the recovered waste heat is accumulated. If the demand exceeds the supply, the excess demand is "pushed up" to the next highest temperature stratum, where it may be supplied. That portion of the demand that is met is accumulated as recovered waste heat. The excess of supply over demand at the highest level is sent to the hot-water storage tank.

The portion of demand that cannot be met at any level is passed on to the boiler. That portion of the supply that cannot be used at the lowest level is sent to the cooling tower as unrecoverable, rejected waste heat. This heat is rejected at the next hour.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

None

Common Blocks

EFFICD, EPARS, HOURTOT, SDATA, SOLARD, STM, STMTUR

Declarations

```
EQUIVALENCE (HSTEAM, SDATA(1,1)), (RFLASH, SDATA(1,3)), (TSATUR, SDATA
             (1,2)), (TWMAKE, SDATA(1,9)), (RHFLASH, SDATA(1,30)), (PEXSTUR,
             SDATA(1,34)), (RPMNOM, SDATA(1,35)), (RWSTUR, SDATA(1,36))
DIMENSION ED(5), ES(5)
```

Input

Source of Data	Name	Description
CCBTEPS ABS(ENGYLD(6,IHR,IDAY))	EHWDOM	Energy required to heat domestic hot water (Btu/hr)
/EPARS/	IABSOR	Absorption chiller type, same as in ABSREF
/EPARS/	OPCAP(17)	Steam turbine operating capacity (Btu/hr)
SDATA(1,1)	HSTEAM	Steam enthalpy (Btu/hr)
SDATA(1,2)	TSATUR	Steam saturation temperature (°F)
SDATA(1,3)	RFLASH	Ratio of boiler flash water to steam losses
SDATA(1,9)	TWMAKE	Temperature of make-up water (°F)
SDATA(1,30)	RHFLASH	Ratio of recoverable heat to steam flash heat
SDATA(1,34)	PEXSTUR	Nominal exhaust steam pressure (psig)
SDATA(1,35)	RPMNOM	Nominal speed of gas turbine (rpm)
SDATA(1,36)	RWSTUR	Ratio of exhaust steam to steam turbine entering steam flow
/EFFICD/	EHEAT	Heat energy (building) load (Btu/hr)
/EFFICD/	EWASTE	Waste heat energy from diesel engines and gas turbines (Btu/hr)
/EFFICD/	ERCVCD	Heat energy recovered from double-bundle chiller (Btu/hr)
/EFFICD/	ESTMAB	Steam energy input to absorption chiller (Btu/hr)
/EFFICD/	EJACKD	Jacket heat energy from diesel engines and gas turbines (Btu/hr)
/EFFICD/	ELUBE	Lube-oil heat from diesel engines and gas turbines (Btu/hr)
/SOLARD/	ESOLC	Solar heat energy used for cooling (Btu/hr)
/SOLARD/	ESOLH	Solar heat energy used for heating (Btu/hr)
/STM/	ESTMS	Total steam energy supplied to steam users except absorption chillers (Btu/hr)
/STM/	EWTRM	Energy of return condensate water from steam users (Btu/hr)
/STM/	FWTRM	Flow rate of return condensate water (Btu/hr)

/STMTUR/	HEXSTM	Enthalpy of exhaust steam from steam turbine (Btu/hr)
/STMTUR/	TEXSTM	Temperature of exhaust steam from steam turbine (°F)

Output

Name	Description
ERECOVR	Total recovered heat energy (Btu/hr)
ESTORBL	Total storable heat (Btu/hr)
EBOILER	Boiler net output (Btu/hr)

Calculation Procedure

1. Calculate total steam loss (lb/hr) from mass balance on steam system.

   FSLOSS = FSTMS - FWTRM,

   where FSTMS = total steam flow rate and
   FWTRM = total return condensate flow rate.

2. Calculate boiler flash water flow rate (lb/hr).

   FWFLASH = FSLOSS * RFLASH,

   where RFLASH = ratio of boiler flash water to steam losses
   = make-up water concentration to boiler water concentration.

3. Calculate flow rate (lb/hr) of boiler make-up water to feed water heater.

   FWMAKE = FSLOSS + FWFLASH

4. Calculate total steam energy requirements (Btu/hr), which is the energy demand at the highest temperature level (ED(5)), from an energy balance on the boiler.

   ED(5) = ESTMS - EWTRM + FWMAKE *(TSAUR-180.),

   where ESTMS = steam supply energy (excluding absorption chillers) and
   EWTRM = return condensate energy.

   It is assumed that the make-up water is supplied to the boiler at 180°F.

5. If the absorption chiller operating is two-stage, add its steam energy requirements to demand level 5.

   If IABSOR > 5, ED(5) = ED(5) + ESTMAB

6. If the absorption chiller operating is single-stage, assign its steam energy requirements to demand level 4.

   ED(4) = 0.
   If IABSOR = 5, ED(4) = ESTMAB

7. Assign the space heating load to demand level 3, the building service hot water (domestic + process) load to level 2, and the boiler feed-water make-up demand to level 1 ($\leq$ 180°F).

   ED(3) = EHEAT
   ED(2) = EHWDOM
   ED(1) = FWMAKE * (180.-TWMAKE)

   This assumes that the boiler feed-water heater exit temperature is 180°F.

8. Initialize the rejected low temperature heat, i.e., waste heat that cannot be recovered at temperature level 1 (≤ 180°F) because of insufficient demand at this level.

   ```
   EREJ = 0.
   ```
   EREJ is assumed to be of too low a temperature to be stored.

9. Assign the waste heat from diesel and gas turbine exhaust to the energy supply at the highest temperature level.

   ```
   ES(5) = EWASTED,
   ```
   where EWASTED = EEXD + EEXG (assigned in CCBTEPS).

10. Assign the diesel jacket heat + solar energy used for cooling to supply level 4, the recovered heat from the double-bundle chiller to level 3, and the lube oil heat + recoverable boiler flash heat to level 2.

    ```
    ES(4) = EJACKD + ESOLC
    ES(3) = ERCVCD
    ES(2) = ELUBE + FWFLASH * (TSATUR-180.) * RHFLASH
    ES(1) = 0.
    ```
    Supply at level 1 is initialized to 0.

    Note that the solar energy used for cooling, ESOLC, is computed in SOCOOL, but the actual application to the load occurs here.

11. If no steam turbine is present, OPCAP(17) ≤ 0, skip to 13; otherwise assign steam turbine exhaust waste heat to supply level 1 if the steam exhaust temperature ≥ 180°F.

    ```
    If TEXSTM ≥ 180,
    ES(1) = RWSTUR * FSTMTUR * (HEXSTM-TEXSTM+32.),
    ```
    where FSTMTUR = steam turbine entering steam flow rate and - TEXSTM + 32 = -(TEXSTM-32) = - enthalpy of saturated liquid at the exhaust temperature.

12. If the steam exhaust temperature < 180°F, add steam turbine exhaust waste heat to rejected low temperature heat.

    ```
    If TEXSTM < 180,
    EREJ = EREJ + RWSTUR * FSTMTUR * (HEXSTM-TEXSTM+32.)
    ```

13. Initialize recovered waste heat.

    ```
    ERECOVR = 0.
    ```

14. Compare recovered heat supplies (ES(I)) to demands (ED(I)) to try and satisfy loads. Steps 14 through 18 are repeated for I = 1,4.

    If the recovered heat supply at level I < the demand at level I, add the net of demand over supply to the next higher level and assign the supply to the net demand.

    ```
    If ES(1) < (ED(I), ED(I+1) = ED(I+1) + ES(I) - ED(I)
      ED(I) = ES(I)
    ```
    Otherwise skip to Step 16.

15. Update (accumulate) recovered waste heat.

    ```
    ERECOVR = ERECOVR + ED(I)
    ```
    Skip to step 19.

16. If the recovered heat supply at level I $\geq$ the demand at level I, update (accumulate) recovered waste heat.

    ERECOVR = ERECOVR + ED(I)

17. Update (accumulate) the rejected low temperature heat,

    EREJ = EREJ + ES(I) - ED(I)

18. Calculate wasted recoverable heat (that that can be neither recovered directly this hour nor stored).

    EWASTED = EREJ + EWASTCD,

    where EWASTCD = the waste heat from the double-bundle chiller that could not be used directly. EWASTCD is assumed nonstorable.

19. Add the recovered waste heat at level 5 to ERECOVR.

    ERECOVR = ERECOVR + AMIN1(ES(5), ED(5))

20. Calculate storable waste heat, always $\geq$ 0, as the excess of supply over demand at level 5.

    ESTORBL = AMAX1(0., ES(5)-ED(5))

21. Calculate boiler load (net output), always $\geq$ 0, as the excess of demand over supply at level 5.

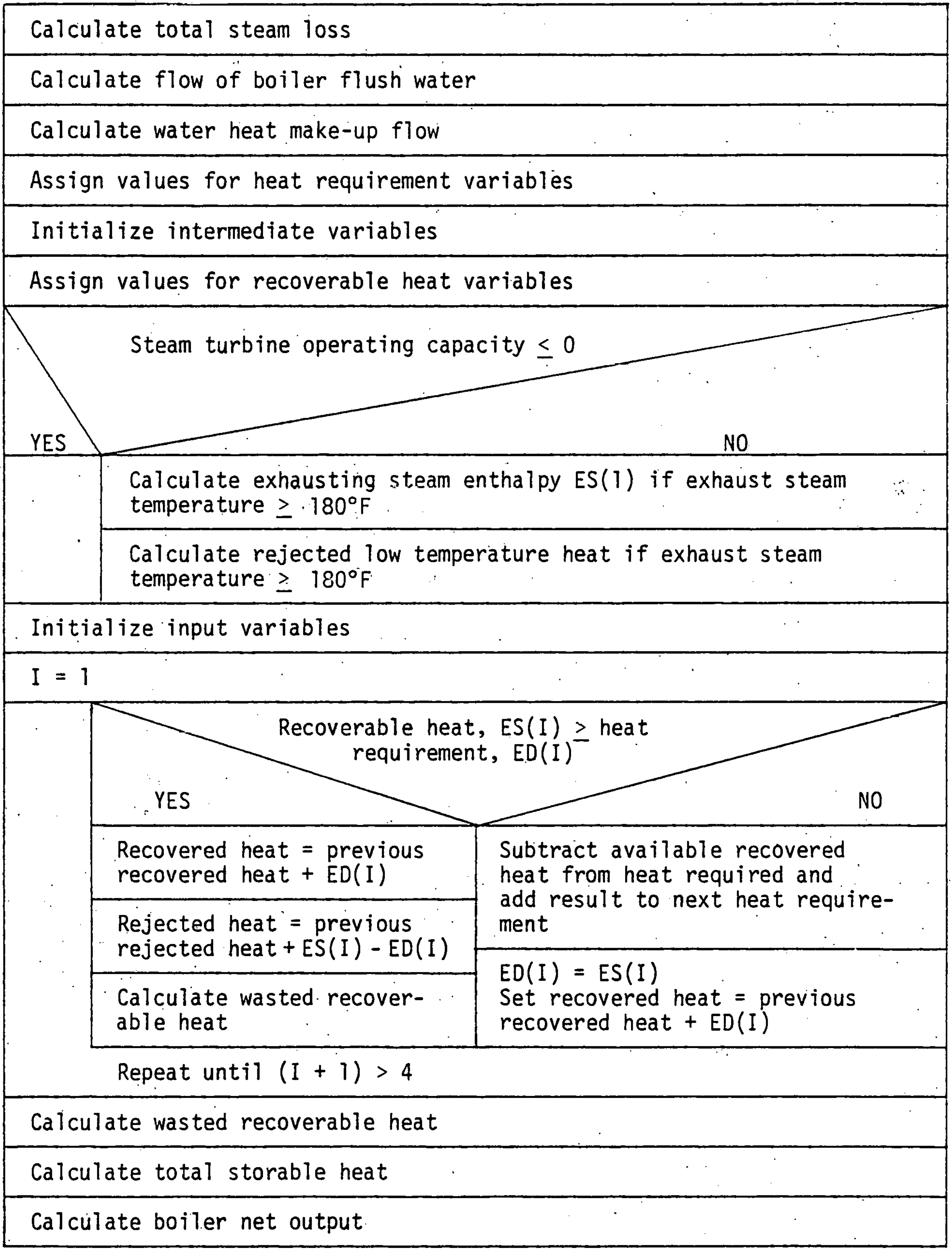
    EBOILER = AMAX1(0, ED(5)-ES(5))

    Thus the portion of demand that cannot be met at any level is passed on to the boiler.

ASHRAE Verification

There is no applicable ASHRAE heat recovery model.

HEATREC Flow Chart


Calculate total steam loss
Calculate flow of boiler flush water
Calculate water heat make-up flow
Assign values for heat requirement variables
Initialize intermediate variables
Assign values for recoverable heat variables
Steam turbine operating capacity ≤ 0
YES
NO
Calculate exhausting steam enthalpy ES(1) if exhaust steam temperature ≥ 180°F
Calculate rejected low temperature heat if exhaust steam temperature ≥ 180°F
Initialize input variables
I = 1
Recoverable heat, ES(I) ≥ heat requirement, ED(I)
YES
NO
Recovered heat = previous recovered heat + ED(I)
Rejected heat = previous rejected heat + ES(I) - ED(I)
Calculate wasted recoverable heat
Subtract available recovered heat from heat required and add result to next heat requirement
ED(I) = ES(I)
Set recovered heat = previous recovered heat + ED(I)
Repeat until (I + 1) > 4
Calculate wasted recoverable heat
Calculate total storable heat
Calculate boiler net output

o. Subroutine KARDRD.

Description

KARDRD reads a data card and echo prints the data.

Subprograms Calling This Routine

SUBROUTINE DESVAR

Subprograms Called by This Routine

None

Common Blocks

CDPR, CARD

Declarations

None

Input

Source of Data	Name	Description
/CDPR/	KARD	Logical unit number of input device
/CDPR/	ICDS	Logical unit number of echo print output device

Output

Name	Description
KEY1	If = *, comment card; continue to next card; otherwise read this card
ISEQ	Sequence number
NAM	Alphabetic designation of card
IN(I)	Input data, I=1,7
ICONT	Indicates whether this data set continues on the next card (blank = end of set)

Calculation Procedure

None

ASHRAE Verification

Inapplicable

p. Subroutine LDIST.

Description

LDIST is a default subroutine to distribute load to similar equipment units. It is called only if assignment tables are not provided by the user. It is assumed that:

(1) Units have identical efficiency functions (in terms of part-load ratio),

(2) Efficiency functions can be approximated by quadratic functions, and

(3) The rule should be simple and practical enough to be implemented without a computerized control.

This is a combination algorithm involving an iterative procedure. However, the number of iterations is at most the number of units. Allocation is done so that the part-load ratio of the operating units is close to the optimum part-load ratio.

Subprograms Calling This Routine

SUBROUTINE OPCOOL

Subprograms Called by This Routine

None

Common Blocks

DIST, EDATA, EPARS, LDISTD

Declarations

DIMENSION KOP(20)

Input

Source of Data	Name	Description
OPCOOL	TLOAD	Total load (Btu/hr)
OPCOOL	IEQTYPE	Index denoting the type of equipment
/EDATA/	RMAX(J)	Maximum part-load ratio of equipment type J
/EDATA/	NEQSIZE(J)	Number of sizes of type J
/EDATA/	CNOM(I,J)	Nominal capacity of type J and size I
/EDATA/	KAV(I,J)	Number of available units of type J and size I
/EDATA/	ROPT(J)	Optimum part-load ratio of equipment type J
/EPARS/	TOTCAP(J)	Total nominal capacity of equipment type J
/DISTB/	NDISTB(K)	Number of load ranges for load distribution by table
/DISTB/	DISTB(I,J)	Load range for equipment type J, load range index I
/DISTB/	IDISTB(I,K,J)	Number of units in use of size index I, load range index K, and equipment type J

Output

Name	Description
NOPR(J)	Number of sizes of equipment operating at current time step for equipment type J

IOPR(J) Number of units operating at current time step for equipment type J with size I

PLOADE Load of the equipment (Btu/hr)

Calculation Procedure

Note that Subroutine LDIST is called from within a DO loop that loops on equipment type (IEQTYPE).

1. Optimum part-load ratio, ROPTM = ROPT (IEQTYPE).
   Number of equipment sizes, NEQ = NEQSIZE (IEQTYPE).

2. If total capacity (IEQTYPE) = zero, or if total load $\leq$ 0.,

   (a) OPCAP(IEQTYPE) = PLOADE = 0
   KOP(1) = NEQ = 0

   (b) Skip to Step 24.

3. If NDISTB(IEQTYPE) $\neq$ 0, Skip to Step 19.

4. Capacity operating, CAPOP = TOTCAP(IEQTYPE).

5. For all equipment sizes, set number of I-size units operating to number of I-size units available.

   DO I = 1,NEQ
   KOP(I) = KAV(I,IEQTYPE)

6. Initialize looping index, II = 1.

7. Set RNEW equal to the total load/total capacity operating, subject to an upper bound, RMAX, maximum load ratio.

   RNEW = AMIN1(RMAX(IEQTYPE), TLOAD/CAPOP)

8. Set DNEW equal to absolute value of the difference between optimum part-load ratio and RNEW.

   DNEW = ABS(ROPTM-RNEW)

9. Set ROLD = RNEW
   DOLD = DNEW

10. If KOP(II) $\leq$ 0, go to Step 15.

11. CAPOP = CAPOP - nominal capacity (II,IEQTYPE).

12. If CAPOP < TLOAD, go to Step 16.

13. Decrease number of units operating by one.

    KOP(II) = KOP(II)-1
    Reset RNEW and DNEW as in Steps 7 and 8
    RNEW = AMIN1(RMAX(IEQTYPE, TLOAD/CAPOP)
    DNEW = ABS(ROPTM-RNEW)

14. If DNEW $\leq$ DOLD, go to Step 9;
    otherwise, increase number of units operating by one,

    KOP(II) = KOP(II) + 1,

    and go to Step 16.

15. If II > NEQ, go directly to Step 16.
Otherwise, increment II and return to Step 9.

16. Set capacity operating, CAPOP = 0.

17. Loop through different equipment sizes.

```
DO J = 1,NEQ
CAPOP = CAPOP + number of units operating (J) * nominal capacity (J,IEQTYPE)
```

18. OPCAP (IEQTYPE) = Capacity operating, CAPOP.
PLOADE = ROLD * Capacity operating.
Go to Step 24.

19. Equipment selection is made by table look-up.

```
I2 = NDISTB (IEQTYPE)
```

20. Loop through load ranges.

```
DO I = 1,12
If TLOAD < DISTB(I,IEQTYPE), jump out of loop to Step 21.
If loop runs to completion, set I = 12, insufficient equipment to handle
load.
```

21. Set number of equipment sizes, NEQ = NEQSIZE (IEQTYPE).

Set CAPOP = 0.

22. Loop through equipment sizes.

```
DO J = 1,NEQ
Set CAPOP = CAPOP + number of units operating (J) * nominal capacity
(J,IEQTYPE)
```

23. OPCAP (IEQTYPE) = CAPOP.

PLOADE = capacity operating, subject to an upper bound of total load present.

PLOADE = AMIN1 (TLOAD, CAPOP)

24. Number of units operating, NOPR (IEQTYPE) = NEQ.

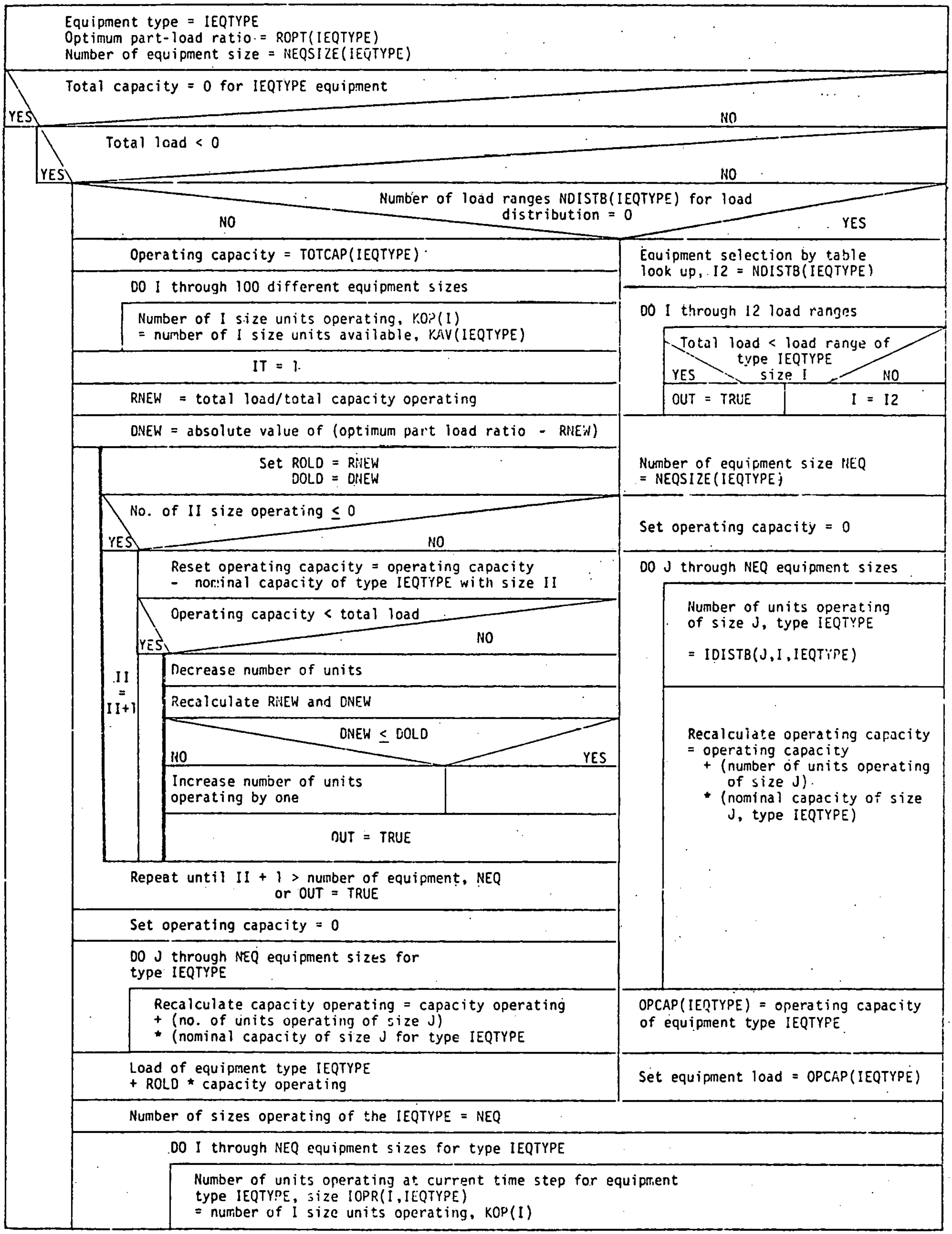
25. Loop through equipment sizes.

```
DO I = 1,NEQ
Operation index, IOPR (I,IEQTYPE) = KOP(I)
```

<u>ASHRAE Verification</u>

There is no applicable ASHRAE algorithm to subroutine LDIST.

LDIST Flow Chart



```
Equipment type = IEQTYPE
Optimum part-load ratio = ROPT(IEQTYPE)
Number of equipment size = NEQSIZE(IEQTYPE)
Total capacity = 0 for IEQTYPE equipment
YES
NO
    Total load < 0
    YES
    NO
        Number of load ranges NDISTB(IEQTYPE) for load distribution = 0
        NO
            Operating capacity = TOTCAP(IEQTYPE)
            DO I through 100 different equipment sizes
                Number of I size units operating, KOP(I)
                = number of I size units available, KAV(IEQTYPE)
            IT = 1
            RNEW = total load/total capacity operating
            DNEW = absolute value of (optimum part load ratio - RNEW)
                Set ROLD = RNEW
                    DOLD = DNEW
                No. of II size operating ≤ 0
                YES
                NO
                    Reset operating capacity = operating capacity
                    - nominal capacity of type IEQTYPE with size II
                    Operating capacity < total load
                    YES
                        Decrease number of units
                        Recalculate RNEW and DNEW
                        DNEW ≤ DOLD
                        NO
                            Increase number of units
                            operating by one
                        YES
                        OUT = TRUE
                    NO
                II = II+1
            Repeat until II + 1 > number of equipment, NEQ
                or OUT = TRUE
            Set operating capacity = 0
            DO J through NEQ equipment sizes for
            type IEQTYPE
                Recalculate capacity operating = capacity operating
                + (no. of units operating of size J)
                * (nominal capacity of size J for type IEQTYPE
            Load of equipment type IEQTYPE
            + ROLD * capacity operating
        YES
            Equipment selection by table
            look up, I2 = NDISTB(IEQTYPE)
            DO I through I2 load ranges
                Total load < load range of type IEQTYPE size I
                YES
                    OUT = TRUE
                NO
                    I = I2
            Number of equipment size NEQ
            = NEQSIZE(IEQTYPE)
            Set operating capacity = 0
            DO J through NEQ equipment sizes
                Number of units operating
                of size J, type IEQTYPE
                = IDISTB(J,I,IEQTYPE)
                Recalculate operating capacity
                = operating capacity
                  + (number of units operating
                    of size J)
                  * (nominal capacity of size
                    J, type IEQTYPE)
            OPCAP(IEQTYPE) = operating capacity
            of equipment type IEQTYPE
            Set equipment load = OPCAP(IEQTYPE)
        Number of sizes operating of the IEQTYPE = NEQ
        DO I through NEQ equipment sizes for type IEQTYPE
            Number of units operating at current time step for equipment
            type IEQTYPE, size IOPR(I,IEQTYPE)
            = number of I size units operating, KOP(I)
```

q. Subroutine LSEARC.

SUBROUTINE LSEARC(KEY, KEYLST, NKEYL, JD, IX)

Description

LSEARC is used to do a linear search in KEYLST.

Subprograms Calling This Routine

SUBROUTINE DESVAR

Subprograms Called by This Routine

None

Common Blocks

CDPR

Declarations

DIMENSION KEYLST(1)

Input

Source of Data	Name	Description
DESVAR	KEY	Variable for which match is sought in table
DESVAR	KEYLST	KYLST (JD, NKEYL) is the table to be searched
DESVAR	NKEYL	Index into table
DESVAR	JD	Index into table

Output

Name	Description
IX	Match index into table; if zero, no match was found

Calculation Procedure

None

ASHRAE Verification

Inapplicable

r. Subroutine OPCOOL.

SUBROUTINE OPCOOL (EELECT, EHEATT, ECOOLT, PA, PC, PCD)

Description

This subroutine distributes the cooling load (remaining after stored cooling energy is applied) to different generic chiller types near optimally. Some approximations are used to avoid an iterative calculation.

The methodology is as follows. If the available high-quality waste heat exceeds the load (remaining after stored heat energy is applied), this excess heating load is met by waste heat from compression chillers that are driven to meet the remaining cooling load ("cheap" cooling). The remaining cooling is provided by a combination of absorption and compression chillers so that steam generated for absorption chillers is supplied by waste heat from electrical generation equipment. This methodolgy is illustrated in the OPCOOL Functional Chart.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

SUBROUTINE DIESEL
SUBROUTINE GASTUR
SUBROUTINE LDIST
SUBROUTINE OPDBUN
SUBROUTINE OPELEC
SUBROUTINE SOCOOL
SUBROUTINE STMTUR

Common Blocks

EPARS, SDATA, STMTUR

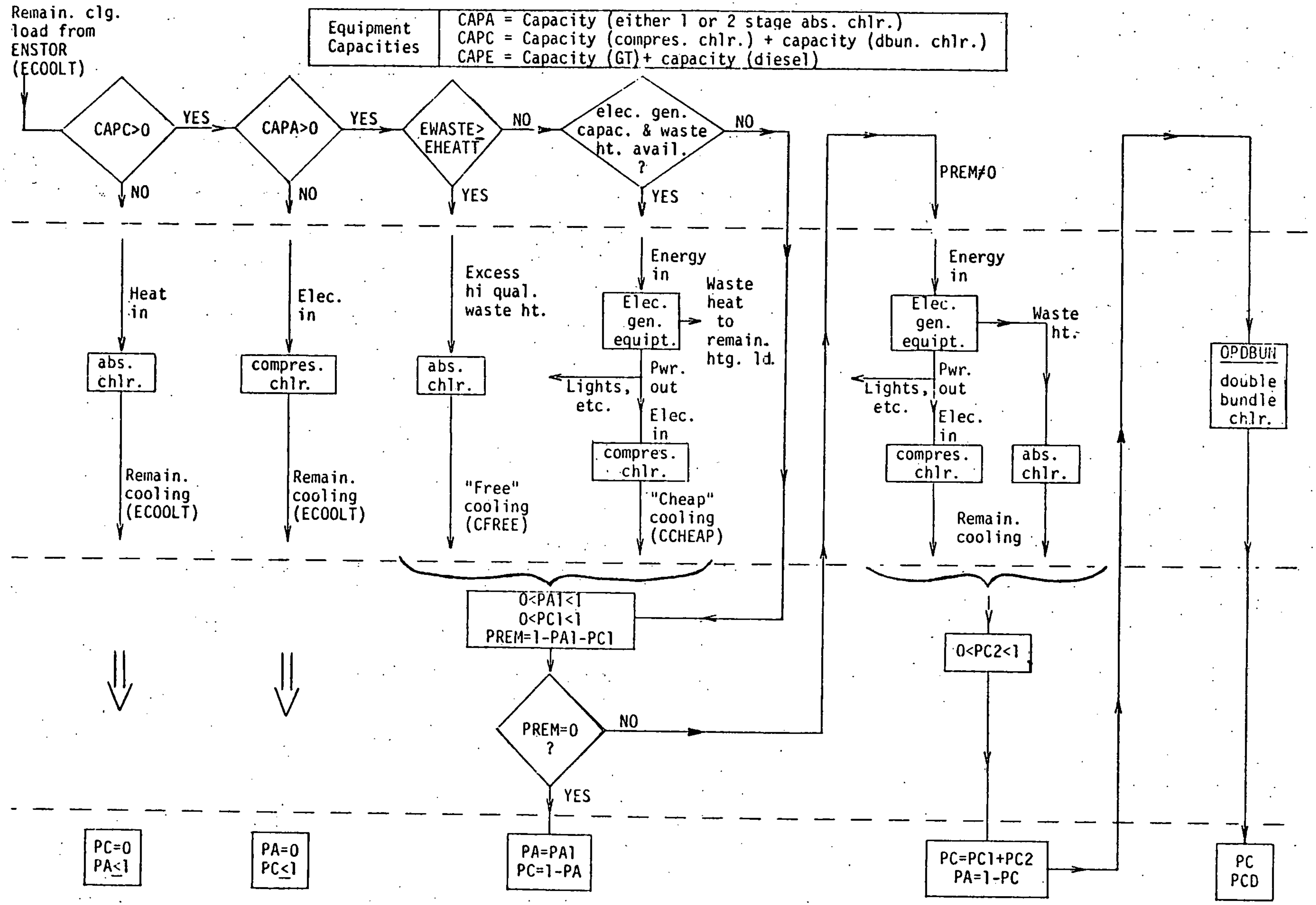
Declarations

```
EQUIVALENCE (CPTYPE, SDATA(1,16)), (PEXSTUR, SDATA(1,34)), (RPMNOM, SDATA
            (1,35))
```

Input

Source of Data	Name	Description
CCBTEPS	EELECT	Total electrical energy load (Btu/hr)
CCBTEPS	EHEATT	Heating load remaining after adjustment for stored energy (Btu/hr)
CCBTEPS	ECOOLT	Total cooling load (Btu/hr)
/EPARS/	IABSOR	Absorption chiller type, same as in ABSREF
/EPARS/	ICOMPR	Compression chiller type, same as in COMREF
/EPARS/	TOTCAP (ICOMPR)	Total capacity of compression chiller type ICOMPR (Btu/hr)
/EPARS/	TOTCAP(1)	Total nominal capacity of gas-turbine engine (Btu/hr)
/EPARS/	TOTCAP(2)	Total nominal capacity of diesel engine (Btu/hr)
/EPARS/	TOTCAP (IABSOR)	Total nominal capacity of absorption chiller type IABSOR (Btu/hr)
SDATA(1,16)	CPTYPE	Plant Type = 1 for utility only = 2 for mixed plant

OPCOOL Functional Chart



SDATA(1,34)	PEXSTUR	Nominal exhaust steam pressure (psig)
SDATA(1,35)	RPMNOM	Nominal speed of steam turbine (rpm)
/STMTUR/	FSTMTUR	Flow of steam entering steam turbine (lb/hr)
/STMTUR/	HSTMTUR	Specific enthalpy of superheated high-pressure steam (Btu/hr)
/STMTUR/	TEXSTM	Temperature of exhaust steam (°F)

Output

Name	Description
PA	Ratio of absorption chiller load to total cooling load
PC	Ratio of compression chiller load to total cooling load
PCD	Ratio of double-bundle chiller load to total cooling load

Calculation Procedure

1. Initialize output variables.

   PA = 0.
   PC = 1.

2. If no cooling is required or no absorption chilling is installed, skip optional distribution calculation.

   If ECOOLT $\leq$ 0 or TOTCAP(IABSOR) = 0, skip to 28.
   Otherwise, continue.

3. Set absorption chiller, compression chiller (single and double-bundle), and electrical generating (gas turbine and diesel) capacities.

   CAPA = TOTCAP(IABSOR)
   CAPC = TOTCAP(IABSOR) + TOTCAP(13)
   CAPE = TOTCAP(1) + TOTCAP(2)

4. If compression chiller capacity, CAPC, $\leq$ 0,

   PC = 0.,
   PA = AMIN1(CAPA, ECOOLT)/ECOOLT (i.e.,PA $\leq$ 1), and
   skip to 28.

   Otherwise, continue.

5. If absorption chiller capacity, CAPA, $\leq$ 0,

   PA = 0.,
   PC = AMIN1(CAPC, ECOOLT)/ECOOLT (i.e., PC $\leq$ 1), and
   skip to 28.

   Otherwise, continue.

6. Compute average specific energy consumption (inverse of efficiency) of boiler.

   SB = 1/0.75

7. Set average specific energy consumption of electrical generation for either of the following cases: (1) off-site generation, (2) a steam turbine is the only installed on-site electrical generation equipment.

SG = 5.

8. If either a gas turbine and/or a diesel generator is installed (CAPE>0), recalculate the average specific energy consumption of electrical generation and calculate the ratio of waste heat to electrical power output, CEX, for on-site generation.

   SG = (3.*TOTCAP(2) + 5.*TOTCAP(1))/CAPE
   = weighted average between diesel (SG = 3) and gas turbine (SG = 5), and
   CEX = 0 (initialized).
   If CAPE > 0,
   CEX = [(1.7*TOTCAP(2) + 2.* TOTCAP(1))/CAPE
   = weighted average between diesel (CEX=1.7) and gas turbine (CEX=2).

9. Calculate average specific energy consumption of compression chillers.

   SC = 0.2.

10. Calculate average specific energy consumption of absorption chillers.

    Single-stage: SA = 1.5.
    Double-stage: If IABSOR > 5, SA = 0.9

11. Initialize intermediate variables.

    EWASTE = CFREE = CCHEAP = 0.
    where EWASTE = waste heat energy (Btu/hr),
    CFREE = "free" cooling (Btu/hr), and
    CCHEAP = "cheap" cooling (Btu/hr)
    (see OPCOOL Functional Chart).

12. If there is either no electrical energy requirement (EELECT < 0) or there is neither a gas turbine nor a diesel generator installed (CAPE≤0), skip the calls to the electrical generation routines (skip to 18). Otherwise, continue.

13. Simulate electrical generation.

    CALL OPELEC
    CALL LDIST
    CALL GASTUR
    CALL LDIST
    CALL DIESEL
    CALL LDIST
    CALL STMTUR

14. Calculate energy input to steam turbine (Btu/hr).

    EBOILS = FSTMTUR * (HSTMTUR-TEXSTM+32.) * SB

    Note that - TEXSTM + 32 = - (TEXSTM -32) = - enthalpy of saturated liquid at the exhaust temperature.

15. Calculate total waste heat from electrical generation(Btu/hr).

    EWASTE = EEXG + ELUBEG + EJACKD + ELUBED,
    where EEXG = exhaust heat from gas turbine,
    ELUBEG = lube oil heat from gas turbine,
    EJACKD = jacket heat from diesel,
    EEXD = exhaust heat from diesel, and
    ELUBED = lube oil heat from diesel.

16. Calculate ratio of total waste heat to electrical power output (load).

    CEX = EWASTE/EELECT

17. Recalculate average specific energy consumption of electrical generation, now including the steam turbine and based on actual operating conditions.

    SG = (EFUELG + EFUELD + EBOILS)/EELECT,

    where EFUELG = fuel energy input rate for gas turbine (calculated in GASTUR), and

    EFUELD = fuel energy input rate for diesel (calculated in DIESEL).

    Note that if the only on-site generating equipment is a steam turbine, its specific energy consumption is calculated in Step 7.

18. Calculate "free" cooling, CFREE (Btu/hr), by using the available portion of high quality waste heat (EEXG+EEXD) to drive the absorption chiller.

    CFREE = AMAX1(EEXG+EEXD-AMAX1(EHEATT-EJACKD, 0.), 0.)/SA.

    Note that the heat load EHEATT will later (in HEATREC) be met, or partially met, by using recovered waste heat. The remaining heat load to which the high quality waste heat will be applied is EHEATT - EJACKD. Thus, the available portion of the high quality waste heat = EEXG + EEXD - AMAX1 (EHEATT-EJACKD, 0.).

19. If either (1) the electrical generators that produce high quality heat (gas turbine or diesel) are in full use and meet the electrical load (EELECT>CAPE) or (2) no high-quality waste heat is available from electrical generating equipment (gas turbine or diesel), then no "cheap" cooling is possible and skip to 20. Otherwise, calculate "cheap" cooling using compression chillers driven by electrical generation. The incremental waste heat from this electrical generation will be used to meet the remaining heating load. Thus,

    CCHEAP = AMAX1 (EHEATT-EWASTE+ELUBED+ELUBEG, 0.)/(SC*CEX).

    Note that the heat load EHEATT will later (in HEATREC) be met, or partially met, by using recovered waste heat. However, the lube oil heat is too low in temperature and cannot be so used. Thus, the waste heat applicable to the heating load is EWASTE - ELUBED - ELUBEG, and the remaining heating load is EHEATT - (EWASTE-ELUBED-ELUBEG).

    Note also that the production of free cooling precludes the production of cheap cooling and vice versa.

20. Calculate the fractions of the total cooling load carried by absorption and compression chillers.

    PA1 = AMIN1(CAPA, CFREE)/ECOOLT,
    PC1 = AMIN1(CAPC, CCHEAP)/ECOOLT, and
    PREM = 1. - PA1 - PC1,

    where the limit of free cooling is the absorption chiller capacity, the limit of cheap cooling is the compression chiller capacity, and PREM is the fraction of total cooling load not satisfied by free or cheap cooling (remaining load).

21. If there is no remaining cooling load, calculate final absorption and compression fractions; otherwise, skip to 22.

PA = AMIN1(PA1, 1.)
PC = 1. - PA
Skip to 28.

22. The remaining cooling load is distributed between absorption and compression chillers so that the steam required by the absorption chiller is generated by the waste heat released, while generating the electrical input to the compression chillers to supply their share of the remaining cooling load (unconstrained optimization) (see the OPCOOL Functional Chart).

    PC2 = PREM * SA/(SA+SC*CEX)

    Thus if the absorption chiller COP is high, PC2 → 0 and the remaining cooling is supplied by the absorption chiller. If the COP is low, PC2 → 1.

23. Compare absorption and compression specific energy consumption.

    If (PREM * SA * SB) < (PC2 * SC * SG), PC2 = 0,

    that is, if the specific energy consumption of the absorption chiller system < specific energy consumption of the compression chiller system, all the remaining cooling load is supplied by the absorption chiller.

24. Imposing capacity constraints, adjust the ratios (constrained optimization). First the compression chiller capacity constraint:

    PC = AMIN1((P01 + PC2), CAPC/ECOOLT, 1.);

    that is, (PC1 + PC2) ≤ (CAPC/ECOOLT).

25. If the electrical load is all supplied by utility, skip on-site generating capacity constraint.

    If CPTYPE = 1, skip to 27.

26. Impose electrical generating capacity constraint.

    PC = AMIN1(PC, AMAX1((CAPE-EELECT),0)/(SC*ECOOLT))
    where (CAPE-EELECT) = available generating capacity. Thus,
    PC * SC * ECOOLT ≤ (CAPE-EELECT)
    or electrical input to chiller ≤ available electrical capacity.

27. Impose absorption chiller capacity constraint.

    PA = AMIN1((1.-PC),CAPA/ECOOLT);
    that is, PA < (CAPE/ECOOLT)
    and PC = AMIN1((1.-PA, CAPC/ECOOLT).

28. Call solar cooling routine and adjust load distribution to account for cooling provided by solar-driven absorption chiller (assumed to be single-stage with SA = 1.5).

    CALL SOCOOL (EHEATT, ECOOLT, PA, PC)

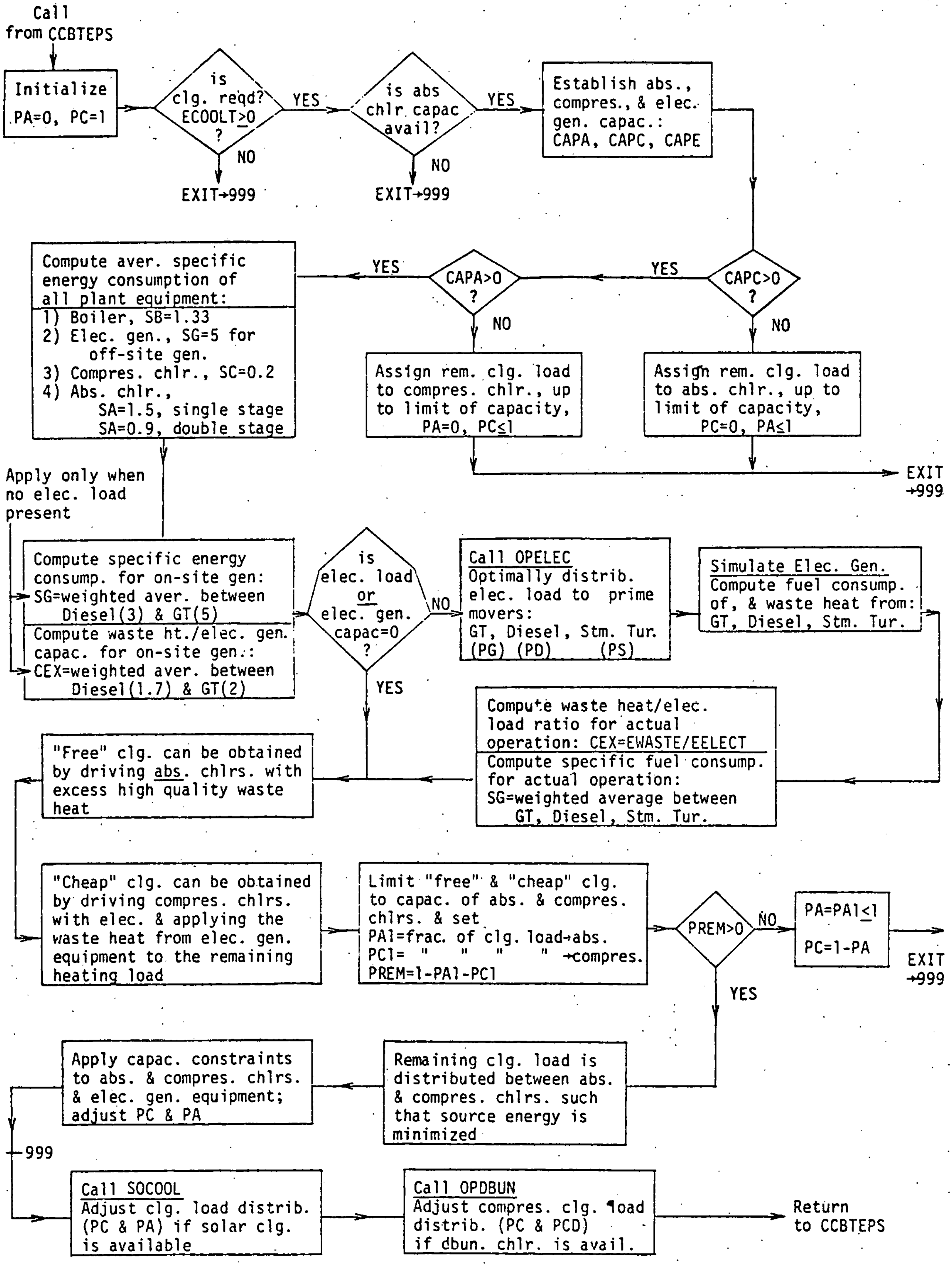
29. Call double-bundle chiller optimization routine and adjust load to account for cooling load distributed to compression and double-bundle chillers.

    CALL OPDBUN (ECOOLT, EHEATT, PC, PCD)

ASHRAE Verification

There is no applicable ASHRAE optimization routine.

Subroutine OPCOOL Flow Chart


Call from CCBTEPS
Initialize PA=0, PC=1
is clg. reqd? ECOOLT≥0 ?
YES
NO
EXIT→999
is abs chlr capac avail?
YES
NO
EXIT→999
Establish abs., compres., & elec. gen. capac.: CAPA, CAPC, CAPE
CAPC>0 ?
YES
NO
Assign rem. clg. load to abs. chlr., up to limit of capacity, PC=0, PA≤1
CAPA>0 ?
YES
NO
Assign rem. clg. load to compres. chlr., up to limit of capacity, PA=0, PC≤1
EXIT →999
Compute aver. specific energy consumption of all plant equipment:
1) Boiler, SB=1.33
2) Elec. gen., SG=5 for off-site gen.
3) Compres. chlr., SC=0.2
4) Abs. chlr., SA=1.5, single stage SA=0.9, double stage
Apply only when no elec. load present
Compute specific energy consump. for on-site gen: SG=weighted aver. between Diesel(3) & GT(5)
Compute waste ht./elec. gen. capac. for on-site gen.: CEX=weighted aver. between Diesel(1.7) & GT(2)
is elec. load or elec. gen. capac=0 ?
NO
YES
Call OPELEC
Optimally distrib. elec. load to prime movers: GT, Diesel, Stm. Tur. (PG) (PD) (PS)
Simulate Elec. Gen.
Compute fuel consump. of, & waste heat from: GT, Diesel, Stm. Tur.
Compute waste heat/elec. load ratio for actual operation: CEX=EWASTE/EELECT
Compute specific fuel consump. for actual operation: SG=weighted average between GT, Diesel, Stm. Tur.
"Free" clg. can be obtained by driving abs. chlrs. with excess high quality waste heat
"Cheap" clg. can be obtained by driving compres. chlrs. with elec. & applying the waste heat from elec. gen. equipment to the remaining heating load
Limit "free" & "cheap" clg. to capac. of abs. & compres. chlrs. & set
PA1=frac. of clg. load→abs.
PC1= " " " " →compres.
PREM=1-PA1-PC1
PREM>0
NO
PA=PA1≤1
PC=1-PA
EXIT →999
YES
Remaining clg. load is distributed between abs. & compres. chlrs. such that source energy is minimized
Apply capac. constraints to abs. & compres. chlrs. & elec. gen. equipment; adjust PC & PA
999
Call SOCOOL
Adjust clg. load distrib. (PC & PA) if solar clg. is available
Call OPDBUN
Adjust compres. clg. load distrib. (PC & PCD) if dbun. chlr. is avail.
Return to CCBTEPS

s. Subroutine OPDBUN.

SUBROUTINE OPDBUN (ECOOL, EHEAT, PC, PCD)

Description

This subroutine calculates the ratio of double-bundle chiller load to total cooling load. Given cooling and heating load, available double-bundle capacity is calculated by simulating the double-bundle chiller. Load is distributed to double-bundle and compression chillers so that all of the recoverable heat is used when possible.

Subprograms Calling This Routine

SUBROUTINE OPCOOL

Subprograms Called by This Routine

SUBROUTINE DBUNDLE

Common Blocks

EDATA, EPARS

Declarations

None

Input

Source of Data	Name	Description
OPCOOL	ECOOL	Total cooling load (Btu/hr)
OPCOOL	EHEAT	Total heating load (Btu/hr)
OPCOOL	PC	Ratio of compression chiller (including double-bundle) load to total cooling load
/EPARS/	ICOMPR	Compression chiller type, same as in COMREF
/EPARS/	TOTCAP(ICOMPR)	Total capacity of compression chiller of type ICOMPR (Btu/hr)
/EPARS/	TOTCAP(13)	Double-bundle chiller capacity (Btu/hr)
/EDATA/	CNOM(1,13)	Nominal size of double-bundle chiller of size 1 (Btu/hr)

Output

Name	Description
PC	Ratio of compression chiller (excluding double-bundle) load to total cooling load
PCD	Ratio of double-bundle chiller load to total cooling load

Calculation Procedure

1. Initialize intermediate variable.

   RATPCD = 0.,

   where RATPCD is the ratio of PCD to PC = ratio of double-bundle load to total compression chiller load.

2. If either PC, the double-bundle chiller capacity (TOTCAP(13)), or the total cooling load (ECOOL) is zero, skip the adjustment of PC (skip to 11). Otherwise, continue.

3. Set total compression chiller capacity.

   TOTCAPC = TOTCAP(ICOMPR) + TOTCAP(13)

4. Initialize the available cooling capcity ratio (see subroutine DBUNDLE).

   RCAV = 1.

5. If heating load < 1% of the smallest double-bundle unit capacity, set available cooling capacity to the nominal double-bundle chiller capacity, i.e., skip to step 6. Otherwise, simulate double-bundle chiller to establish available capacity.

   OPCAP(13) = TOTCAP(13)
   = operating capacity of double-bundle chiller
   CALL DBUNDLE

6. Set available cooling capacity to the nominal double-bundle chiller capacity.

   TOTCAV = TOTCAP(13) * RCAV,
   where RCAV = 1 at this point.

7. If the heating load ≤ 0, first allocate the regular compression chillers to the cooling load; then allocate the double-bundle chillers to the remainder of the compression cooling load and return.

   If EHEAT > 0, skip to 8; otherwise,
   DBLOAD = AMAX1(0., ECOOL*PC-TOTCAP(ICOMPR)),
   RATPCD = DBLOAD/(ECOOL*PC),

   where RATPCD is the fraction of compression chiller load that is double-bundle chiller load.

   RETURN

8. Allocate double-bundle chillers so that their waste heat satisfies all the heating load, subject to the available double-bundle capacity constraint.

   RATPCD = AMIN1(EHEAT/(1.2*ECOOL*PC), TOTCAV/(ECOOL*PC))

   This assumes that the (waste heat)/(double-bundle cooling load) = 1.2

   RATPCD = AMIN1(1., RATPCD), i.e., RATPCD ≤ 1

9. If double-bundle cooling load < 45% of total double-bundle capacity (empirical relation), try to allocate all of the double-bundle capacity, if necessary, to avoid inefficient operation at low part-load ratios.

   If (RATPCD*PC*ECOOL/TOTCAP(13)< 0.45,
   RATPCD = AMIN1(TOTCAV/(PC*ECOOL),1.)

10. If total allocated double-bundle load plus the regular compression chiller capacity < the compression cooling load, redistribute more load to double-bundle chillers.

    If (RATPCD*PC*ECOOL+TOTCAP(ICOMPR)*RMAX(ICOMPR))>ECOOL*PC),
    RATPCD = TOTCAV/TOTCAV+TOTCAP(ICOMPR)),

    where RMAX(ICOMPR) = maximum part-load ratio of compression chiller type ICOMPR. That is, TOTCAP(ICOMPR * RMAX(ICOMPR) is the amount of cooling that could be allocated to nondouble-bundle chillers.

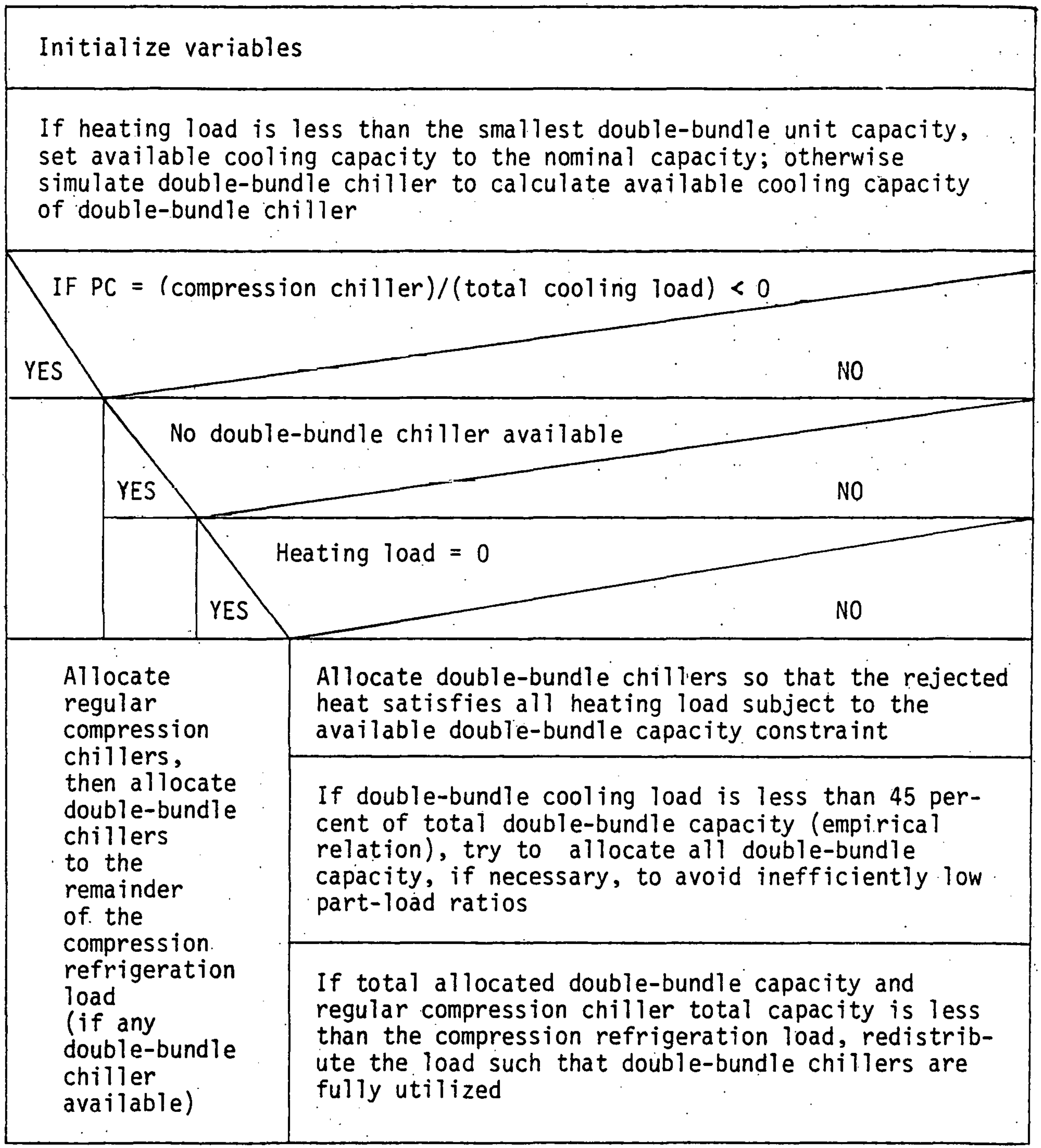
11. Calculate the ratio of double-bundle chiller load to total cooling load (PCD) and the ratio of compression chiller load to total cooling load (PC).

    PCD = PC * RATPCD
    PC = PC - PCD

ASHRAE Verification

There is no applicable ASHRAE optimization routine.

OPDBUN Flow Chart



Initialize variables

If heating load is less than the smallest double-bundle unit capacity, set available cooling capacity to the nominal capacity; otherwise simulate double-bundle chiller to calculate available cooling capacity of double-bundle chiller

IF PC = (compression chiller)/(total cooling load) < 0

YES

NO

No double-bundle chiller available

YES

NO

Heating load = 0

YES

NO

Allocate regular compression chillers, then allocate double-bundle chillers to the remainder of the compression refrigeration load (if any double-bundle chiller available)

Allocate double-bundle chillers so that the rejected heat satisfies all heating load subject to the available double-bundle capacity constraint

If double-bundle cooling load is less than 45 percent of total double-bundle capacity (empirical relation), try to allocate all double-bundle capacity, if necessary, to avoid inefficiently low part-load ratios

If total allocated double-bundle capacity and regular compression chiller total capacity is less than the compression refrigeration load, redistribute the load such that double-bundle chillers are fully utilized

t. Subroutine OPELEC.

SUBROUTINE OPELEC (EELECT, EHEAT, ECOOLT, PG, PD, PS)

Description

OPELEC distributes electrical load to three types of prime mover-generator sets: gas turbine, diesel engine, and steam turbine. It is assumed that diesel engines and gas turbines will not normally coexist in the same plant, and, if they do coexist, they will share the load proportional to total capacities.

The steam turbine load is determined with respect to the waste heat requirement. An empirical constraint on the steam turbine load is used. Average specific energy consumption factors are used for absorption chillers.

Subprograms Calling This Routine

SUBROUTINE OPCOOL

Subprograms Called by This Routine

None

Common Blocks

EDATA, EPARS

Declarations

EQUIVALENCE (D, TOTCAP(2)), (G, TOTCAP(1)), (S, TOTCAP(17))

Input

Source of Data	Name	Description
OPCOOL	EELECT	Total electrical load (Btu/hr)
OPCOOL	EHEAT	Total heating load (Btu/hr)
OPCOOL	ECOOLT	Total cooling load (Btu/hr)
/EPARS/	IABSOR	Absorption chiller type, same as in ABSREF
/EPARS/	TOTCAP(IABSOR)	Total nominal capacity of absorption chiller type IABSOR (Btu/hr)
TOTCAP(1))(EPARS)	G	Total nominal capacity of gas turbine (Btu/hr)
TOTCAP(2)(EPARS)	D	Total nominal capacity of diesel engine (Btu/hr)
TOTCAP(17)	S	Total nominal capacity of steam turbine (Btu/hr)
/EDATA/	CNOM(1,17)	Nominal size of steam turbine with size 1 (Btu/hr)
/EDATA/	ROPT(17)	Optimum part-load ratio of steam turbine
/EDATA/	RMAX(1)	Maximum part-load of gas turbine

/EDATA/	RMAX(2)	Maximum part-load ratio of diesel engine
/EDATA/	RMAX(17)	Maximum part-load ratio of steam turbine

Output

Name	Description
PG	Ratio of gas turbine load to total electrical load
PD	Ratio of diesel engine load to total electrical load
PS	Ratio of steam turbine load to total electrical load

Calculation Procedure

1. Initialize output variables.

   PD = PG = PS = 0.

2. If no electrical energy requirement, EELECT ≤ 0, or no electrical generators installed, (D+G+S) ≤ 0, skip calculation, i.e., RETURN; otherwise, continue.

3. Calculate maximum ratios of gas turbine, diesel, and steam turbine load to total electrical load.

   PGMAX = RMAX(1) * G/EELECT
   PDMAX = RMAX(2) * D/EELECT
   PSMAX = RMAX(17) * S/EELECT

4. If no steam turbine is installed, S ≤ 0, skip turbine calculation (i.e., skip to 9); otherwise, continue.

5. If the total electrical load is too small for the steam turbine to operate at nearly optimal load ratio (i.e., if the load ratio will be < 5% below the optimal part-load ratio), do not use steam turbine.

   If EELECT≤(CNOM(1,17)*(ROPT(17)-0.05)), skip to 9.

6. Set average specific energy consumption factors for absorption chillers.

   Single-stage: SA = 1.5
   Double-stage: If (IABSOR>5) SA = 0.9

7. Calculate the ratio, CWASTE, of the total potential use for waste heat from electrical generating equipment to the total electrical load.

   CWASTE = (EHEATT+AMIN1(ECOOLT, TOTCAP(IABSOR))*SA)/EELECT

8. Impose an empirical constraint, based on CWASTE, on steam turbine load.

   IF CWASTE > 0.9, PS = AMIN1(PSMAX, ((CWASTE/0.9)-1.))

9. If neither diesel nor gas turbine is installed, (D+G) < 0, skip diesel and gas turbine load allocation (i.e., skip to 13); otherwise, continue.

10. Distribute remaining electrical generation load to diesel engines and gas turbines proportional to their total capacities.

    PREM = 1. - PS
    PD = PREM * (D/(D+G))
    PG = PREM * (G/(D+G))

11. Compare diesel and gas turbine load ratios with their maximum values based on machine capabilities.

    If (PD<PDMAX) and (PG<PGMAX),
    skip to 13; otherwise, continue.

12. Impose maximum load constraints on electrical generating equipment and adjust load ratios.

```
DGM = AMAX1(D+G+S, EELECT)
PS = S/DGM
PD = D/DGM
PG = G/DGM
```

13. RETURN

ASHRAE Verification

There is no applicable ASHRAE optimization routine.

OPELEC Flow Chart

Initialize variables
Set average specific energy consumption factor for chillers
Calculate the ratio of waste heat needed to total electrical load (CWASTE)
Impose an empirical constraint on steam turbine load (in terms of CWASTE) Determine steam turbine load
Distribute remaining electrical generation load to diesel engines and gas turbines proportional to their total capacities

u. Subroutine REPORT.

Description

REPORT is used to print requested results. It calls R1PRNT or STATPR, depending on an input variable, as many times as indicated by variable NREPOP.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

SUBROUTINE R1PRNT
SUBROUTINE STATPR

Common Blocks

REPOPT

Declarations

None

Input

Source of Data	Name	Description
/REPOPT/	IREPOP(1,I) (I = 1,NREPOP)	Subroutine call index: M=R1PRNT S=STATPR
/REPOPT	NREPOP	Number of times printing subroutine is to be called

Output

None

Calculation Procedure

None

ASHRAE Verification

Inapplicable

v. Function RFACT

FUNCTION RFACT(R,A,T)

Description

This function calculates rating factors (tower units/gpm of tower water flow rate) that characterize cooling tower performance as a function of the range, approach, and wet-bulb temperature. A tower unit is defined as the area required to cool 1 gpm at a standard set of conditions, namely, 90°F inlet water, 80°F outlet water, and 70°F wet-bulb (see Ref. 14, p. 21.11).

The rating factor is calculated as follows:

Rating Factor RFACT = RF * $f_1(x_1,c^1)$

where f_1 = a quadratic polynomial in x_1, and c^1 is a set of polynomial coefficients,
x_1 = range = (tower water inlet temperature) - (tower water outlet temperature),
c^1 = RFR, and
RF = a function of wet-bulb temperature that is calculated based on six different approach temperatures.

For the first five approaches, the factor RF = $(A*f_1(x_1,c^1)+B*f_2(x_1,c^1))/2$,

where f_1 and f_2 = quadratic polynomials in x_1,
x_2 = wet-bulb temperature (°F),
c^1 = a set of three polynomial coefficients
= RF1 for first approach (6-8°F)
= RF2 for second approach (8-10°F)
= RF3 for third approach (10-12°F)
= RF4 for fourth approach (12-14°F)
= RF5 for fifth approach (14-16°F;

A = a constant corresponding to each of the approaches;
c^2 = a set of three polynomial coefficients
= RF2 for first approach,
= RF3 for second approach,
= RF4 for third approach,
= RF5 for fourth approach,
= RF6 for fifth approach; and
B = another constant corresponding to each of the approaches.

In the last approach, $RF = f_1(x_1, c^1)$,

where f_1 and x_1 are as defined above and

c^1 = a set of three polynomial coefficients = RF6.

Subprograms Calling This Routine

SUBROUTINE TOWER

Subprograms Called by This Routine

None

Common Blocks

PDATA

Declarations

```
DIMENSION RF1(8), RF2(8), RF3(8), RF4(8), RF5(8), RF6(8), RFR(8)
EQUIVALENCE (RF1, PDATA(1,26)), (RF2, PDATA(1,27)), (RF3, PDATA(1,28)),
            (RF4, PDATA(1,29)),(RF5, PDATA(1,30)), (RF6, PDATA(1,31)),
            (RFR, PDATA(1,39))
```

Input

Source of Data	Name	Description
TOWER	R	R = Range = tower inlet temperature - tower water outlet temperature (°F)
TOWER	A	A = Approach = tower water outlet temperature - wet-bulb temperature (°F)
TOWER	T	Wet-bulb temperature (°F)
PDATA(1,26)	RF1	Quadratic polynomial coefficients as described in previous section
PDATA(1,27)	RF2	Quadratic polynomial coefficients as described in previous section
PDATA(1,28)	RF3	Quadratic polynomial coefficients as described in previous section
PDATA(1,29)	RF4	Quadratic polynomial coefficients as described in previous section
PDATA(1,30)	RF5	Quadratic polynomial coefficients as described in previous section

PDATA(1,31)	RF6	Quadratic polynomial coefficients as described in previous section
PDATA(1,39)	RFR	Quadratic polynomial coefficients as described in previous section

Output

Name	Description
RFACT	Rating factor for the cooling tower

Calculation Procedure

1. Assign input value approach of tower outlet water temperature to ambient wet-bulb temperature AP.

   AP = A

2. Set lower and upper bounds on approach.

   If approach < 6, AP = 6.
   If approach > 16, AP = 16.

3. Compute approach interval, ISLOT.

   ISLOT = INT(AP/2.) - 2,
   where INT(x) takes the integer portion x only.

4. Calculate the first five values of the intermediate variables RFL and RFU as quadratic polynomials in the wet-bulb temperature by interpolation in five intervals of the value of approach.

   RFL = RF(I)(1) + RF(I)(2) * T + RF(I)(3) * T * T, and
   RFU = RF(I+1)(1) + RF(I+1) (2) * T + RF(I+1)(3) * T * T,

   where T = wet-bulb temperature
   and I = 1 for 6-8°F approach,
   I = 2 for 8-10°F approach,
   I = 3 for 10-12°F approach,
   I = 4 for 12-14°F approach, and
   I = 5 for 14-16°F approach.

   See RFACT listing for detailed equations.

5. Calculate the intermediate variable RF, depending upon the value of approach.

   RF = (DFL * RFU + DFU * RFL)/2.,
   where DFL = AP - 6, DFU = 8 - AP for 6-8 °F approach,
   DFL = AP - 8, DFU = 10-AP for 8-10 °F appraoch,
   DFL = AP - 10, DFU = 12 - AP for 10-12 °F approach,
   DFL = AP - 12, DFU = 14 - AP for 12-14 °F approach, and
   DFL = AP - 14, DFU = 16 - AP for 14-16 °F approach.

   For the sixth approach interval,

   RFL = RF6(1) + RF6(2) * T + RF6(3) * T * T.

6. Calculate rating factor.

RFACT + RF *[C(1)+C(2)+C(3)*R*R],

where R = range and C = PFR.

ASHRAE Verification

ASHRAE documentation (Ref. 2) expresses the tower water outlet temperature as the product of polynomials in the wet-bulb and tower water inlet temperatures. This formulation requires a different set of coefficients for each size of cooling tower.

On the other hand, the rating factor formulation of cooling tower performance used in Cal-ERDA generalizes the cooling tower size on the basis of relative size/gpm, but still takes into account the dependence on wet-bulb and tower water inlet temperature. This rating factor method is recommended by ASHRAE in Ref. 14. The rating charts given in Ref. 14 have been curve fit in the Cal-ERDA routine.

RFACT Flow Chart

Calculate AP = approach of wet-bulb air temperature to water temperature leaving tower
Set lower and upper limits on approach
Determine quadratic coefficients RF based on wet-bulb temperature
Calculate rating factor RFACT

w. Subroutine RTPRNT.

Description

This is a subroutine to print report titles.

Subprograms Calling This Routine

SUBROUTINE RIPRNT
SUBROUTINE STATPR

Subprograms Called by This Routine

None

Common Blocks

CDPR, TITLED

Declarations

None

Input

Source of Data	Name	Description
/CDPR/	IPRT	Logical unit number of print output
/TITLED/	NTIT	Number of title index IT

Output

Name	Description
ITIT(1-8,IT)	Title data of index IT(80H...FORMAT)

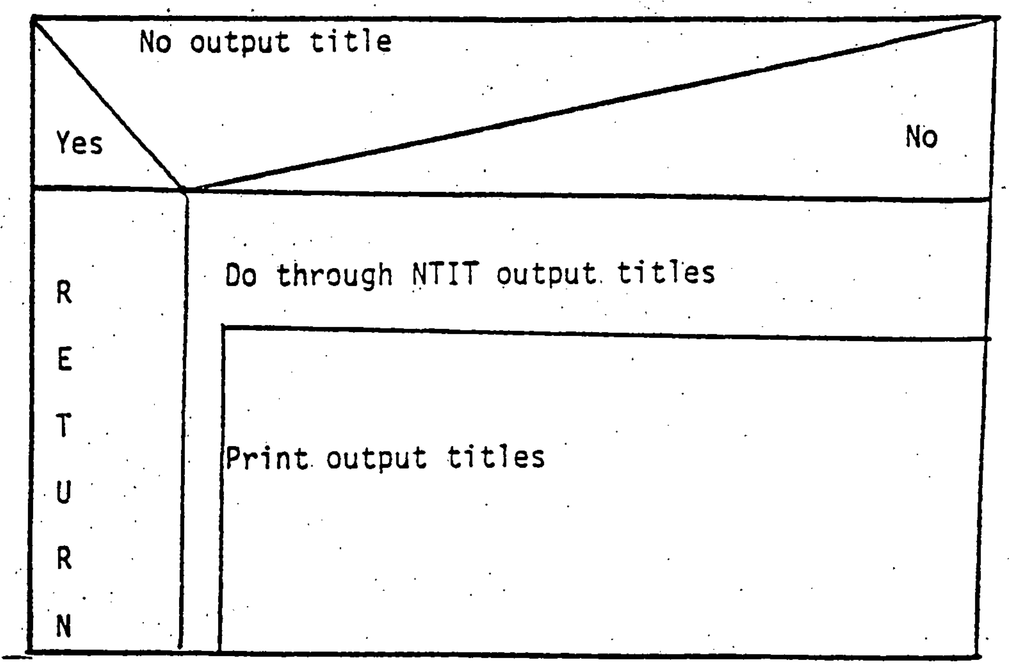
Calculation Procedure

None

ASHRAE Verification

Inapplicable

RTPRNT Flow Chart



x. Subroutine R1PRNT.

Description

This subroutine prints monthly values.

Subprograms Calling This Routine

SUBROUTINE REPORT

Subprograms Called by This Routine

SUBROUTINE RTPRNT

Common Blocks

CDPR, PRNTA1

Declarations

```
DIMENSION IHDNG(5,13)
DIMENSION PRNTAS(12)
DATA IHDNG1/

7H        , 7H        ,7HMONTH  , 7H        , 7H-----  ,
7H TOTAL  , 7H HEAT   ,7HENERGY , 7H(GBTU)  , 7H------ ,
7H TOTAL  , 7HELECTR  ,7HENERGY , 7H(GBTU)  , 7H------ ,
7H        , 7HCOOLING ,7H ENERGY, 7H (GBTU), 7H-------,
7H        , 7HRCVRED  ,7HENERGY , 7H(GBTU)  , 7H------ ,
7H WASTED, 7HRCVRABL,7H ENERGY, 7H (GBTU), 7H-------,
7HHEAT EN, 7H INPUT ,7HCOOLING, 7H (GBTU), 7H-------,
7HELEC EN, 7H INPUT ,7HCOOLING, 7H (GBTU), 7H-------,
7H ENERGY, 7H INPUT ,7HHEATING, 7H (GBTU , 7H-------,
7H ENERGY, 7H INPUT ,7HELECTRC, 7H (GBTU), 7H-------,
7H TOTAL , 7H FUEL  , 7H INPUT, 7H(GBTU) , 7H-------,
7H TOTAL , 7H ENERGY, 7H INPUT, 7H(GBTU) , 7H-------,
7HAVERAGE, 7H PLANT , 7H EFFIC, 7H(PERCT), 7H-------/

DATA IBLNK/10H          /, IEQSGN/10H==========/
```

Input

Source of Data	Name	Description
/CDPR/	IPRT	Logical unit number for print output
REPORT	UNIT	= METRIC (6H . . . FORMAT) for SI units = ENGLISH (7H . . . FORMAT) for English units

Output

Name	Description
PRNTA1 (IMON,I)	Monthly output information; same as described in CCBTEPS

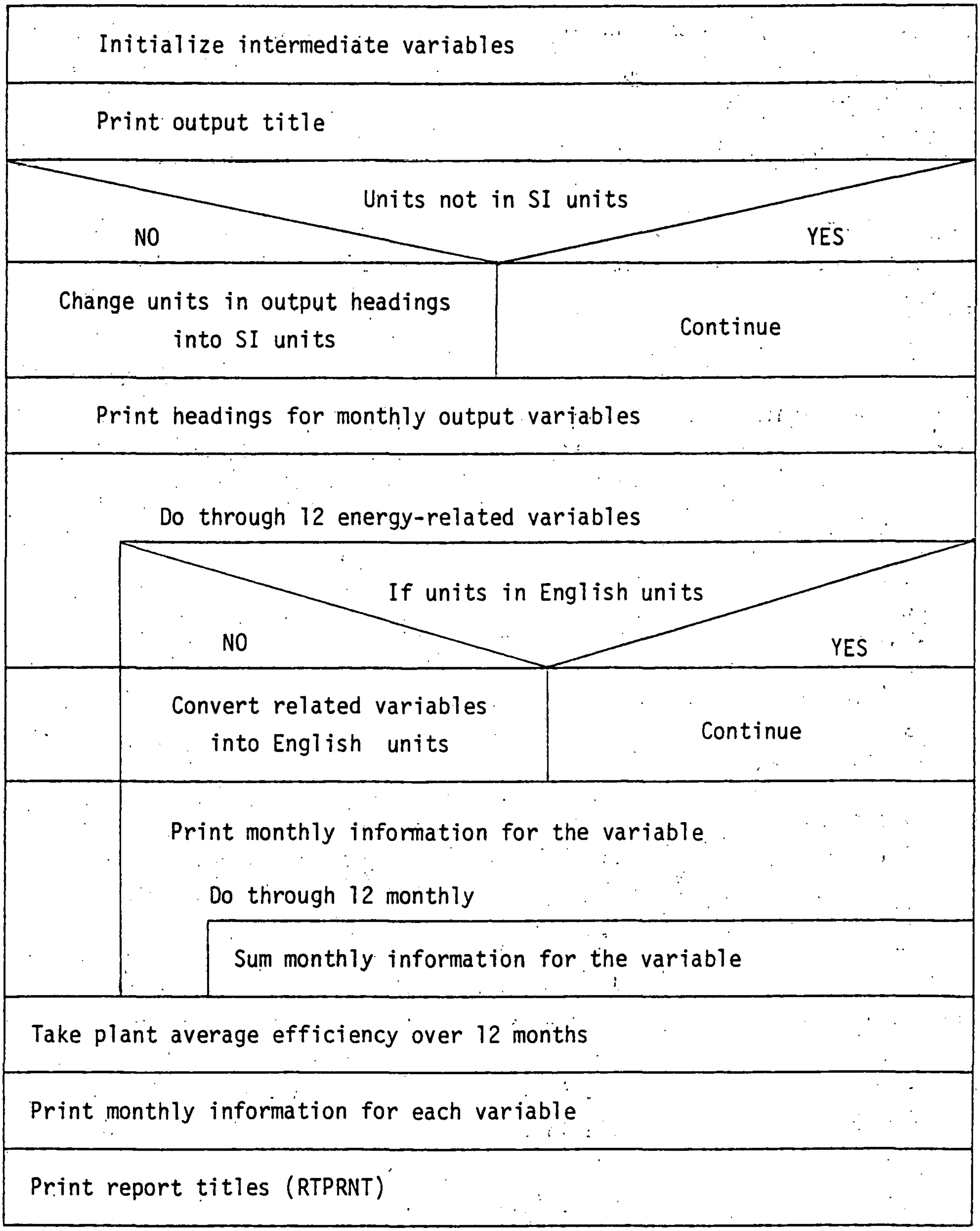
Calculation Procedure

None

ASHRAE Verification

Inapplicable

R1PRNT Flow Chart


Initialize intermediate variables
Print output title
Units not in SI units
NO
YES
Change units in output headings into SI units
Continue
Print headings for monthly output variables
Do through 12 energy-related variables
If units in English units
NO
YES
Convert related variables into English units
Continue
Print monthly information for the variable
Do through 12 monthly
Sum monthly information for the variable
Take plant average efficiency over 12 months
Print monthly information for each variable
Print report titles (RTPRNT)

y. Subroutine R3PRNT.

Description

This is a subroutine to print all internal parameters.

Subprograms Calling This Routine

SUBROUTINE DESVAR

Subprograms Called by This Routine

None

Common Blocks

CDPR, DISTB, EDATA, KEYLST, LDISTD, PDATA, REPOPT, SDATA, TITLED

Declarations

None

Input

Source of Data	Name	Description

/EDATA/
IENAME(204,I)
CNOM(J,I)
KINS(J,I)
KAV(J,I)
	(ES) Data:	Data for equipment size; same as described in CCBTEPS

/EDATA/
RMIN(I)
RMAX(I)
ROPT(I)
PEL(I)
	(ER) Data:	Data for part-load ratios; same as described in CCBTEPS

/SDATA/ SDATA(J,I)	(S) Data:	Data for special parameters, same as described in CCBTEPS
/PDATA/ PDATA(J,I)	(EP) Data:	Data for equipment performance coefficients
/TITLED/ NTIT ITIT(I,IT)	(T) Data:	Report title; same as described in RTPRNT
/EDATA/	(EA) Data:	Equipment assignment data
/EDATA/	IENAME(J,I)	Same as described in CCBTEPS
/EDATA/	NEQSIZE(I)	Same as described in CCBTEPS
/DISTB/	NDISTB	Same as described in LDIST
/DISTB/	DISTB(J,I)	Same as described in LDIST
/DISTB/	IDISTB(J,K,I)	Same as described in LDIST
/CDPR/	IPRT	Logical unit number of print output
/KEYLST/	KEYLST(J,I)	Data card type name (30H...FORMAT)

Output

Output data are the same as input data and are output to the printer.

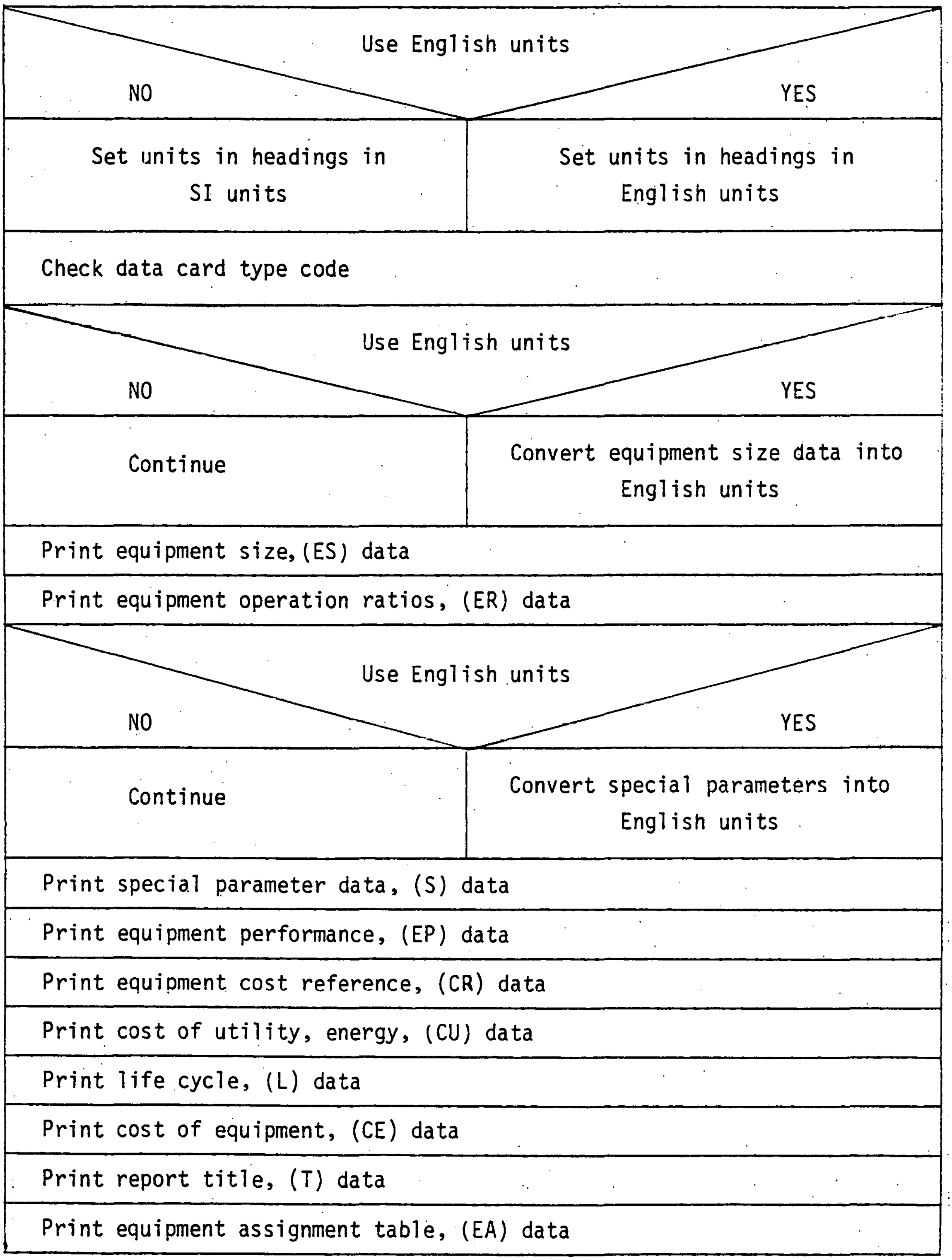
Calculation Procedure

None

ASHRAE Verification

Inapplicable

R3PRNT Flow Chart



```
Use English units
    NO:  Set units in headings in SI units
    YES: Set units in headings in English units
Check data card type code
Use English units
    NO:  Continue
    YES: Convert equipment size data into English units
Print equipment size,(ES) data
Print equipment operation ratios, (ER) data
Use English units
    NO:  Continue
    YES: Convert special parameters into English units
Print special parameter data, (S) data
Print equipment performance, (EP) data
Print equipment cost reference, (CR) data
Print cost of utility, energy, (CU) data
Print life cycle, (L) data
Print cost of equipment, (CE) data
Print report title, (T) data
Print equipment assignment table, (EA) data
```

z. Function SATUR.

Function SATUR(PSTEAM)

Description

SATUR calculates the saturation temperature (°F) as a function of steam pressure (psig). This is applicable for temperatures of 200-700 °F.

Subprograms Calling This Routine

SUBROUTINE STMTUR
FUNCTION ENTROP

Subprograms Called by This Routine

None

Common Blocks

None

Declarations

None

Input

Source of Data	Name	Description
STMTUR or ENTROP	PSTEAM	Steam pressure (psig)

Output

Name	Description
SATUR	Steam saturation temperature (°F)

Calculation Procedure

The calculation uses the single equation SATUR = 1./(0.0017887-0.00011429 *ALOG(PSTEAM+14.7)) - 460.

ASHRAE Verification

There is no applicable ASHRAE algorithm. This function was taken directly from the NECAP program (Ref. 3, Vol. II, pp. 5-124).

aa. Subroutine SOCOOL.

SUBROUTINE SOCOOL (EHEAT, ECOOL, PA, PC)

Description

SOCOOL calculates solar heat energy used for cooling and adjusts the cooling load distribution to the absorption and compression chillers. It is called from subroutine OPCOOL after the solar simulator has computed the solar energy collected, stored, and used for heating. SOCOOL tries to use the remaining solar heat in a single-stage absorption chiller to satisfy cooling load.

Subprograms Calling This Routine

SUBROUTINE OPCOOL

Subprograms Called by This Routine

None

Common Blocks

EPARS, SDATA, SOLARD, BLKEL

Declarations

EQUIVALENCE (TMINH, SDATA(1,14)), (TMINC, SDATA(1,15))

Input

Source of Data	Name	Description
OPCOOL	EHEAT	Heating load (Btu/hr)
OPCOOL	ECOOL	Cooling load (Btu/hr)
/SOLARD/	ESOLAR	Available stored solar energy (Btu)
/SOLARD/	TSOLAR	Solar heat storage tank temperature (°F)
/EPARS/	IABSOR	Absorption chiller type as in subroutine ABSREF.
/BLKEL/	TNKT	Solar storage tank temperature (°F)
/BLKEL/	TNKMCP	Tank thermal capacity, MC_p (Btu/°F)
/BLKEL/	ISOLEQ	Solar equipment available flag (=0, if not available; =1. if available
SDATA(1,15)	TMINC	Minimum solar storage tank temperature required for cooling (°F)

Output

Name	Description
PA	Ratio of absorption chilling to total cooling load, with solar
PC	Ratio of compression chilling to total cooling load, with solar
ESOLC	Solar energy used for cooling (Btu/hr)
CSOLAR	Solar energy used for cooling (Btu/hr)

Calculation Procedure

1. Initialize output variables.

ESOLH = ESOLC = 0.
PAO = PA
PCO = PC,

where PAO = ratio of absorption cooling to total cooling load without solar, and

PCO = ratio of compression cooling to total cooling load without solar.

2. If either no solar collectors are installed (ISOLEQ = 0) or no single-stage absorption chiller is installed (IABSOR>5), or if there is no cooling load (ECOOL=0), RETURN. Otherwise, continue.

3. Calculate available stored solar energy at this hour.

   ESOLAR = (TNKT-TMINC) * TNKMCP

4. Calculate the available solar energy remaining after heating has been serviced.

   ESOLREM = ESOLAR

5. If either no available solar heat is remaining (ESOLREM<0) or the solar tank temperature is less than its minimum effective value for absorption cooling (TNKT<TMINC), RETURN. Otherwise, continue.

6. Try to satisfy cooling load by absorption cooling. Set free-of-load absorption cooling capacity to the (total absorption cooling capacity) - (absorption cooling capacity operating to satisfy its assigned share of the cooling load without solar heat), subject to lower and upper bounds.

   Minimum free-of-load absorption cooling capacity = 0.
   Maximum free-of-load absorption cooling capacity = the compression cooling capacity operating to satisfy its assigned share of the cooling load = PCO * ECOOL

   CAPAREM = AMAX1(TOTCAP(5)-PAO*ECOOL),0.)
   CAPFREE = AMIN1(CAPAREM,(PCO*ECOOL))

7. Set average specific energy consumption for single-stage absorption chiller.

   SA = 1.5

   Note that this is equivalent to a COP = 1/SA = 1/1.5 = 0.67. In actual practice, a solar hot-water fixed absorption chiller will have an operating COP in the range of 0.4 to 0.7. The assumed average value of 0.67 may therefore produce unrealistic results.

8. Calculate heat input required to drive the absorption chiller.

   HEATREQ = CAPFREE * SA

9. Set solar heat used for cooling to the heat required for absorption cooling, subject to an upper bound of the remaining available solar heat.

   ESOLC = AMIN1(ESOLREM, HEATREQ)

   This value will be sent back to the main routine CCBTEPS.

10. Calculate the solar heat used for cooling that is to be sent back to the solar simulator for removal from the solar storage tank at the next hour.

    CSOLAR = ESOLC

11. Set cooling energy provided by solar.

    SOLCOOL = ESOLC/SA

12. Adjust cooling load distribution to take advantage of solar heat. Set the absorption fraction of total cooling load to the absorption fraction without solar + (the cooling energy supplied by solar/cooling load) = PAO + SOLCOOL/ECOOL, subject to an upper bound.

    Maximum absorption fraction with solar = absorption fraction without solar + compression fraction without solar (PAO+PCO)

    PA = AMIN1(PAO+SOLCOOL/ECOOL, PAO+PCO)

13. Adjust the compression fraction of total cooling with solar.

    PC = PAO + PCO - PA.

ASHRAE Verification

There is no applicable ASHRAE solar cooling algorithm.

bb. Subroutine STATIS.

Description

Subroutine STATIS calculates monthly statistics for energy-related input and output variables.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

None

Common Blocks

DATE, EDATA, EPARS, HOURTOT, LDISTD, MONTOT, STATD

Declarations

DIMENSION ENHOUR (10)

Input

Source of Data	Name	Description
/HOURTOT/	EUT	Total utility electrical power (Btu/hr)
/HOURTOT/	ENTOT	Total power required by plant and utility (Btu/hr)
/HOURTOT/	EFUELB	Total boiler fuel consumption rate (Btu/hr)
/HOURTOT/	EFUELD	Total diesel fuel consumption rate (Btu/hr)
/HOURTOT/	EFUELG	Total gas turbine fuel consumption (Btu/hr)
/HOURTOT/	EHEATT	Total heat power (Btu/hr)
/HOURTOT/	ECOOL	Total cooling load (Btu/hr)

/HOURTOT/	ERECOVR	Rate of total recovered heat energy (Btu/hr)
/HOURTOT/	EWASTED	Total wasted energy (Btu/hr)
/HOURTOT/	EFIHC	Total rate of fuel input for heat energy consumed by cooling stage (Btu/hr)
/HOURTOT/	EFIEC	Total rate of fuel input for electrical energy consumed by cooling stage (Btu/hr)
/HOURTOT/	EFUELIH	Total rate of fuel input for heat energy generation (Btu/hr)
/HOURTOT/	EFUELIE	Total rate of fuel input for electricity generation by the plant and utility (Btu/hr)
/HOURTOT/	EFUEL	Total rate of fuel energy input, including to utility (Btu/hr)
/LDISTD/	NOPR(I)	Number of different sizes of equipment operating at current time step
/EPARS/	PLOAD(I)	Part load of equipment I for current hour (Btu/hr)
/STATD/	PLOADY(I)	Part load totaled over the year of equipment I (Btu)
/EPARS/	OPCAP(I)	Operating capacity of equipment I (Btu/hr)
/STATD/	OPCAPY(I)	Operating capacity totaled over the year for equipment I (Btu)
/LDISTD/	IOPR(J,I)	Number of operating hours of equipment I, size J
/STATD/	IOPRHR(J,I)	Number of operating hours of equipment I, size J
/DATE/	IMON	Month index
/DATE/	IDAY	Day index
/DATE/	IHR	Hour index

Output

Name	
EUTS ENTOTS EFUELBS EFUELDS EFUELGS EHEATTS EELECTS ECOOLS ERECOVS EWASTES EFIHCS EFIECS EFUELHS EFUELES EFUELS	Monthly totals of energy-related variables as described in Input Data. Also see description in STATSM
ENPEAK (IMON,I)	Monthly energy peak for utility index I (Btu)
AMAXLD(I)	Maximum load of equipment I (Btu)
MAXTIM(1,I)	Month index for the time when equipment I reaches its maximum part load
MAXTIM(2,I)	Day index for the time when equipment I reaches its maximum part load
MAXTIM(3,I)	Hour index for the time when equipment I reaches its maximum part load

Calculation Procedure

1. Monthly Totals. Add this hour's value to the monthly sum value for the variables EUTS through EFUELS (see Output above).

2. Monthly Energy Peaks. Compute energy peak (Btu), ENPEAK(IMON,I), for each utility index for the month and note the day, KENPDY(IMON,I), and hour, KENPHR(IMON,I), of its occurrence in that month.

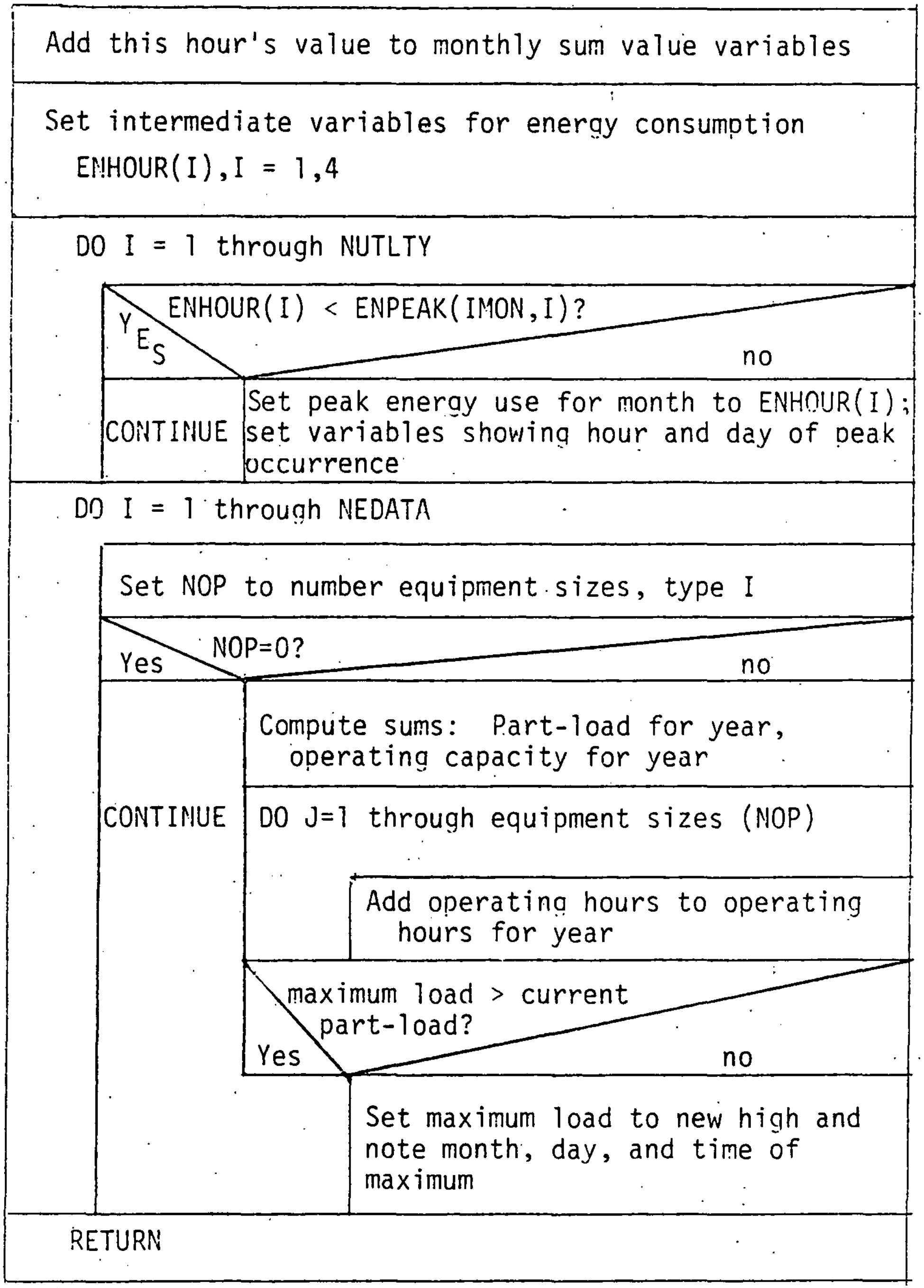
3. Maximum Loads (Annual). Compute maximum load (Btu) in AMAXLD(I) for each type of equipment for the year and note the month, MAXTIM(1,I), day, MAXTIM(2,I), and hour, MAXTIM(3,I), of its occurrence.

4. Annual Totals. Compute part load totaled over the year, PLOADY(I), operating capacity totaled over the year, OPCAPY(I), and number of operating hours for the year, IOPRHR(J,I), for each type of equipment.

ASHRAE Verification

Does not apply.

STATIS Flow Chart


Add this hour's value to monthly sum value variables
Set intermediate variables for energy consumption ENHOUR(I),I = 1,4
DO I = 1 through NUTLTY
ENHOUR(I) < ENPEAK(IMON,I)?
YES
no
CONTINUE
Set peak energy use for month to ENHOUR(I); set variables showing hour and day of peak occurrence
DO I = 1 through NEDATA
Set NOP to number equipment sizes, type I
NOP=0?
Yes
no
CONTINUE
Compute sums: Part-load for year, operating capacity for year
DO J=1 through equipment sizes (NOP)
Add operating hours to operating hours for year
maximum load > current part-load?
Yes
no
Set maximum load to new high and note month, day, and time of maximum
RETURN

cc. Subroutine STATPR.

Description

This subroutine prints equipment-use statistics such as average part-load ratio, annual operating hours and maximum loads.

Subprograms Calling This Routine

SUBROUTINE REPORT

Subprograms Called by This Routine

SUBROUTINE RTPRNT

Common Blocks

CDPR, EDATA, STATD

Declarations

None

Input

Source of Data	Name	Descriptions
/CDPR/	IPRT	Logical unit number for print output
/EDATA/	NEQSIZE(I)	Same as described in CCBTEPS
/STATD/	PLAODY(I)	Part load totaled over the year of equipment type I

Output

Name	Description
AVGOPR	Same as described in CCBTEPS
OPCAPY(I)	Same as described in CCBTEPS
IENAME(J,I)	Same as described in CCBTEPS
AMAXLD(I)	Same as described in CCBTEPS
MAXTIM(J,I)	Same as described in CCBTEPS
IOPRHR(J,I)	Same as described in CCBTEPS
CNOM(J,I)	Same as described in CCBTEPS

Calculation Procedure

1. Compute annual average part-load ratio for each equipment type,

   AVGOPR = PLOADY (IE)/OPCAPY (IE), where IE = equipment type index.

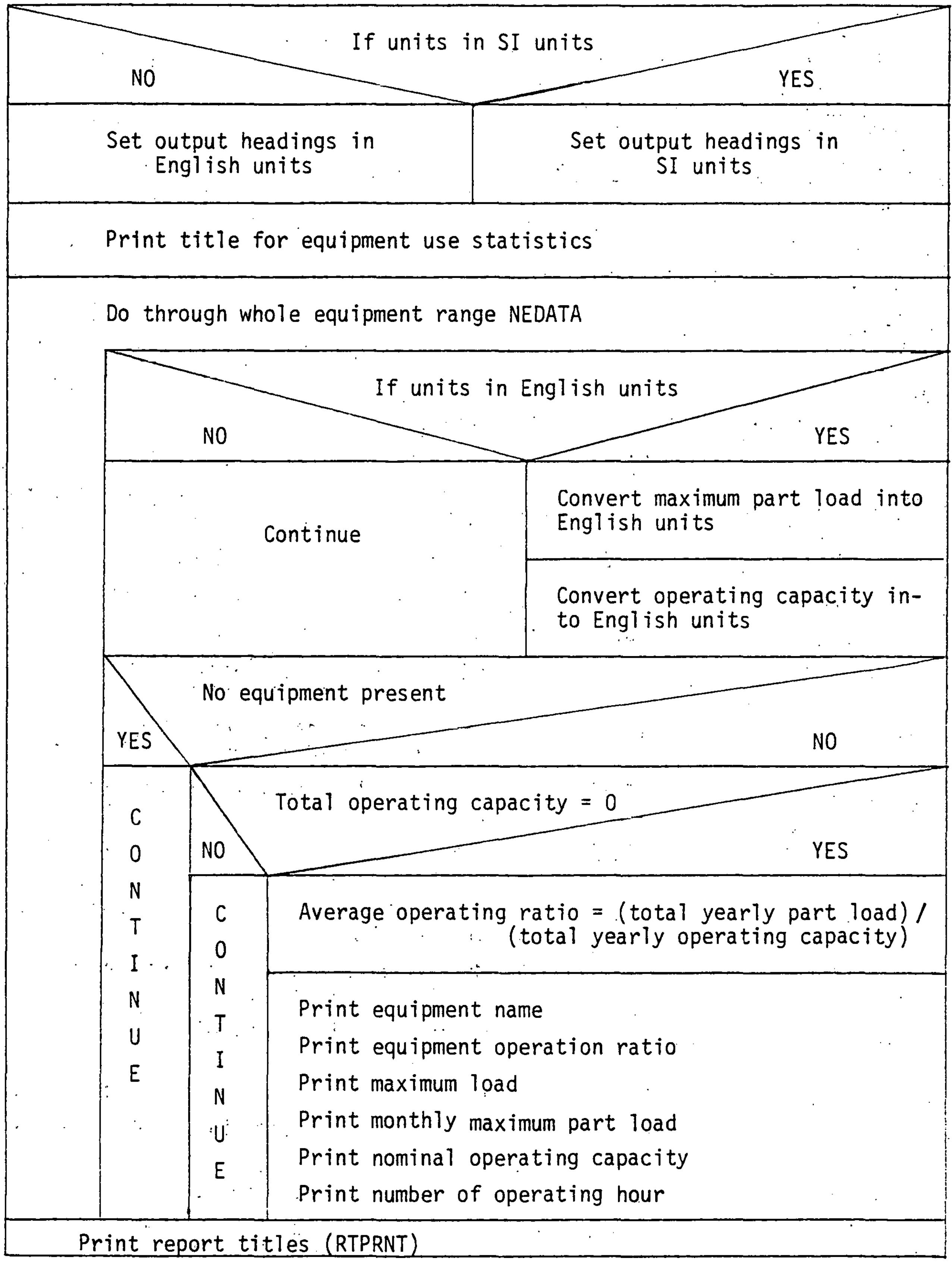
2. For each equipment type write:

   - Annual average part-load ratio,
   - Maximum load and time of occurrence,
   - Nominal capacity, and
   - Annual operating hours.

ASHRAE Verification

Does not apply

STATPR Flow Chart


If units in SI units
NO
YES
Set output headings in English units
Set output headings in SI units
Print title for equipment use statistics
Do through whole equipment range NEDATA
If units in English units
NO
YES
Continue
Convert maximum part load into English units
Convert operating capacity into English units
No equipment present
YES
NO
CONTINUE
Total operating capacity = 0
NO
YES
CONTINUE
Average operating ratio = (total yearly part load) / (total yearly operating capacity)
Print equipment name
Print equipment operation ratio
Print maximum load
Print monthly maximum part load
Print nominal operating capacity
Print number of operating hour
Print report titles (RTPRNT)

dd. Subroutine STATSM

Description

This subroutine stores energy-related monthly results.

Subprograms Calling This Routine

None

Subprograms Called by This Routine

None

Common Blocks

DATE, HOURTOT, MONTOT, PRNTA1, STATD

Declarations

None

Input

Source of Data	Name	Description
/MONTOT/	EHEATTS	Monthly total heat energy (Btu)
/MONTOT/	EELECTS	Monthly total electrical energy demanded including that used by the plant (Btu)
/MONTOT/	ECOOLS	Monthly cooling load (Btu)
/MONTOT/	ERECOVS	Monthly total recovered heat energy (Btu)
/MONTOT/	EWASTES	Monthly total wasted heat (Btu)
/MONTOT/	EFIHCS	Monthly fuel energy input for heat energy consumed by cooling stage (Btu)
/MONTOT/	EFIECS	Monthly fuel energy input for electrical energy consumed by cooling stage (Btu)
/MONTOT/	EFUELHS	Monthly total input for heat energy generation (Btu)
/MONTOT/	EFUELES	Monthly total input for electrical energy generation (Btu)
/MONTOT/	EFUELS	Monthly fuel energy input including to utility (Btu)
/MONTOT/	ENTOTS	Monthly total energy required by the plant and utility (Btu)
/MONTOT/	EUTS	Monthly total utility electrical energy (Btu)
/MONTOT/	EFUELDS	Monthly total diesel fuel consumption (Btu)
/MONTOT/	EFUELGS	Monthly total gas turbine fuel consumption (Btu)
/MONTOT/	EFUELBS	Monthly total boiler fuel consumption (Btu)
/STATD/	ENPEAK(J,I)	Monthly energy peak use rate for utility index I, month J (Btu/hr)

Output

Name	Description
PRNTA1(1,IMON)	Total heat energy (Btu)
PRNTA1(2,IMON)	Total electrical energy (Btu)
PRNTA1(3,IMON)	Cooling electrical energy (Btu)
PRNTA1(4,IMON)	Recovered energy (Btu)
PRNTA1(5,IMON)	Wasted recovered energy (Btu)
PRNTA1(6,IMON)	Heat energy input for cooling (Btu)
PRNTA1(7,IMON)	Electrical energy input for cooling (Btu)
PRNTA1(8,IMON)	Energy input for heating (Btu)
PRNTA1(9,IMON)	Energy input for electricity (Btu)
PRNTA1(10,IMON)	Total fuel energy input (Btu)
PRNTA1(11,IMON)	Total energy input (Btu)
PRNTA1(12,IMON)	Average plant efficiency
ENUSE (IMON,I)	Monthly energy used for utility index I (Btu)
ENPEAK(13,I)	Monthly energy peak use rate for utility index I (Btu/hr)

Calculation Procedure

1. If monthly total fuel energy input required for plant and utility is greater than zero,

   AVPTEFF = (EHEATTS+EELECTS)/ ENTOTS,

   i.e., total monthly heat energy produced and electrical energy produced divided by total fuel energy input.

   Otherwise AVPTEFF = 0.

2. PRNTA1(I,IMON) I = 1, 11 are assigned the values of the first 11 input variables, each divided by 1×10^{-9}.

   PRNTA1(12,IMON) = AVPTEFF * 100

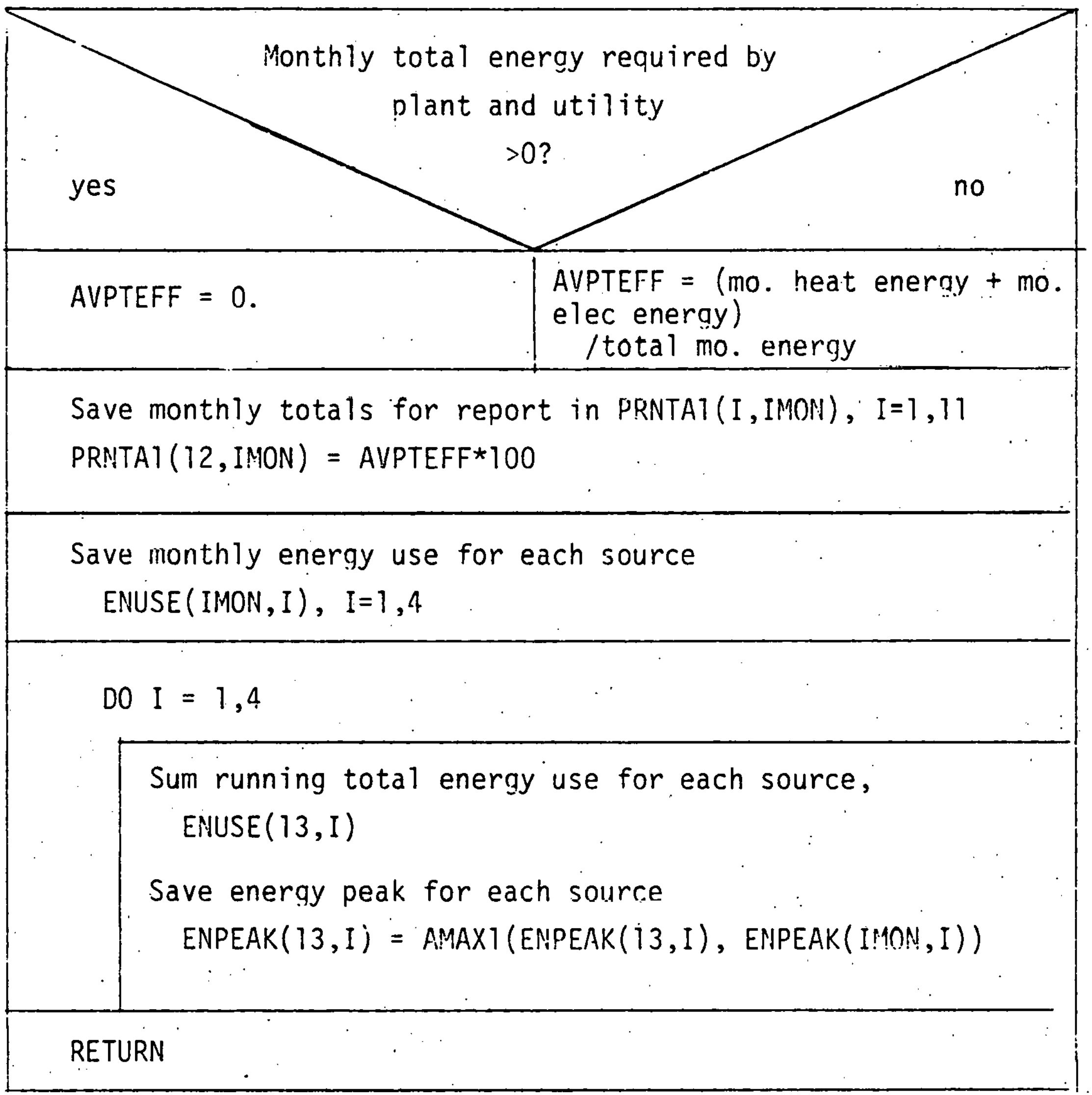
3. ENUSE((IMON,I),I = 1,4) equal respectively, EUTS, EFUELDS, EFUELGS, EFUELBS.

   The yearly energy use sums are increased by the ENUSE for this month, and monthly peak energy use is changed if this month's peak is greater than the previously established peak.

ASHRAE Verification

Does not apply.

STATSM Flow Chart



```
Monthly total energy required by
plant and utility
>0?
yes                                         no
AVPTEFF = 0.            AVPTEFF = (mo. heat energy + mo.
                        elec energy)
                          /total mo. energy
Save monthly totals for report in PRNTA1(I,IMON), I=1,11
PRNTA1(12,IMON) = AVPTEFF*100
Save monthly energy use for each source
  ENUSE(IMON,I), I=1,4
  DO I = 1,4
      Sum running total energy use for each source,
        ENUSE(13,I)
      Save energy peak for each source
        ENPEAK(13,I) = AMAX1(ENPEAK(13,I), ENPEAK(IMON,I))
RETURN
```

ee. Subroutine STMTUR.

SUBROUTINE STMTUR (EELEC, PEXSTM, RPM)

Description

This subroutine simulates a single-stage condensing steam turbine. Parameters relating to output variables are evaluated as follows:

(1) Based steam rate (lb/hp-hr) = theoretical steam rate * $f_1(x_1,c^1) + f_2(x_2,c^2)$,

where f_1 = a quadratic polynomial in x_1
= slope factor;

x_1 = turbine speed (rpm/1000);

c^1 = a set of three coefficients, each of which are quadratic polynomials in the nominal hp rating of the turbine (HPNOM/1000);

f_2 = a quadratic polynomial in x_2
= "B factor" = intercept;

x_2 = turbine speed (rpm/1000); and

c^2 = a set of three coefficients, the first of which is a quadratic polynomial in the nominal hp rating, the second a linear equation in the nominal hp rating, and the third a constant.

Note that the polynomial coefficients, c^1 and c^2, are constants that are not input by the user.

The theoretical steam rate = (2545 Btu/hp-hr)/(enthalpy of superheated entering steam - enthalpy of exhaust steam).

(2) Superheat correction factor is computed by interpolation using 20 equations that are quadratic curve fits representing manufacturers' catalog data for superheat correction.

(3) Horsepower loss = $c_1 * x_1^{2.42} * x_2^{1.47}$,

where x_1 = turbine speed (rpm/1000),

x_2 = nominal hp rating (HPNOM/1000), and

c_1 = 0.0334.

(4) Full-load steam rate = (base steam rate/superheat correction factor) * (1. + hp loss/nominal hp rating).

(5) Entering steam flow rate at nominal capacity, but adjusted for off-nominal turbine speed = full-load steam rate * nominal hp rating * $(f_1(x_1,c^1) + (f_2(x_1,c^2)*R)$,

where f_1 = a third order polynomial in x_1 = PLB,

f_2 = a third order polynomial in x_1 = PLM,

x_1 = (actual turbine speed)/(nominal turbine speed),

c^1 = a set of four polynomial coefficients,

c^2 = a set of four polynomial coefficients, and

R = part-load ratio = (output power)/(nominal capacity).

Note that the polynomial coefficients, c^1 and c^2, are constants that are not input by the user.

(6) Part-load steam flow rate = entering steam flow rate at nominal capacity (as in (5) above) * $f_1(x_1, c^1)$,

where f_1 = as defined in (1) above;

x_1 = part-load ratio, as in (5) above; and

c^1 = RFSTUR.

Subprograms Calling This Routine

SUBROUTINE OPCOOL

Subprograms Called by This Routine

SUBROUTINE SATUR

Common Blocks

EDATA, EPARS, PDATA, SDATA, STMTUR

Declarations

```
DIMENSION RFSTUR(8)
EQUIVALENCE (RFSTUR, PDATA(1,36))
EQUIVALENCE (PSTMTUR, SDATA(1,32)), (TSTMTUR, SDATA(1,33)), (PEXSTUR,
             SDATA(1,34)), (RPMNOM, SDATA(1,35))
```

Input

Source of Data	Name	Description
OPCOOL	EELEC	Electrical energy output (Btu/hr)
OPCOOL	PEXSTM	Exhaust steam pressure (psig)
OPCOOL	RPM	Operating speed (rpm)
SDATA(1,32)	PSTMTUR	Pressure of entering steam (psig)
SDATA(1,33)	TSTMTUR	Temperature of entering steam (°F)
SDATA(1,34)	PEXSTUR	Nominal exhaust steam pressure (psig)
SDATA(1,35)	RPMNOM	Nominal speed of steam turbine (rpm)
/STMTUR/	HSTMTUR	Specific enthalpy of superheated, high-pressure steam (Btu/hr)
/STMTUR/	SSTMTUR	Specific entropy of superheated high-pressure steam (Btu/°F-lb)

Output

Name	Description
FSTMTUR	Flow rate of steam entering turbine (lb/hr)
TEXSTM	Temperature of exhaust steam (°F)
HEXSTM	Specific enthalpy of exhaust steam (Btu/lb)

Calculation Procedure

1. Initialize output variables (turbine off conditions).

```
FSTMTUR = 0.
TEXSTM = TSTMTUR
HEXSTM = HSTMTUR
```

2. If no electrical energy is required (i.e., EELEC ≤ 0.), skip the following calculations and return. Otherwise, continue.

3. If exhaust steam pressure ≥ entering steam pressure and/or turbine speed ≤ 0 (i.e., turbine is off), skip the following calculations and return. Otherwise, continue.

4. Convert electrical power output from Btu/hr to hp.

```
POWER = EELEC/2545.
```

5. Function SATUR returns saturation temperature as a function of steam pressure.

```
TSAT1 = SATUR(PSMTUR), and
TEXSTM = SATUR(PEXSTM),
```

where TSAT1 = saturation temperature of entering steam (°F).

6. Calculate the enthalpy of the exhaust steam.

```
HEXSTM = 1.0045 * TEXSTM - 32.448 + (TEXSTM+460.) * (SSTMTUR-1.0045*
         ALOG(TEXSTM+460.)+6.2264)
```

This expression was taken directly from the NECAP program (Ref. 3, Vol. II, pp. 5-124).

7. Convert nominal capacity to hp.

```
SIZE = HPNOM = OPCAP(17)/2545.
```

8. Calculate intermediate variables.

```
HPKILO = HPNOM/1000.
RPMKILO = RPM/1000.
HPKSQ = HPKILO * HPKILO
```

9. Calculate base steam rate, BSR (lb/hp-hr).

(a) Determine "B factor" (intercept).

```
B0 = 84.-17. * HPKILO + 1.5625 * HPKSQ
B1 = -19.7 + 1.025 * HPKILO
B2 = 1.4
B = B0 + B1 * RPMKILO + B2 * RPMKILO * RPMKILO
```

(b) Determine slope factor.

```
S0 = 3.88 - 1.1865 * HPKILO + 0.1173 * HPKSQ
S1 = -1.1 + 0.533 * HPKILO - 0.0581 * HPKSQ
S2 = 0.116 - 0.057 * HPKILO + 0.00709 * HPKSQ
SLOPE = S0 + S1 * RPMKILO + S2 * RPMKILO * RPMKILO
```

(c) Calculate theoretical steam rate (lb/hp-hr).

```
TSR = 2545./(HSTMTUR-HEXSTM),
```

(d) Calculate base steam rate.

BSR = SLOPE * TSR + B

This calculation is based on a curve fit of Elliott YR single-stage steam turbine catalog data. It comes directly from the NECAP program (Ref. 3, Vol. II, pp. 5-124).

10. Calculate steam superheat (°F).

SH = TSTMTUR - TSAT1

11. Calculate superheat correction factor, SC.

Interpolate using a series of 20 equations based on curve fits of Elliott YR single-stage steam turbine catalog data. See program listing for these equations.

This formulation comes directly from the NECAP program (Ref. 3, Vol. II, pp. 5-125).

12. Calculate hp loss term.

HPLOSS = 0.0334 * (RPMKILO**2.42) * (HPKILO**1.47)

This expression is based on a curve fit of Elliott YT single-stage steam turbine catalog data for a condensing turbine at an exhaust pressure of 2 psia. It comes directly from the NECAP program (Ref. 3, Vol. II, pp. 5-124).

13. Calculate full-load steam rate (lb/hp-hr).

FLSR = (BSR/SC) * (1.+HPLOSS/HPNOM)

This expression comes directly from NECAP (Ref. 3, Vol. II, pp. 5-125).

14. Calculate the actual-to-nominal turbine speed ratio, RPMR, and intermediate variables.

RPMR = RPM/RPMNOM
RPMRSQ = RPMR * RPMR
RPMRCU = RPMRSQ * RPMR

15. Determine off-nominal turbine speed factors.

PLB = 0.09163 + 0.0404 * RPMR - 0.0076 * RPMRSQ + 0.0003167 * RPMRCU
PLM = 5.219 - 14.627 * RPMR + 16.62 * RPMRSQ - 6.2524 * RPMRCU

Note, however, that in Cal-ERDA the turbines are assumed to operate only at nominal speed.

16. Calculate entering steam flow rate (lb/hr) at nominal capacity, adjusted for off-nominal speed.

FSTMTUR = FLSR * HPNOM * (PLB + PLM * POWER/HPNOM)

17. Calculate part-load entering steam flow rate (lb/hr).

FSTMTUR = FSTMTUR * [C(1)+C(2)*R+C(3)*R*R],

where R = part-load ratio = POWER/HPNOM, and
C = RFSTUR.

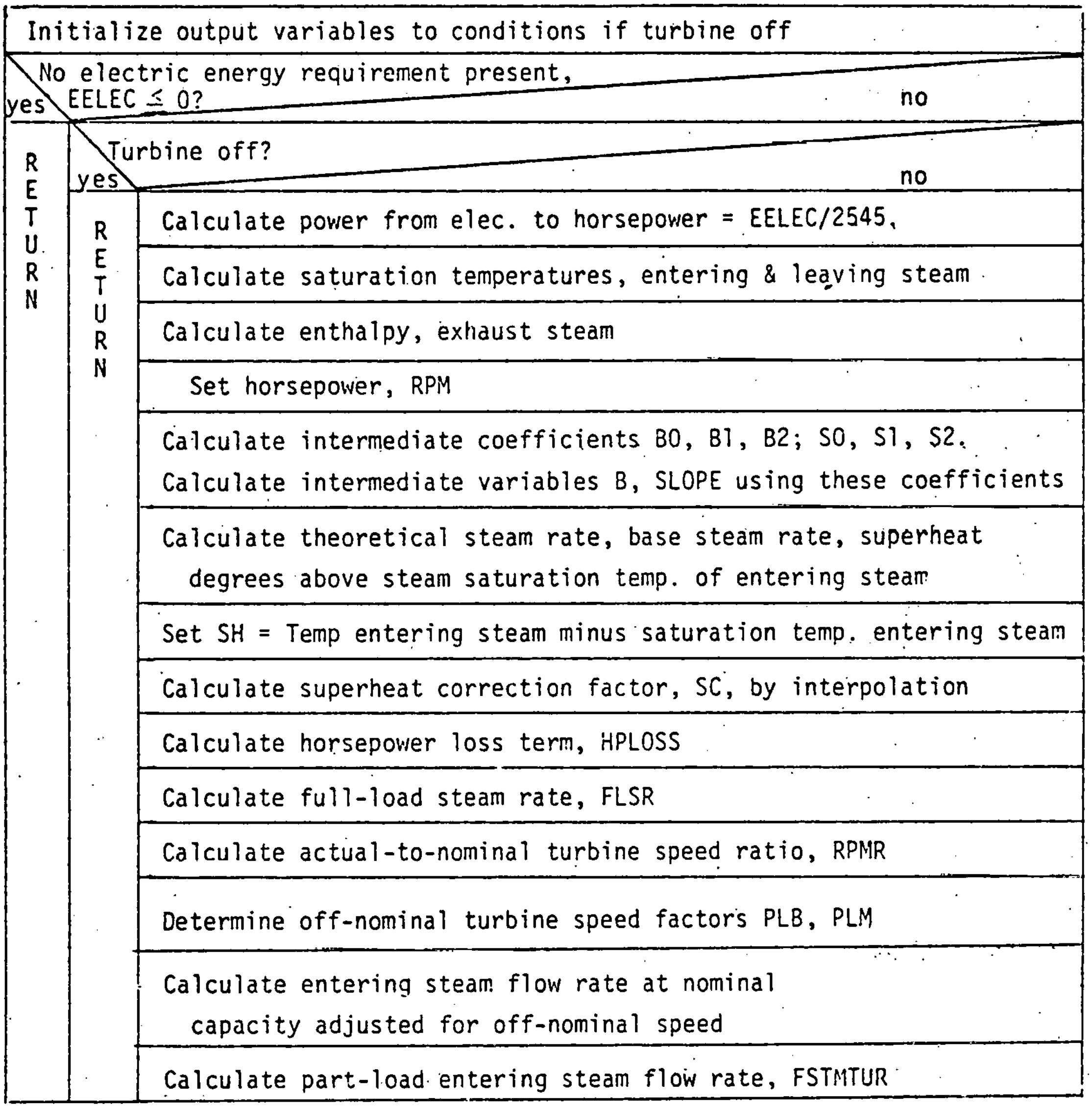
ASHRAE Verification

This routine is considerably more sophisticated than that recommended by ASHRAE for engines (Ref. 2). Reference 2 expresses part-load consumption rate as a linear function of part-load ratio. Cal-ERDA expresses this relationship more accurately using a quadratic equation in part-load ratio (defined with respect to nominal capacity), but also takes into account the effect of off-nominal speed, larger turbine size, and a greater amount of superheat in the entering steam.

The recoverable heat rate here applies to the energy in the exhaust steam. Although the steam turbine waste heat term is not calculated in STMTUR, it is calculated in HEATREC as the energy content of the exhaust steam. This recoverable heat rate is expressed in Ref. 2 as a quadratic in the part-load ratio. In contrast, the Cal-ERDA expression involves the user-specified parameter of the ratio of exhaust steam to entering steam flow rate used in an energy balance on the exhaust steam condensing process.

Relationships for the dependence of the steam rate on turbine speed, nominal capacity, superheat correction factor, horsepower loss term, and off-nominal turbine speed factor are not presented by ASHRAE. The Cal-ERDA relationships for these were taken directly from the NECAP program.

STMTUR Flow Chart



```
Initialize output variables to conditions if turbine off
No electric energy requirement present,
EELEC ≤ 0?
  yes: RETURN
  no:
    Turbine off?
      yes: RETURN
      no:
        Calculate power from elec. to horsepower = EELEC/2545,
        Calculate saturation temperatures, entering & leaving steam
        Calculate enthalpy, exhaust steam
          Set horsepower, RPM
        Calculate intermediate coefficients B0, B1, B2; S0, S1, S2,
        Calculate intermediate variables B, SLOPE using these coefficients
        Calculate theoretical steam rate, base steam rate, superheat
          degrees above steam saturation temp. of entering steam
        Set SH = Temp entering steam minus saturation temp. entering steam
        Calculate superheat correction factor, SC, by interpolation
        Calculate horsepower loss term, HPLOSS
        Calculate full-load steam rate, FLSR
        Calculate actual-to-nominal turbine speed ratio, RPMR
        Determine off-nominal turbine speed factors PLB, PLM
        Calculate entering steam flow rate at nominal
          capacity adjusted for off-nominal speed
        Calculate part-load entering steam flow rate, FSTMTUR
```

ff. Subroutine STMUSE.

Description

This is a subroutine to calculate total steam consumption of steam users (including steam turbine, but excluding space heating and absorption chiller). Each steam user is characterized by:

(1) Ratio of return water to steam flow input, and
(2) Return water temperature.

The energy consumed by the steam user equals the energy of the portion of the steam that is not returned, plus energy loss of the portion that is returned to the boiler.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

None

Common Blocks

EPARS, SDATA, STM, STMTUR

Declarations

EQUIVALENCE (HSTEAM, SDATA(1,1)), (RWSTUR, SDATA(1,36))

Input

Source of Data	Name	Description
CCBTEPS	ESTUSE	Total steam energy load of steam users (Btu/hr)
CCBTEPS	RWTR	Ratio of return water to steam flow
/STMTUR/	FSTMTUR	Flow of steam entering steam turbine (Btu/hr)
/STMTUR/	HSTMTUR	Enthalphy of superheated high-pressure steam (Btu/hr)
/STMTUR/	TEXSTM	Temperature of exhaust steam (°F)
SDATA(1,1)	HSTEAM	Steam enthalpy (Btu/hr)
SDATA(1,36)	RWSTUR	Ratio of exhaust steam to steam turbine entering flow
/EPARS/	TOTCAP(4)	Total capacity of boiler (Btu/hr)
/EPARS/	OPCAP(17)	Operating capacity of steam turbine (Btu/hr)

Output

Name	Description
ESTMS	Total steam energy (Btu/hr)
FSTMS	Total steam flow (lb/hr)
EWTRM	Energy of return water mixture (Btu/hr)
FWTRM	Flow of return water mixture (lb/hr)

Calculation Procedure

1. Output variables are initialized to zero. If no steam boiler is in the system (i.e., TOTCAP(4) $\leq$ 0), skip the calculation and return.

2. If no steam turbine is operating, the calculation of steam consumption of the turbine is skipped, and calculation continues at Step 3. Otherwise the following calculations are made:

   a. total steam flow = flow of steam entering steam turbine,

      FSTMS = FSTMTUR;

   b. total steam energy = entering steam flow times enthalpy of superheated steam,

      ESTMS = FSTMTUR * HSTMTUR;

   c. return condensate flow = ratio of exhaust steam to entering steam flow, times entering steam flow,

      FWTRM = RWSTUR * FSTMTUR; and

   d. return condensate energy = return condensate flow times, the quantity, temperature of exhaust steam minus 32,

      EWTRM = FWTRM * (TEXSTM-32.).

3. The following temporary variables are calculated.

   Flow to steam users, FSTM = ESTUSE/(HSTEAM-RWTR*180.)(i.e., total steam load/(steam enthalpy-enthalpy of return condensate), where enthalpy of condensate = return condensate to steam flow ratio * 180°F).

   Energy of steam to users = flow to steam users * steam enthalpy.

   ESTM = FSTM * HSTEAM

   Return condensate flow = return water to steam flow ratio * steam flow.

   FWTR = RWTR * FSTM

   Return condensate energy = return condensate flow * 180°F.

   EWTR = FWTR * 180.

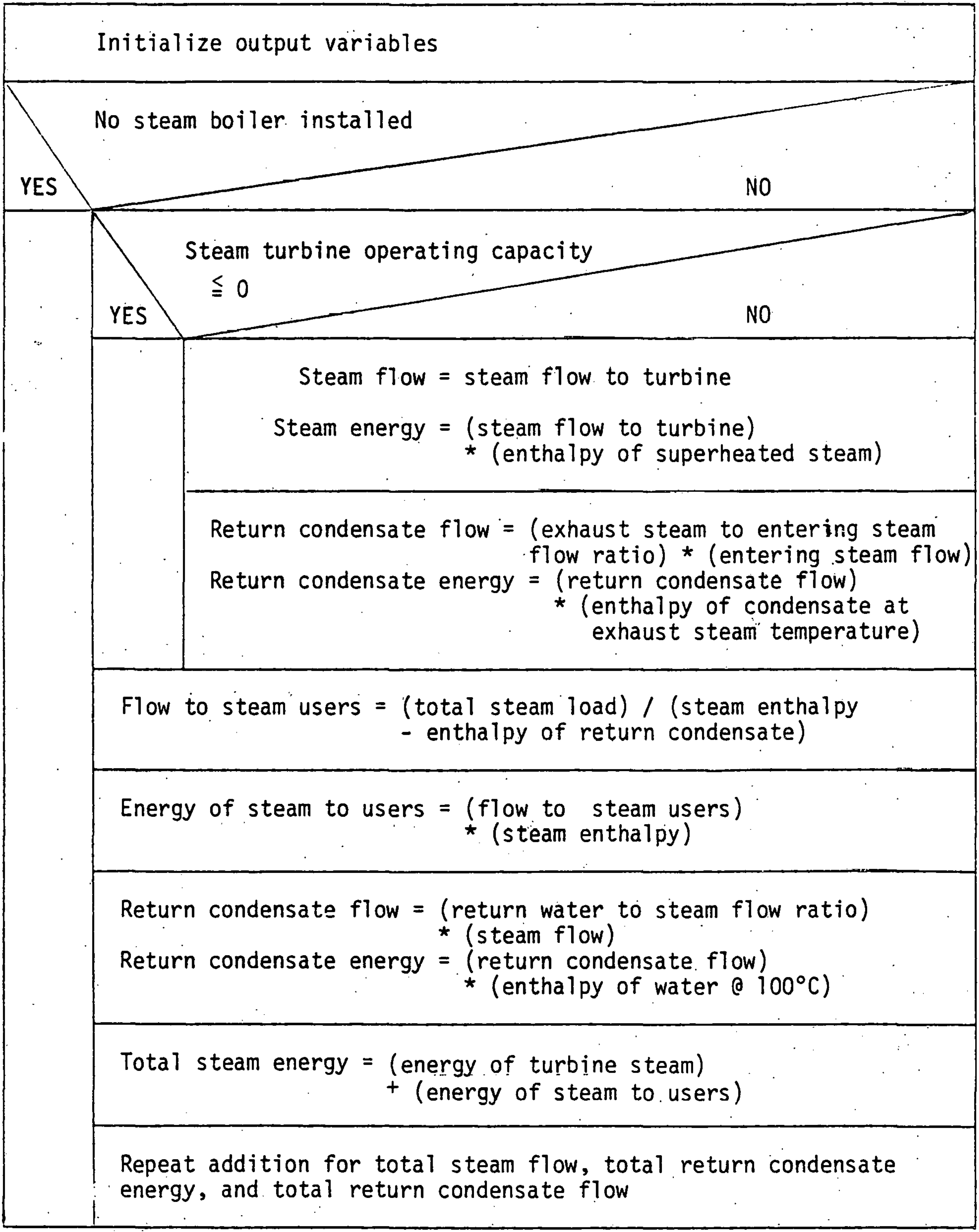
4. Output variables are calculated as sums of the values from Steps 2 and 3.

   Total steam energy, ESTMS = ESTMS + ESTM.
   Total steam flow, FSTMS = FSTMS + FSTM.
   Total return condensate energy, EWTRM = EWTRM + EWTR.
   Total return condensate flow, FWTRM = FWTRM + FWTR.

ASHRAE Verification

There is no applicable ASHRAE algorithm for the calculations performed in subroutine STMUSE.

STMUSE Flow Chart



```
Initialize output variables
No steam boiler installed
YES
NO
    Steam turbine operating capacity
      ≦ 0
    YES
    NO
        Steam flow = steam flow to turbine
        Steam energy = (steam flow to turbine)
                       * (enthalpy of superheated steam)
        Return condensate flow = (exhaust steam to entering steam
                                 flow ratio) * (entering steam flow)
        Return condensate energy = (return condensate flow)
                                   * (enthalpy of condensate at
                                      exhaust steam temperature)
    Flow to steam users = (total steam load) / (steam enthalpy
                          - enthalpy of return condensate)
    Energy of steam to users = (flow to  steam users)
                               * (steam enthalpy)
    Return condensate flow = (return water to steam flow ratio)
                             * (steam flow)
    Return condensate energy = (return condensate flow)
                               * (enthalpy of water @ 100°C)
    Total steam energy = (energy of turbine steam)
                         + (energy of steam to users)
    Repeat addition for total steam flow, total return condensate
    energy, and total return condensate flow
```

gg. Subroutine TOWER.

SUBROUTINE TOWER (ETOWER, EWASTED, TENT, EELEC, TCOLD, PLOADT)

Description

This subroutine simulates a cooling tower. Two generic types of tower, conventional and ceramic, are included. The towers can be operated under variable water rate, variable range, or fixed water rate, fixed range conditions.

The tower load (energy to be dissipated), in conjunction with the condenser water temperature and flow rate, and ambient wet-bulb temperature, are used to calculate the number of cells in operation and the tower outlet water temperature. The number of cells in operation, in turn, determines the electrical requirements for the fan motors, which run at either full or half speed. The simulation involves an iterative process using a 2°F increment. At each iteration, the tower rating factor is recalculated by calling the function RFACT.

A lower bound on the tower outlet water temperature is taken to be 2°F above the ambient wet-bulb temperature.

If the tower capacity is not specified in the input, i.e., TOTCAP (ITOWR) = 0, then the special parameter PELTWR is used to approximate the average electrical consumption of the cooling tower. Otherwise, the iterative process described above is used.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

FUNCTION RFACT

Common Blocks

EDATA, EPARS, LDISTD, SDATA, TOWERD, WEATHR

Declarations

```
EQUIVALENCE (TOWOPR, SDATA(1,7)), (TTOWER, SDATA(1,12)), (RWCA, SDATA(1,
            25)), (RWCC, SDATA(1,26)), (RWCDB, SDATA(1,27)), (PELTWR,
            SDATA(1,6))
```

Input

Source of Data	Name	Description
CCBTEPS	ETOWER	Tower cooling load (Btu/hr)
CCBTEPS	EWASTED	Wasted recoverable heat not used the previous hour (added to tower load) (Btu/hr)
CCBTEPS	TENT	Tower inlet water temperature (°F)
/TOWERD/	KT	Number of cells
/TOWERD/	PNTK	Fan motor power for one cell (Btu/hr)
/TOWERD/	CNT	Cooling capacity at 90-80-70 point (Btu/hr)
/TOWERD/	CNTU	Cooling capacity of one tower cell, in TU = CNT * 5000
/TOWERD/	CNTH	Cooling capacity of one cell at half speed (Btu/hr)
/TOWERD/	CNTUH	Cooling capacity of one cell at half speed, in TU = CNTH * 5000
/EPARS/	IABSOR	Absorption chiller type, same as in ABSREF
/EPARS/	ICOMPR	Compression chiller type, same as in COMREF
/EPARS/	ITOWR	Cooling tower type: 14 for conventional cooling tower 15 for ceramic cooling tower
/EPARS/	OPCAP(IABSOR)	Operating capacity of absorption chiller (Btu/hr)
/EPARS/	OPCAP(ICOMPR)	Operating capacity of compression chiller (Btu/hr)
/EPARS/	OPCAP(13)	Operating capacity of double-bundle chiller (Btu/hr)
/EPARS/	OPCAP(ITOWR)	Operating capacity of cooling tower (Btu/hr)
/WEATHR/	TWET	Wet-bulb temperature (°F)
/LDISTD/	NOPR(ITOWR)	Number of tower units in operation
/LDISTD/	IOPR(1,ITOWR)	Number of tower cells in operation
SDATA(1,7)	TOWOPR	Type of tower operation: 1 for variable water rate, variable temperature range 2 for fixed water rate, variable temperature range
SDATA(1,12)	TTOWER	Lower bound for temperature of tower outlet water (°F)

SDATA(1,25)	RWCA	Tower water flow rate/absorption chiller capacity
SDATA(1,26)	RWCC	Tower water flow rate/compression chiller capacity
SDATA(1,27)	RWCDB	Tower water flow rate/double-bundle chiller capacity
SDATA(1,6)	PELTWR	Electrical input to tower/tower cooling load

Output

Name	Description
EELEC	Required electric energy (Btu/hr)
TCOLD	Tower outlet water temperature (°F)
PLOADT	Amount of load tower is handling (Btu/hr)
IOPR(1,ITOWR) (NCELL)	Number of tower cells in operation

Calculation Procedure

1. Initialize output variables (conditions when tower is off).

   TCOLD = TWET
   NCELL = 0.
   OPCAP(ITOWER) = PLOADT = EELEC = 0.

2. If no tower load is present (i.e., ETOWER $\le$ 0), skip the following calculations and return. Otherwise, continue.

3. Set tower outlet water temperature to the ambient wet-bulb temperature + 2°F, subject to a lower bound, TTOWER (a special parameter).

   TCOLD = AMAX1(TTOWER, TWET + 2)

4. If tower capacity is not specified in input (i.e., TOTCAP(ITOWR) = 0), calculate average electrical consumption and return.

   EELEC = ETOWER * PELTWR,

   where PELTWR = (electrical input)/(tower load) = a special parameter.

   Skip to 25.

5. Begin iteration; initialize electrical power requirement.

   ELEC = 0,

   where ELEC is in kW units.

6. Calculate variable water rate (if TOWOPR $\le$ 1) for variable temperature range.

   RANGE = TENT-TCOLD
   = temperature drop through tower,

   GPM = ETOWER/(RANGE*500.)
   = tower water flow rate (gpm),

   where 500. = 60(min/hr) * 8.34 (lb/gal) * 1(Btu/lb°F) converts the flow rate units to gpm.

   Skip to 8.

7. Calculate fixed water rate (if TOWOPR > 1) for variable temperature range.

   GPM = tower water flow rate (gpm)
   = (OPCAP(IABSOR) * RWCA + OPCAP(ICOMPR) * RWCC + OPCAP(13) * RWCDB
   + 2. * EWASTED)/ 12000.,

   where 12000 = Btu/hr-ton. Thus the water flow rate is set at a value that accommodates the waste heat from all cooling equipment and the wasted recoverable heat not used the previous hour because of insufficient demand and not storable.

   RANGE = ETOWER/(GPM*500.)
   = temperature drop through tower.

8. Calculate approach to wet-bulb temperature.

   APPR = TCOLD - TWET

9. If approach is < 5°F, skip to 18. Otherwise, calculate tower rating factor using function RFACT.

   RF = RFACT(RANGE, APPR, TWET)

   The rating factor is the number of tower units/gpm of tower water flow rate.

10. Compute rated area in tower units.

    RATEDA = RF * GPM.

11. Set number of tower cells operating = 1.

    NCELL = 1

12. Initialize intermediate electrical power requirement in kW.

    ELEC1 = ELEC

13. If CNTUH, the cooling capacity of one cell running at half speed, > 0,

    ELEC = ELEC1 + 0.5 * PNTK * RATEDA/CNTUH,

    where 0.5 * PNTK = 1/2 of 1 cell's fan motor power.

14. If the capacity required (RATEDA) ≤ capacity provided (CNTUH) with NCELL cells on at half speed, skip to 20. Otherwise, calculate electrical power requirement with NCELL cells on at full speed (capacity).

    ELEC = ELEC1 + PNTK * RATEDA/CNTU,

    where PNTK = 1 cell's fan motor power.

15. If the capacity required (RATEDA) ≤ capacity provided (CNTU) with NCELL cells on at full speed, skip to 20. Otherwise:

    ELEC = ELEC1 + PNTK,

    where PNTK = 1 cell's fan motor power.

16. If all available cells are on (i.e., NCELL = KT), skip to 18.
    Otherwise, decrease the capacity required (RATEDA) by the capacity provided by NCELL cells (CNTU).

    RATEDA = RATEDA - CNTU

17. Since all cells are not operating, turn on another cell and try to satisfy load.

    NCELL = NCELL + 1

    Go back to 12.

18. Load cannot be met under these conditions. Therefore, the tower water outlet temperature is increased by 2°F, subject to an upper bound that TCOLD $\leq$ TENT.

    TCOLD = AMIN1(TENT, TCOLD+2)

19. If TCOLD < TENT, iterate and go back to 5. Otherwise, continue.

20. Load is satisfied and tower operating capacity is calculated.

    OPCAP(ITOWR) = NCELL * CNT,

    where CNT = CNTU/5000. = cooling capacity of 1 cell at the 90-80-70 point (TENT = 90°F, TCOLD = 80°F, TWET = 70°F).

21. Set cooling load being handled by tower equal to the tower operating capacity, subject to upper bound ETOWER.

    PLOADT = AMIN1(OPCAP(ITOWR), ETOWER),

    where ETOWER = tower cooling load.

22. Convert the tower electrical power requirement from kW to Btu/hr.

    EELEC = ELEC * 3412.

23. If fixed water rate tower (i.e., TOWOPR > 1), skip to 24. Otherwise, if RANGE > 0, recalculate required electrical power at variable water rate.

    EELEC = EELEC + PELTWR * ETOWER * 10./RANGE

    Skip to Step 25.

24. Recalculate required electrical power at fixed water rate.

    EELEC = EELEC + PELTWR * CNT * NCELL

25. Calculate the number of tower units in operation.

    NOPR(ITOWR) = 1.

    If no cells are in operation, NOPR(ITOWR) = 0.

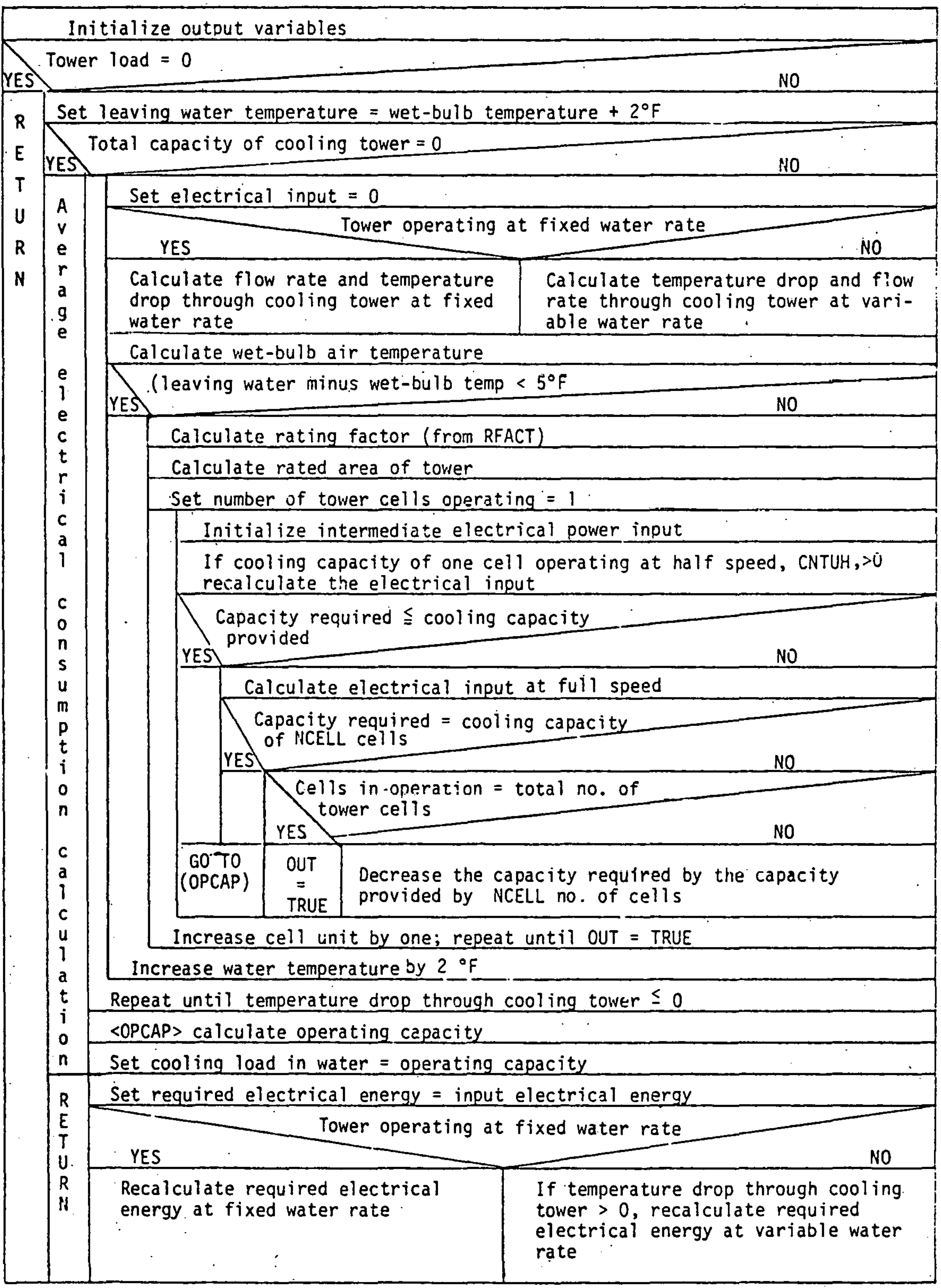
26. Calculate the number of tower cells in operation.

    IOPR(1,ITOWR) = NCELL

ASHRAE Verification

The Cal-ERDA cooling tower simulation model differs completely from the ASHRAE model (Ref. 2). ASHRAE does not recommend the iterative approach used here, but rather expresses the tower outlet water temperature simply as the product of quadratic polynomials in the ambient wet-bulb and the tower inlet water temperature. The Cal-ERDA model also takes into account the dependence on the wet-bulb and tower inlet water temperature, but does so using the tower rating factor function (see RFACT write-up).

TOWER Flow Chart



```
Initialize output variables
Tower load = 0
  YES:
    RETURN
  NO:
    Set leaving water temperature = wet-bulb temperature + 2°F
    Total capacity of cooling tower = 0
      YES:
        Average electrical consumption calculation
      NO:
        Set electrical input = 0
        Tower operating at fixed water rate
          YES:
            Calculate flow rate and temperature drop through cooling tower at fixed water rate
          NO:
            Calculate temperature drop and flow rate through cooling tower at variable water rate
        Calculate wet-bulb air temperature
        (leaving water minus wet-bulb temp < 5°F
          YES:
            Calculate rating factor (from RFACT)
            Calculate rated area of tower
            Set number of tower cells operating = 1
              Initialize intermediate electrical power input
              If cooling capacity of one cell operating at half speed, CNTUH,>0 recalculate the electrical input
              Capacity required ≦ cooling capacity provided
                YES:
                  GO TO (OPCAP)
                NO:
                  Calculate electrical input at full speed
                  Capacity required = cooling capacity of NCELL cells
                    YES:
                    NO:
                      Cells in operation = total no. of tower cells
                        YES:
                          OUT = TRUE
                        NO:
                          Decrease the capacity required by the capacity provided by NCELL no. of cells
            Increase cell unit by one; repeat until OUT = TRUE
          NO:
        Increase water temperature by 2 °F
      Repeat until temperature drop through cooling tower ≤ 0
      <OPCAP> calculate operating capacity
      Set cooling load in water = operating capacity
      RETURN
      Set required electrical energy = input electrical energy
      Tower operating at fixed water rate
        YES:
          Recalculate required electrical energy at fixed water rate
        NO:
          If temperature drop through cooling tower > 0, recalculate required electrical energy at variable water rate
```

hh. Function WETBULB.

FUNCTION WETBULB(TAIR, HR)

Description

WETBULB calculates wet-bulb temperature at sea level with positive enthalpy, H, given dry-bulb temperature and humidity ratio.

Subprograms Calling This Routine

CCBTEPS

Subprograms Called by This Routine

None

Common Blocks

None

Declarations

None

Input

Source of Data	Name	Description
CCBTEPS	TAIR	Dry-bulb temperature (°F)
CCBTEPS	HR	Humidity ratio

Output

Name	Description
WETBULB	Wet-bulb temperature (°F)

Calculation Procedure

1. If dry-bulb temperature, TAIR, is $\leq$ 32°F, WETBULB = TAIR.
2. Otherwise, the mixture enthalpy is calculated as:
   H = 0.24 * TAIR + (1061.+0.444*TAIR) * HR,
   and Y is taken as logH.
3. If H is greater than 11.758,
   $WETBULB = 30.9185 - 39.682Y + 20.5841(Y^2) - (1.758(Y^3)$.
   If H is less than or equal to 11.758,
   $WETBULB = 0.604 + 3.4841Y + 1.360Y^2 + 0.9731Y^3$.

ASHRAE Verification

This routine is taken directly from ASHRAE documentation (Ref. 1, p. 163), with the following exception. Whereas Ref. 1 recommends the solution of an iterative equation for the wet-bulb temperature if the dry-bulb temperature $\leq$ 32°F (i.e., if enthalpy < 0.), Cal-ERDA sets the wet-bulb temperature equal to the dry-bulb temperature for TAIR $\leq$ 32°F.

F. Solar Simulator

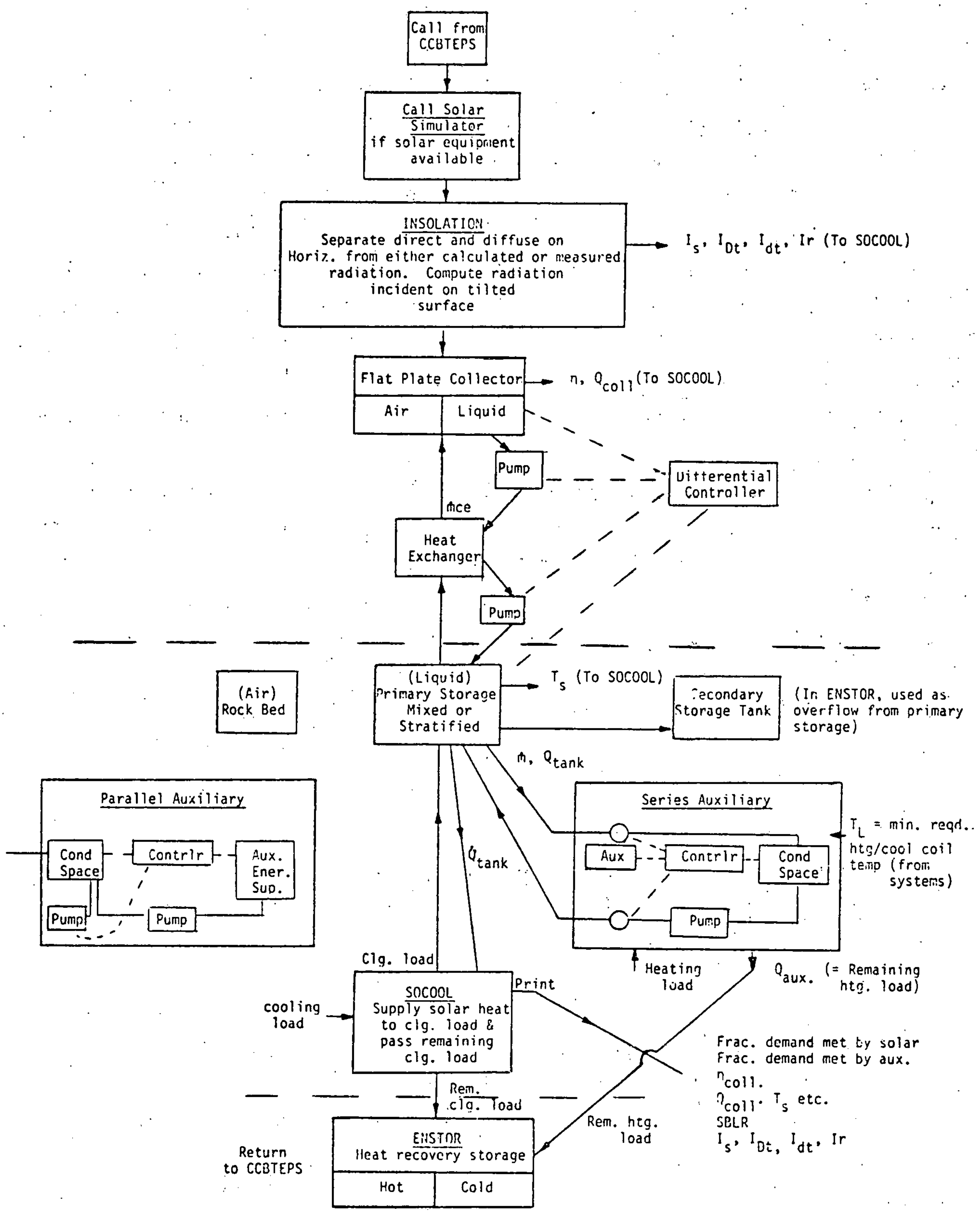
1. Introduction. The solar system simulation package of Cal-ERDA is a multilevel design tool for analysis of both liquid and air active solar energy systems. It was developed by LASL. It uses the same basic component connection philosophy as the University of Wisconsin TRNSYS program (Ref. 15) (i.e., execution time linking of components by their inputs and outputs). Several TRNSYS component models are used, with varying degrees of modification.

Four preconnected systems are presently available; a fully user-designed approach will be implemented in the near future. While only solar heating is simulated in the solar simulator, solar cooling is modeled by the subroutine SOCOOL in the PLANT Program. The Solar Simulator Functional Chart shows the simulation structure.

The backbone of the solar simulator is the modeling of equipment on a component basis. Each component is a separate subroutine or several combined in one subroutine; each component requires inputs and parameters (values that define the component) and produces outputs. Standard systems, composed of preconnected components, are available to the user so that no user connection is required. Each of the liquid and air systems has two levels of modeling complexity. The complexity pertains to the detail involved, which implies a tradeoff between accuracy and both user input and run time.

There is now only one insolation model available, the Boeing Cloud Cover Modifier method (the same as used in the LOADS Program, see Ref. 10). This method uses calculated clear-day insolation modified for cloud cover. Future versions will include hourly horizontal surface total insolation read from a SOLMET tape. These tapes are provided by the Environmental Data Service, National Oceanic and Atmospheric Administration, U. S. Department of Commerce.

Solar Simulator Functional Chart


Call from CCBTEPS
Call Solar Simulator if solar equipment available
INSOLATION Separate direct and diffuse on Horiz. from either calculated or measured radiation. Compute radiation incident on tilted surface
I_s, I_{Dt}, I_{dt}, Ir (To SOCOOL)
Flat Plate Collector
Air
Liquid
n, Q_{coll} (To SOCOOL)
Pump
Differential Controller
ṁce
Heat Exchanger
Pump
(Air) Rock Bed
(Liquid) Primary Storage Mixed or Stratified
T_s (To SOCOOL)
Secondary Storage Tank
(In ENSTOR, used as overflow from primary storage)
ṁ, Q_{tank}
Parallel Auxiliary
Cond Space
Contrlr
Aux. Ener. Sup.
Pump
Pump
Q_{tank}
Series Auxiliary
Aux
Contrlr
Cond Space
Pump
T_L = min. reqd. htg/cool coil temp (from systems)
Clg. load
Heating load
$Q_{aux.}$
(= Remaining htg. load)
cooling load
SOCOOL Supply solar heat to clg. load &amp; pass remaining clg. load
Print
Frac. demand met by solar
Frac. demand met by aux.
$\eta_{coll.}$
Q_{coll}. T_s etc.
SBLR
I_s, I_{Dt}, I_{dt}, Ir
Rem. clg. load
Rem. htg. load
Return to CCBTEPS
ENSTOR Heat recovery storage
Hot
Cold

2. Liquid Systems. The liquid systems consist of a flat-plate collector, a differential controller, two pumps, a heat exchanger, and a storage tank (see Fig. V.10). Solar-heated water is first applied to the preheat coil, second the heating coil, and then the reheat coils. Each of these respective loads is passed from the SYSTEMS Program in the energy load file. If a load cannot be met entirely by solar, auxiliary energy is used either as a boost or as the sole energy source, depending upon the control mode.

The main difference between the simple systems (Liquid System No. 1 and Air System No. 1) and their respective complex systems (Liquid System No. 2 and Air System No. 2) is the level of sophistication and detail in the models. For example, the simple systems use constant collector loss coefficients, while the complex systems calculate an hourly value of collector loss coefficient based upon number of glazings, collector tilt, wind speed, absorber emittance, etc.

In addition to the actual equipment simulation, the selection of a pre-connected system automatically performs the required input/output connections. The user must specify only those component definition values that do not have default values.

a. Liquid System No. 1. This system uses the following component subroutines:

(1)	TYPE28	weather reader
(2)	TYPE16	insolation (MODE 1)
(3)	TYPE29	auxiliary controller
(4)	TYPE21	collector, pumps, heat exchanger, tank, differential controller
(5)	TYPE25	report (MODE 2)

For details see the subroutine descriptions (Sec. 4).

b. Liquid System No. 2. This system uses the following component subroutines:

(1)	TYPE28	weather reader
(2)	TYPE16	insolation (MODE 1)
(3)	TYPE4	tank
(4)	TYPE1	collector (MODE 4)
(5)	TYPE2	differential controller
(6)	TYPE3	pump

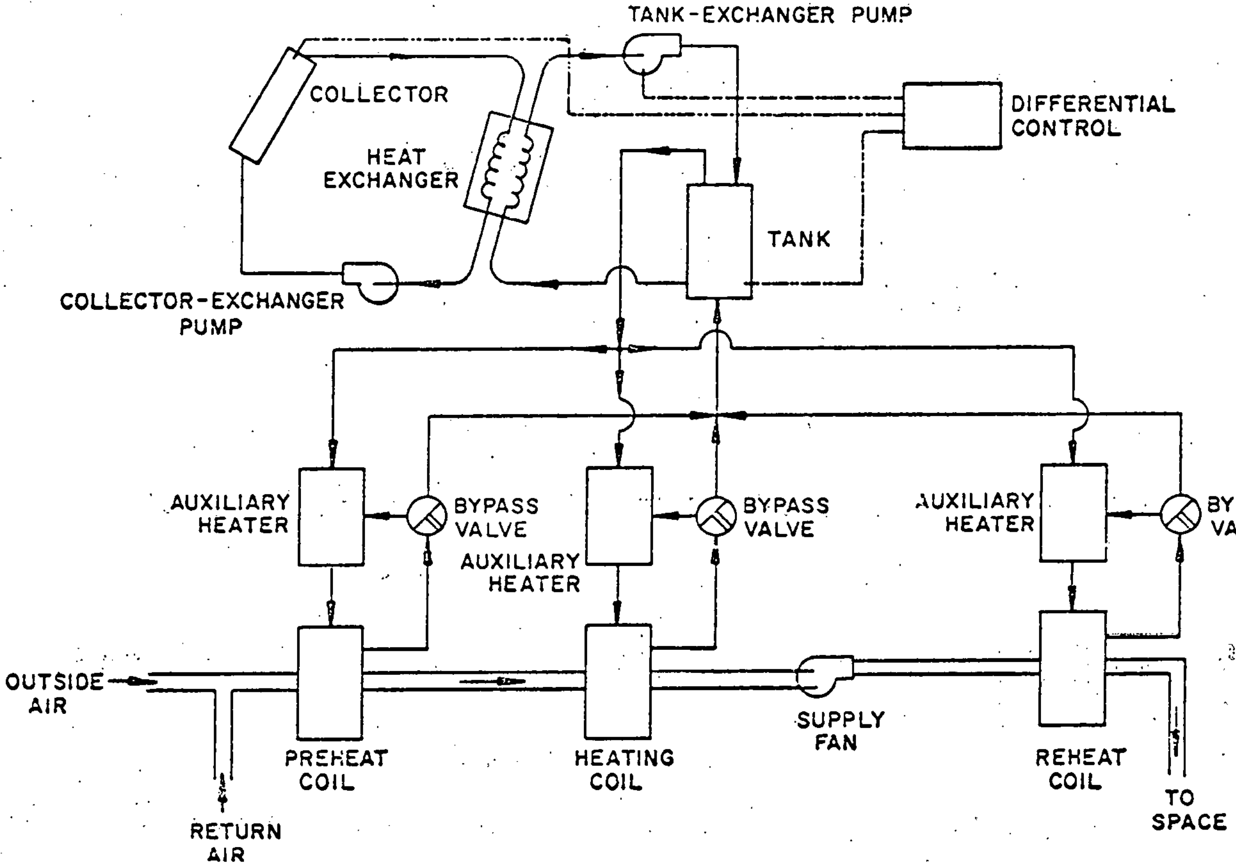


Fig. V.10. Liquid solar energy system.

(7) TYPE5 heat exchanger
(8) TYPE31 auxiliary controller
(9) TYPE25 report (MODE 2)

For details see the subroutine descriptions (Sec. 4).

3. Air Systems. The air heating systems consist of a flat-plate controller, a differential controller, a collector fan, a rock bed, and an auxiliary control model. A series of dampers controls the amount of mixed return and outside air to be diverted through the rock bed to meet the preheat load. Auxiliary energy is added only when the rock bed is unable to supply sufficient heating. Note that since the supply fan is part of the SYSTEMS Program, the rock bed pressure drop must be added to the supply fan static pressure in the SYSTEMS input when a system with a rock bed is specified. See Fig. V.11.

a. Air System No. 1. This system uses the following component subroutines:

(1) TYPE28 weather reader
(2) TYPE16 insolation (MODE 1)
(3) TYPE31 auxiliary controller
(4) TYPE22 collector, collector fan, differential controller, rock bed

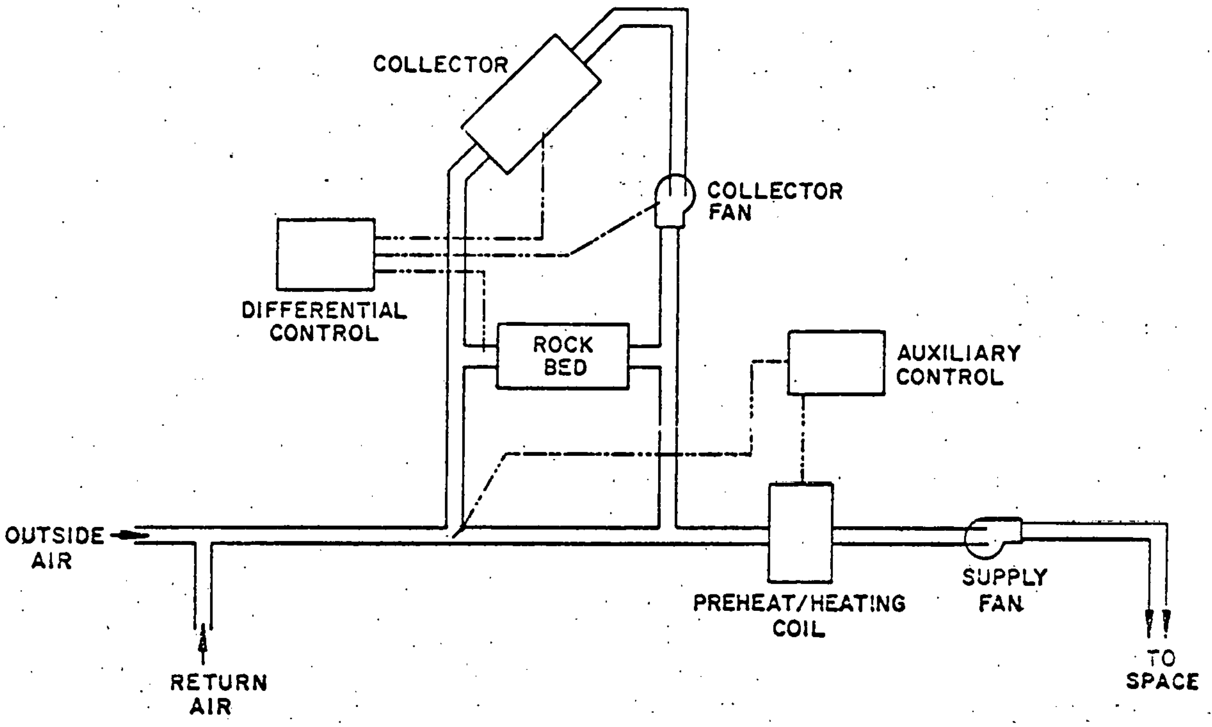


Fig. V.11. Air solar energy system.

b. Air System No. 2. This system uses the following component routines:

(1) TYPE28 weather reader
(2) TYPE16 insolation (MODE 1)
(3) TYPE10 rock bed
(4) TYPE1 collector (MODE 4)
(5) TYPE2 differential controller
(6) TYPE3 fan
(7) TYPE31 auxiliary controller
(8) TYPE25 report (MODE 2)

For details see the subroutine descriptions (Sec. 4).

4. Description by Subroutine.

These subroutine descriptions cover only the calculational sequences. A description of the inputs, outputs, and constants for each routine is given in the Users Manual.

a. Subroutine CBS.

Description

Subroutine CBS is the solar simulator senior executive routine. The main tasks of CBS are (1) to initialize component routines, (2) to sequence the three categories of components, and (3) to recall iterative components until convergence is attained.

Calculation Procedure

1. If this is not an initialization call, go to Step 6.
2. Call RDCOMP to input the component data from the input processor.
3. Set index limits for beginning-of-step, iterative, and end-of-step routines.
4. Call JEXEC to initialize all routines.
5. Exit.
6. Call INCRTM to increment time values.
7. Call JEXEC to call beginning-of-step routines.
8. Call JEXEC to call iterative routines for the first iteration.
9. Call JEXEC to call noncoverged interative routines.
10. If any routines have not converged, go to Step 9.
11. Call JEXEC to call end-of-step routines.
12. Exit.

b. Subroutine CCM.

Description

For a discussion of this subroutine, see routine of same name in the LOADS Program.

c. Subroutine ERR.

Description

Subroutine ERR prints a terminal error message, then aborts the program.

d. Subroutine INCRTM.

Description

Subroutine INCRTM increments the month, day-of-month, and hour indexes by one hour.

e. Subroutine JEXEC.

Description

Subroutine JEXEC controls calling of individual components routines. Routines are called if (1) they fall within the component sequencing groups as set by CBS and (2) they meet the nonconvergence criteria as set by CBS.

Calculation Procedure

1. Loop through 7 for all components within sequencing group.
2. Fetch inputs, if there are any.
3. If nonconvergence is not required, skip to Step 5.
4. Check all inputs for convergence. If all have converged, skip to Step 1.
5. Get data pointers and call the component routine.
6. Save current input values for convergence testing.
7. Call the trace routine if the trace flag was set by the input processor.
8. Exit.

f. Subroutine RDCOMP.

Description

Subroutine RDCOMP reads the component data assembled by the input processor, including data pointers, initial output values, etc.

g. Subroutine REDUC.

Description

Subroutine REDUC analyzes data intended for reports according to various assigned functions and reduces the data to a particular frequency (monthly or daily).

h. Subroutine SOLCOM.

Description

Subroutine SOLCOM calculates the hourly solar components for TYPE16. The three calculational methods are:

1. Boeing cloud cover,
2. Liu and Jordan (Ref. 16) separation of measured direct and diffuse radiation (not currently used), and
3. Solmet measured data (not currently used).

Calculation Procedure (Boeing cloud cover)

1. Calculate beam and diffuse values assuming a clear day.
2. Calculate cloud cover modifier.
3. Calculate cloudy day beam and diffuse values.
4. Calculate beam and diffuse values for tilted collector surface.

i. Subroutine SOLDAY.

Description

Subroutine SOLDAY calculates daily solar data for TYPE16 for the insolation method in use.

Calculation Procedure

1. Calculate Fourier coefficients for curve fits of items in Step 2.
2. Calculate tangent of solar declination, equation of time solar constant, sunrise and sunset hour angle, and atmospheric extinction coefficients.
3. If cloud cover method is used, calculate sky diffusivity.
4. Calculate the time of sunrise and sunset.
5. If cloud cover method is used, set clearness number.
6. If cloud cover method is not used, set ground reflectance.

j. Subroutine SOLPOS.

Description

Subroutine SOLPOS calculates the solar hour angle and the solar cosines.

k. Subroutine TRACE.

Description

Subroutine TRACE prints the descriptions and values of constants, inputs, and outputs of a component if the current time is between trace limits for that component.

l. Subroutine TYPE1.

Description

TYPE1 consists of four collector models of varying complexity. The differences are as follows:

1. MODE 1 - The collector loss coefficient (UL) and the transmittance-absorptance ($\tau\alpha$) product are constant.
2. MODE 2 - UL is calculated; $\tau\alpha$ is constant.
3. MODE 3 - UL is constant; $\tau\alpha$ is calculated.
4. MODE 4 - UL and $\tau\alpha$ are calculated.

Currently MODE 4 is the only mode in use.

Calculation Procedure

1. If this is not an initialization iteration, go to Step 4.
2. Calculate the constants.
3. Exit.
4. If there is no insolation, go to Step 11.
5. Calculate $\tau\alpha$ if it is not constant.
6. Calculate UL if it is not constant.
7. Calculate the collector outlet temperature.
8. If UL is not constant, go to Step 6 for one iteration only.
9. Set the output values.
10. Exit.
11. Set the outlet temperature to zero to insure that the collector is not on.
12. Exit.

m. Subroutine TYPE2.

Description

TYPE2 models a differential controller with hysteresis. The controller does not turn on until the steady-state difference between the two signals is greater than the first prescribed difference. Once on, the controller does not turn off until the steady-state difference between the two inputs is less than the second prescribed difference. To avoid excessive oscillation, the controller "sticks" the output after a prescribed number of iterations.

Calculation Procedure

1. If this is the first iteration of the time step, save the output signal.
2. If the time is to "stick" controller, exit.
3. If the time step initial signal is "on", go to Step 7.
4. Set the signal to "off."
5. If the input difference is greater than the first prescribed difference, set the signal to "on."
6. Exit.
7. Set the signal to "on."
8. If the input difference is less than the second prescribed difference, set the signal to "off."
9. Exit.

n. Subroutine TYPE3.

Description

TYPE3 models a fluid circulation device (pump, fan, etc). The flow rate is calculated to be the user-specified capacity multiplied by the control signal, which may vary from 0.0 to 1.0.

o. Subroutine TYPE4.

Description

TYPE4 models an unstratified liquid storage tank with an optional heater.

Calculation Procedure

1. If this is not an initialization iteration, go to Step 4.
2. Calculate the constants.
3. Exit.
4. If this is not the first iteration of the time step, go to Step 7.
5. Turn on the heater if the tank temperature is below set point.
6. If the environment flag is zero, set the environment temperature to ambient temperature.
7. Calculate the losses to environment.
8. Calculate the heat from the collector.
9. Calculate the heat to load.
10. Calculate the change in tank temperature.
11. Set the output values.
12. Exit.

p. Subroutine TYPE5.

Description

TYPE5 models four types of heat exchangers: parallel, counterflow, crossflow, and constant effectiveness. (The first three types are not currently used.)

Calculation Procedure (constant effectiveness)

1. If this is not an initialization iteration, go to Step 4.
2. Set the constants.
3. Exit.
4. Calculate the hot and cold capacitance rates.
5. Calculate the minimum and maximum capacitance rates.

6. Calculate the heat transfer rate.
7. Calculate the hot and cold outlet temperatures.
8. Set the output values.
9. Exit.

q. Subroutine TYPE10.

Description

TYPE10 models a multinode rock-bed storage unit. Included in the model is a mixing plenum at each end of the rock bed. The number of nodes is specified by the user.

Calculation Procedure

1. If this is not an initialization iteration, go to Step 4.
2. Calculate the constants.
3. Exit.
4. If this is a first iteration of the time step, update the node temperatures.
5. If the environment flag is zero, set the environment temperature to ambient temperature.
6. Calculate the net flow through the rock bed.
7. If the net flow is negative, go to Step 8.
   If the net flow is zero, go to Step 10.
   If the net flow is positive, go to Step 12.
8. Calculate temperature changes for all nodes based on flow rate and upstream node temperature.
9. Go to Step 13.
10. Calculate temperature changes for all nodes based on rock-bed conductivity and environment temperature.
11. Go to Step 13.
12. Calculate temperature changes for all nodes based on the flow rate and upstream node temperature.
13. Set the output values.
14. Exit.

r. Subroutine TYPE16.

Description

TYPE16 is the executive routine of the insolation component that calculates insolation on the collector surface. The three methods of computation are (1) Boeing cloud cover, (2) Liu and Jordan, and (3) Solmet; currently only the first method is in use.

Calculation Procedure (MODE 1)

1. If this is not an initialization iteration, go to Step 4.
2. Calculate the constants.
3. Exit.
4. If it is the first hour of the day, call SOLDAY to calculate the daily variables.
5. If the sun is not up at all during the current hour, go to Step 10.
6. Call SOLPOS to get the sun position.
7. Call SOLCOM to get the solar components on the collector surface.
8. Set the output values.
9. Exit.
10. Set the output values to zeros.
11. Exit.

s. Subroutine TYPE21.

Description

TYPE21 is a component routine that includes a constant loss coefficient collector, an unstratified liquid storage tank, two pumps, a constant effectiveness heat exchanger, and a differential temperature controller in a standard liquid system configuration.

Calculation Procedure

1. If this not an initialization iteration, go to Step 4.
2. Calculate the run constants.
3. Exit.
4. If this is not the first iteration of the time step, go to Step 7.
5. Set the tank environment temperature to ambient temperature if the environment flag is zero.
6. Calculate the time step constants.
7. Calculate the collector outlet temperature.
8. If the temperature is too low, turn off the collector pump and go to Step 10.
9. Calculate the energy lost because of boiling.
10. Calculate the change in tank temperature.
11. Calculate the tank losses to environment.
12. Set the output values.
13. Exit.

t. Subroutine TYPE22.

Description

TYPE22 is a component that models the following types of equipment in a standard air system configuration: a constant loss coefficient collector, a collector fan, a differential temperature controller, and a five-node rock bed storage unit.

Calculation Procedure

1. If this is the first iteration of the time step, go to Step 5.
2. If this is after the first iteration of the time step, go to Step 6.
3. Calculate the run constants.
4. Exit.
5. Update the rock bed temperatures.
6. Calculate the collector inlet temperature.
7. Calculate the collector outlet temperature.
8. If the iteration limit has not been exceeded, turn the collector fan on or off in accordance with temperature difference requirement of types 6 and 7. Otherwise, leave the fan as is.
9. Determine the amount and direction of air flow through the rock bed.
10. Calculate the rock bed node temperature changes based on air flow rate, conductivity, and environment temperature.
11. Set the output values.
12. Exit.

u. Subroutine TYPE25.

Description

TYPE25 controls the output of data to the report generator. If the current time is within the report period, REDUC is called to reduce the data to the proper frequency. At the end of a given frequency, the necessary report data are written to the report file for processing by the report generator at the termination of the simulation.

v. Subroutine TYPE28.

Description

TYPE28 calls the weather-reading routine and puts hourly weather values into the output array.

w. Subroutine TYPE29.

Description

TYPE29 models an auxiliary controller for a liquid solar system.

Three loads are passed to the routine: preheat, heating, and reheat loads. All three loads are assumed to be delivered through liquid-to-air heat exchangers. The liquid side of the exchanger is assumed to be the side with the minimum capacitance rate. The heat exchanger is assumed to have a constant effectiveness.

Calculation Procedure

1. If this is not an initialization iteration, go to Step 4.
2. Calculate the run constants.
3. Exit.
4. If the maximum preheat flow rate is zero, assume there is no preheat coil and add the preheat load into the heating load.
5. If the time is to "stick" controller, modify the return temperature and then exit.
6. Loop through Step 18 for all three loads.
7. If the load is zero, go to Step 18.
8. Increment the total load.
9. If there is no flow to these coils, go to Step 18.
10. Calculate the required inlet temperature on the liquid side of the exchanger to satisfy the load at maximum flow rate.
11. If the storage temperature is high enough, go to Step 16.
12. If the mode is parallel, go to Step 18.
13. Calculate return temperature using boost.
14. If there is less than 0% solar, go to Step 18.
15. Increment the total flow rate and total load on the tank; then go to Step 18.
16. Calculate the return temperature with modulated flow rate.
17. Increment the total flow rate and the total load on the tank.
18. This is the end of the loop for three loads.
19. If there is a nonzero flow rate, calculate the new return temperature.
20. Set the output values.
21. Exit.

x. Subroutine TYPE31.

Description

TYPE31 models an auxiliary controller for an air system. The load is defined to be the preheat load as determined by the SYSTEMS Program. Air is diverted through the rock bed from the main supply duct. The inlet air to the rock bed is at mixed return and outside air temperature. The rock-bed air-flow rate is modulated between zero and the user-supplied maximum in an attempt to deliver the required load. Auxiliary energy is added to meet only the unmet portion of the load.

Calculation Procedure

1. If this is an initialization iteration, calculate the run constants and then exit.
2. If the time is to "stick" controller, go to Step 12.
3. If the hot end of the rock bed is too cold, set the load on the rock bed to zero; then go to Step 10.
4. If the rock bed can meet the entire load, go to Step 8.
5. Set the flow rate to maximum.
6. Calculate the load on the rock bed.
7. Go to Step 10.
8. Modulate the flow rate to deliver the required load.
9. Calculate the load on the rock bed.
10. Set the output values.
11. Exit.
12. Calculate the new flow rate.
13. If the flow rate must be raised, go to Step 4.
14. Exit.

y. Subroutine WDREAD.

Description

Subroutine WDREAD reads the weather file and places values into hourly variables.

VI. ECONOMICS PROGRAM

A. Objective and Description

The ECONOMICS Program calculates the Life-Cycle Costs (LCC) for each of the various alternatives examined using the LOADS, SYSTEMS, or PLANT Programs. The methodology used is derived from the ERDA manual Life Cycle Costing Emphasizing Energy Conservation (Ref. 17). These guidelines were developed in response to rapidly rising energy costs and the consequent need to examine existing and proposed DOE buildings with a view to possible modifications that would be both cost effective and energy conservative. By the life-cycle costing method, a few numbers, "investment statistics," are calculated that measure the cost effectiveness or "profitability" of each project as compared to a reference or "baseline" case defined by the user. Competing alternatives can then be ranked on the basis of these statistics to assess which will be the most cost-effective system.

The subroutines in the ECONOMICS Program are designed to follow the step-by-step life-cycle costing process described in the Cal-ERDA Users Manual, Chap. VI. They perform the following series of operations.

1. Input. The ECONOMICS Program requires two standard input files. The file ECOSTD contains all nonplant cost items, including items in the baseline case to which alternatives will be compared by the program. A description of these data is found in the Cal-ERDA Users Manual, Chap. VI.B, "BDL Input Instructions." Subroutine RDESF reads this file, multiplying certain variables that were input in convenient form by scaling factors (e.g., 10^9) required to bring them to their true values. All plant cost data are contained in PCF, the plant costs file, and are read by subroutine RDPCF. In addition, RDPCF calculates totals of input equipment costs, utilities, and energy use.

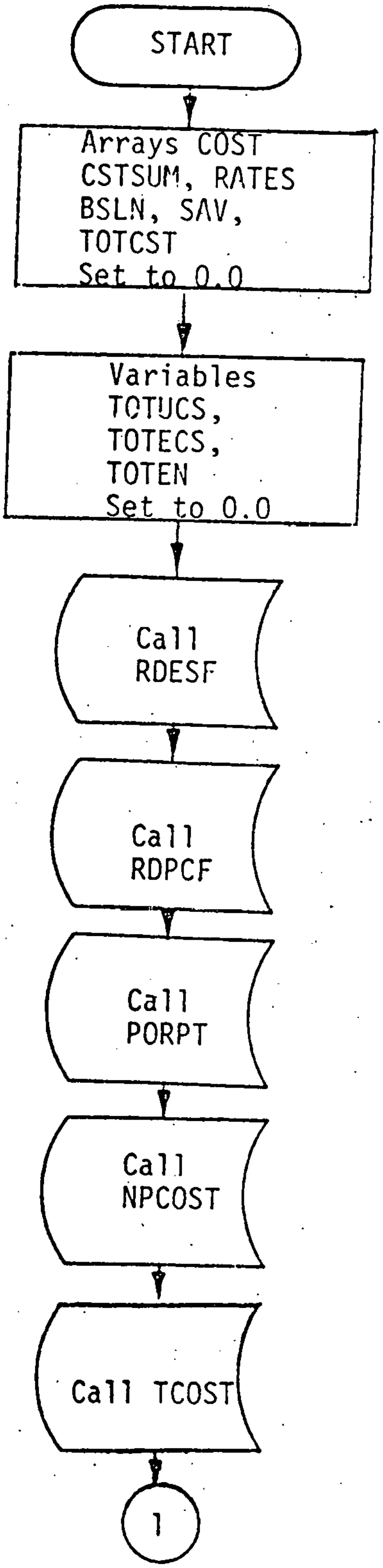
2. Calculations. Subroutine NPCOST calls the function PVF to calculate "present value" of annual costs, maintenance costs, consumables costs, annual costs (maintenance plus consumables), cyclical costs (major and minor overhauls), replacement costs, and yearly cyclical costs (major and minor overhaul plus replacement cost/year) for each nonplant cost item. Present value calculation is a method for adjusting the value of a dollar for the time-value of money, that is, its potential earning over time. (See the Cal-ERDA Chap. VI.C, for a detailed explanation of present value estimation.) NPCOST then sums costs for each item into the CSTUM array.

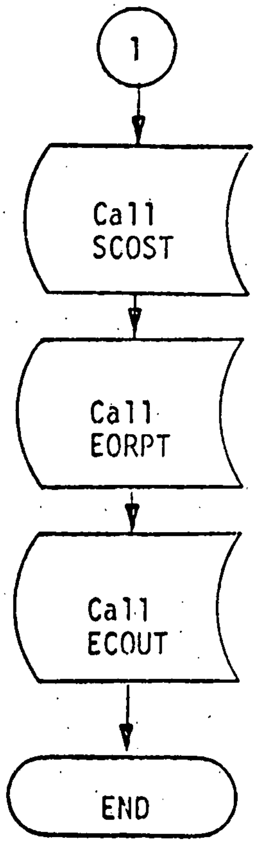
Subroutine TCOST uses the CSTSUM array to calculate total costs of various kinds, including both plant and nonplant costs, across all the items, into the TOTCST array. Subroutine SCOST uses these sums to determine cost and fuel savings of the particular alternative being examined over the baseline case. Note that the treatment of discounted payback period in Cal-ERDA ECON does not follow the method suggested in Ref. 17.

3. Output. Four output subroutines (ECOUT, PORPT, EVRPT, EORPT) are available. ECOUT and PORPT are incomplete now. EVRPT is called by NPCOST to print a report of costs, present value costs, and totals calculated in NPCOST. EORPT produces a formal output report of input costs, totals, and savings over the baseline case. Additionally, the function PVF and all the other subroutines except one contain instructions for printing lists of variable values for debugging purposes.

Note that four subprograms (R4PRNT COSTEN, COSTEQ, and Function CYC), are discussed here, but do not appear in the October 1, 1977, version of ECON. At that date, these subprograms appeared in the PLANT program of Cal-ERDA. However, since future versions of ECON will contain them and because their concern is with cost analysis, they are discussed in this section of the Program Manual.

B. ECONOMICS Program Flow Chart.


START
Arrays COST CSTSUM, RATES BSLN, SAV, TOTCST Set to 0.0
Variables TOTUCS, TOTECS, TOTEN Set to 0.0
Call RDESF
Call RDPCF
Call PORPT
Call NPCOST
Call TCOST
1


1
Call
SCOST
Call
EORPT
Call
ECOUT
END

C. List of Variables Used in ECON

Variable	Also Called	Contains	Source
AC	COST (I,10), CSTSUM (I,3)	Annual cost ($)	Input, or calculated in NPCOST
ACPU	COST (I,11)	Annual cost/unit ($)	Input
ACPV	CSTSUM (I,4)	Present value, annual cost ($)	Calculated in NPCOST
ALFCYC (1)			Input
ALFCYC (2)			Input
ALFCYC (3)			Input
ALFCYC (4)			Input
ALFCYC (5)			Input
ALFCYC (6)			Input
ALFCYC (7)			Input
BSLN (1)		Plant (nonfuel) life-cycle cost ($K)	Input
BSLN (2)		Total fuel use (10^9 Btu)	Input
BSLN (3)		Total fuel cost ($K)	Input
BSLN (4) -- BSLN (10)		Presently unused	
CC	COST (I,16)	Consumables cost ($)	Input, or calculated in NPCOST
CCPV	CSTSUM (I,8)	Present value consumables cost ($)	Calculated in NPCOST
CI	COST (I,8) CSTSUM (I,2)	Installation cost ($)	Input, or calculated in NPCOST
CM	COST (I,14)	Maintenance cost ($)	Input, or calculated in NPCOST
CMAJ	COST (I,23) CSTSUM (I,13)	Major overhaul cost ($)	Input, or calculated in NPCOST
CMAJPV	CSTSUM (I,14)	Present value major overhaul cost ($)	Calculated in NPCOST
CMAJY		Major overhaul cost/year ($)	Calculated in NPCOST
CMIN	COST (I,20) CSTSUM (I,11)	Minor overhaul cost ($)	Input, or calculated in NPCOST
CMINPV	CSTSUM (I,12)	Present value minor overhaul cost ($)	Calculated in NPCOST
CMINY		Minor overhaul cost/yr ($)	Calculated in NPCOST

CMPV	CSTSUM (I,6)	Present value annual maintenance cost ($)	Calculated in NPCOST
COST, I=1 to NPV			
COST (I,1) and COST (I,2)		Two words; cost-name (user name)	Input
COST (I,3) and COST (I,4)		Two words; unit name	Input
COST (I,5)	U	Number of units	Input
COST (I,6)	FC	First cost ($)	Input
COST (I,7)		First cost per unit ($)	Input
COST (I,8)	CI	Installation costs ($)	Input
COST (I,9)		Installation cost per unit ($)	Input
COST (I,10)	AC	Annual cost ($)	Input
COST (I,11)	ACPU	Annual cost per unit ($)	Input
COST (I,12)	CYC	Cyclical cost ($)	Input
COST (I,13)	CYCPU	Cyclical cost per unit ($)	Input
COST (I,14)	CM	Maintenance cost ($)	Input
COST (I,15)		Maintenance cost per unit ($)	Input
COST (I,16)	CC	Consumables cost ($)	Input
COST (I,17)		Consumables cost per unit ($)	Input
COST (I,18)	EQLIFE	Equipment life (yrs)	Input
COST (I,19)		Minor overhaul interval (yrs)	Input
COST (I,20)	CMIN	Minor overhaul cost ($)	Input
COST (I,21)		Minor overhaul cost per unit ($)	Input
COST (I,22)		Major overhaul interval (yrs)	Input
COST (I,23)	CMAJ	Major overhaul cost ($)	Input
COST (I,24)		Major overhaul cost per unit ($)	Input
CREP	CSTSUM (I,15)	Cost of replacement ($)	Calculated in NPCOST
CREPPV	CSTSUM (I,16)	Present value replacement cost ($)	Calculated in NPCOST
CREPY		Replacement cost/year ($)	Calculated in NPCOST

CSTSUM, I=1 to NPV			
CSTSUM (I,1)	FC	First cost ($)	Calculated in NPCOST
CSTSUM (I,2)	CI	Installation cost ($)	Calculated in NPCOST
CSTSUM (I,3)	AC	Annual cost ($)	Assigned in NPCOST
CSTSUM (I,4)	ACPV	Present value annual cost ($)	Calculated in NPCOST
CSTSUM (I,5)	CM	Maintenance cost ($)	Calculated in NPCOST
CSTSUM (I,6)	CMPV	Present value maintenance cost ($)	Calculated in NPCOST
CSTSUM (I,7)	CC, COST (I,12)	Consumables cost ($)	Assigned in NPCOST
CSTSUM (I,8)	CCPV	Present value consumables cost ($)	Calculated in NPCOST
CSTSUM (I,9)	CYC, COST (I,12)	Cyclical cost ($)	Assigned in NPCOST
CSTSUM (I,10)	CYCPV	Present value cyclical cost ($)	Calculated in NPCOST
CSTSUM (I,11)	CMIN	Minor overhaul cost ($)	Calculated in NPCOST
CSTSUM (I,12)	CMINPV	Present value minor overhaul cost ($)	Calculated in NPCOST
CSTSUM (I,13)	CMAJ	Major overhaul cost ($)	Calculated in NPCOST
CSTSUM (I,14)	CMAJPV	Present value major overhaul cost ($)	Calculated in NPCOST
CSTSUM (I,15)	CREP	Replacement cost ($)	Calculated in NPCOST
CSTSUM (I,16)	CREPPV	Present value replacement cost ($)	Calculated in NPCOST
CSTSUM (I,17)	CTOTPV	Overall cost for this item ($)	Calculated in NPCOST
CSTSUM (I,18)-CSTSUM (I,20)		Presently unused	
CTOTPV	CSTSUM (I,17)	Overall cost for this item ($)	Calculated in NPCOST
CYC	COST (I,12), CSTSUM (I,9)	Cyclical cost ($)	Input
CYCPU	COST (I,13)	Cyclical cost per unit ($)	Input
CYCPV	CSTSUM (I,10)	Present value cyclical cost ($)	Calculated in NPCOST
DR	RATES (1)	Discount rate (%)	Input
ENCOST (1)-ENCOST (10)		Energy cost/utility ($)	Input
ENUSE (1-13, 1-10)		Energy use	Input
EQHT (1-5, 1-25)			

EQHT (5,I) Sum is TOTECS			
EQLIFE	COST (I,18)	Equipment life (yrs)	Input
IESF		Tape unit number for energy standard input file	Assigned in ECON
IEOUTF		Unit number for energy output file	Assigned in ECON
ILSF		Unit number	Assigned in ECON
IPCF		Tape unit number for plant costs input file	Assigned in ECON
IRFPF		Unit number	Assigned in ECON
ISSF		Unit number	Assigned in ECON
NEDATA		Number of plant energy data	Input
NEQSIZE(I)		Number of equipment sizes of type I	Input
NPC		Number of nonplant costs	Input
NUTLTY		Number of utilities	Input
RATES (1)	DR	Discount rate (%)	Input
RATES (2)	RL	Labor inflation rate (%)	Input
RATES (3)	RM	Material inflation rate (%)	Input
RATES (4)		Energy inflation rate (%)	Input
RATES (5)	Y	Project life (yrs)	Input
RATES (6)-RATES (10)		Presently unused	
RL	RATES (2)	Labor inflation rate (%)	Input
RM	RATES (3)	Material inflation rate (%)	Input
SAV (1)		Cost savings ($)	Calculated in SCOST
SAV (2)		Investment ($)	Calculated in SCOST
SAV (3)		Savings/investment ratio (SIR)	Calculated in SCOST
SAV (4)		Btu savings	Calculated in SCOST
SAV (5)		Btu savings/investment ratio	Calculated in SCOST
SAV (6)		Discounted pay-back period (yrs)	Calculated in SCOST
TOTCST (1)		Sum of first costs ($)	Calculated in TCOST
TOTCST (2)		Sum of installation costs ($)	Calculated in TCOST

TOTCST (3)		Sum of present value annual costs ($)	Calculated in TCOST
TOTCST (4)		Sum of present value cyclical costs ($)	Calculated in TCOST
TOTCST (5)		Sum of TOTCST (1)-(4) ($)	Calculated in TCOST
TOTCST (6)	TOTECS	Total equipment cost ($)	Calculated in TCOST
TOTCST (7)	TOTUCS	Total utilities cost ($)	Calculated in TCOST
TOTCST (8)	TOTEN	Sum of energy use ($)	Calculated in TCOST
TOTCST (9)		Sum of TOTCST (5)-(7) ($)	Calculated in TCOST
TOTCST (10)		Sum of present value annual maintenance costs ($)	Calculated in TCOST
TOTCST (11)		Sum of present value annual consumables costs ($)	Calculated in TCOST
TOTCST (12)		Sum of present value minor overhaul costs ($)	Calculated in TCOST
TOTCST (13)		Sum of present value major overhaul costs ($)	Calculated in TCOST
TOTCST (14)		Total replacement costs	Calculated in TCOST
TOTECS	TOTCST (6)	Total equipment cost ($)	Calculated in RDPCF
TOTEN	TOTCST (8)	Total energy use ($)	Calculated in RDPCF
TOTUCS	TOTCST (7)	Total utilities cost ($)	Calculated in RDPCF
U	COST (I,5)	Number of units	Input
UDATA (1-6, 1-10)			
Y	RATES (5)	Project life (yrs)	Input

D. Subroutine Listing

1. Subroutine COSTEN
2. Subroutine COSTEQ
3. Function CYC
4. Subroutine ECOUT
5. Subroutine EORPT
6. Subroutine EVRPT
7. Subroutine NPCOST
8. Subroutine PORPT
9. Function PVF
10. Subroutine RDESF
11. Subroutine RDPCF
12. Subroutine R4PRNT

13. Subroutine SCOST
14. Subroutine TCOST

E. Description by Subroutine

Each subroutine is described in detail in terms of the calling and called subprograms, common blocks, declarations, input and output variable definitions, and the calculation procedure used.

1. Subroutine COSTEN*

Description

Subroutine COSTEN calculates energy present-value usage costs for up to ten different energy sources, default being for four sources.

Subprograms Calling This Routine

SUBROUTINE R4PRNT

Subprograms Called by This Routine

None

Common Blocks	Variable Obtained from Common Blocks	Variables Placed in Common Blocks
/STATD/*	ENUSE array, ENPEAK array	None
/UCOSTD/*	UBLK array, NUTLTY, NUBLK array	ENCOST array
/LFCYCD/*	ALFCYC(1), ALFCYC(5)	None

Declarations

None

Input

Name	Description
ENUSE(I,IU)	Monthly (I=1,12) and annual(I=13) amount of energy consumed by each source (Btu)
ENPEAK(I,IU)	Monthly (I=1,12) and annual (I=13) highest demand of each each source (Btu/hr)
UDATA(1,IU)	Energy/unit for utility index IU (Btu)
UDATA(2,IU)	Uniform cost/unit for utility index IU ($)
UDATA(3,IU)	Cost escalation factor for life cycle for utility index IU
UDATA(4,IU)	Minimum peak load charge for utility index IU ($)

*In the October 1, 1977, version of Cal-ERDA, subroutine COSTEN is in the PLANT program. In subsequent versions it will be moved to the ECON program, and all required data from PLANT will be sent in the plant costs file to ECON.

UDATA(5,IU)	Minimum peak load for utility index IU (unit)
UDATA(6,IU)	Peak load unit cost for utility index IU ($/unit)
UBLK(I,J,IU)	Parameters specifying each block of a graduated charge for utility index IU
NUTILTY	Number of different energy sources
NUBLK(IU)	Number of different blocks of a graduated charge for utility index IU
ALFCYC(1)	Interest rate (%)
ALFCYC(5)	Project life (yrs)

Output

Name	Description
ENCOST (IU)	Total escalated yearly charge for each utility index IU ($)

Calculation Procedure

1. Set intermediate variables.

```
A1 = ALFCYC(1)/100.
E  = ALFCYC(5)
```

2. Do Steps 3 through 11 for each energy source.

```
DO 1000 IU = 1,NUTLTY
```

3. Set intermediate variables.

```
NB = NUBLK(IU)
and initialize
CST = 0.
```

4. Do Steps 5 through 10 for each month of the year.

```
DO 500 M = 1,12
```

5. Initialize CSTM = 0.

6. Divide energy consumption by energy for this utility giving unit of energy consumption.

7. If there are no graduated charges, monthly cost is equal to cost per unit of energy times the number of units.

```
CSTM = UDATA (2,IU) * EMON
```

8. If there are graduated charges, this step is performed until all graduated charges have been covered or until all energy use has been charged for.

```
DO 400 IB = 1,NB
```

a. If uniform cost is greater than or equal to zero, block size = block multiplier times the highest hourly peak of energy consumption units in the month.

```
BLOK = UBLK(1, IB,IU)
IF(UDATA(2,IU).GE.0.)BLOK = BLOK * EMPEAK(7,IU)/(UDATA(1,IU))
IF(BLOK.GE.EMON) GO TO 425
```

b. Monthly cost is a running sum of previous month cost and the charge per unit in the block.

```
CSTM = CSTM + BLOK * UBLK(2,IB,IU)
```

c. Energy consumption units are decreased by those units accounted for in the charge block.

```
EMON = EMON - BLOK
```

d. Remaining energy consumption units are charged.

```
CSTM = CSTM + EMON * UBLK(2,IB,IU)
```

9. Peak load charges are added to the cost.

```
CSTM = CSTM + UDATA(6,IU) * AMAX1(5*(ENPEAK(M,IU)
       +ENPEAK(13,IU)/(UDATA(1,IU), UDATA(5,IU))
```

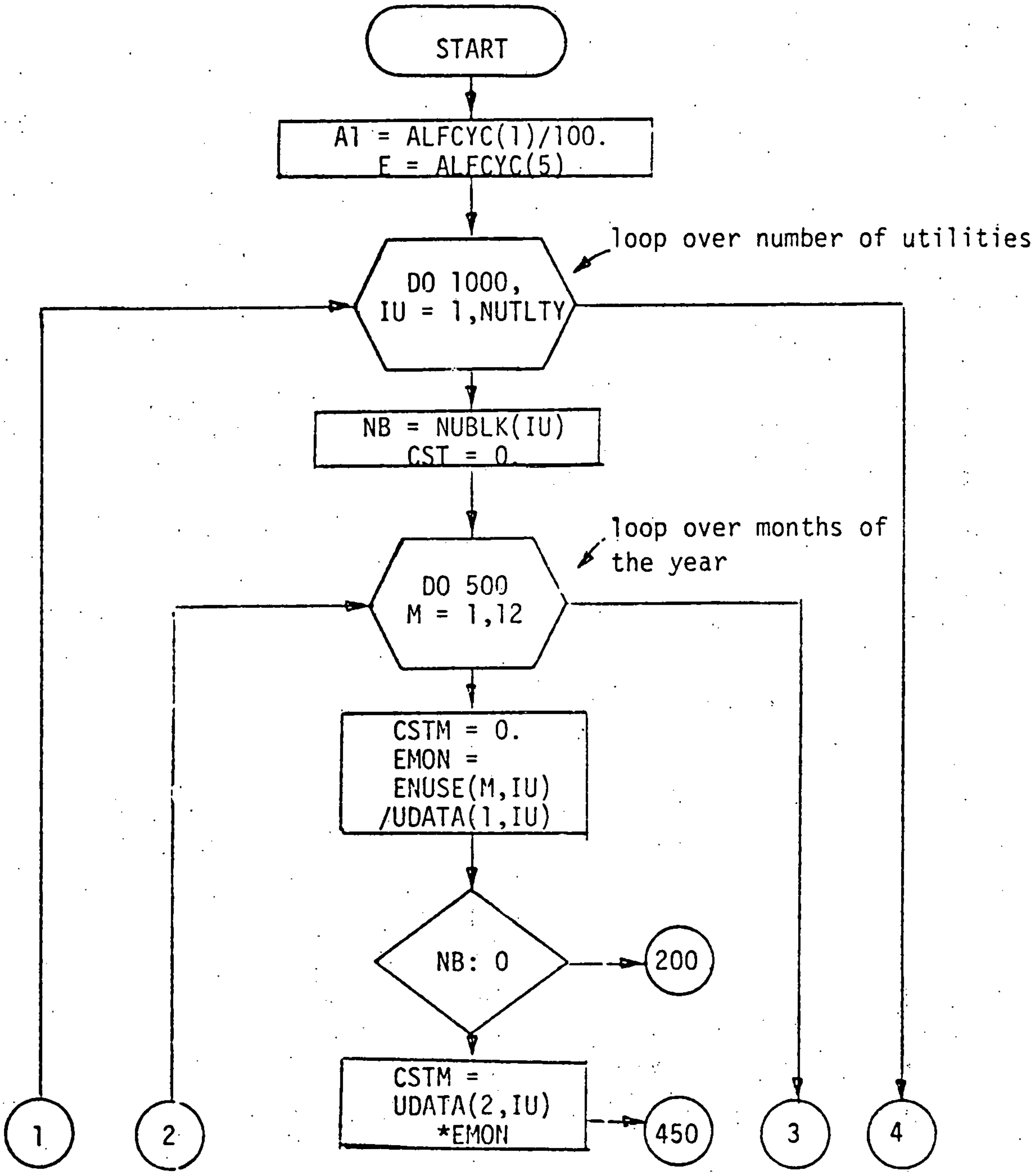
10. Total cost for this utility is calculated as a running sum of the monthly maxima of either the minimum monthly charge or the monthly energy use cost.

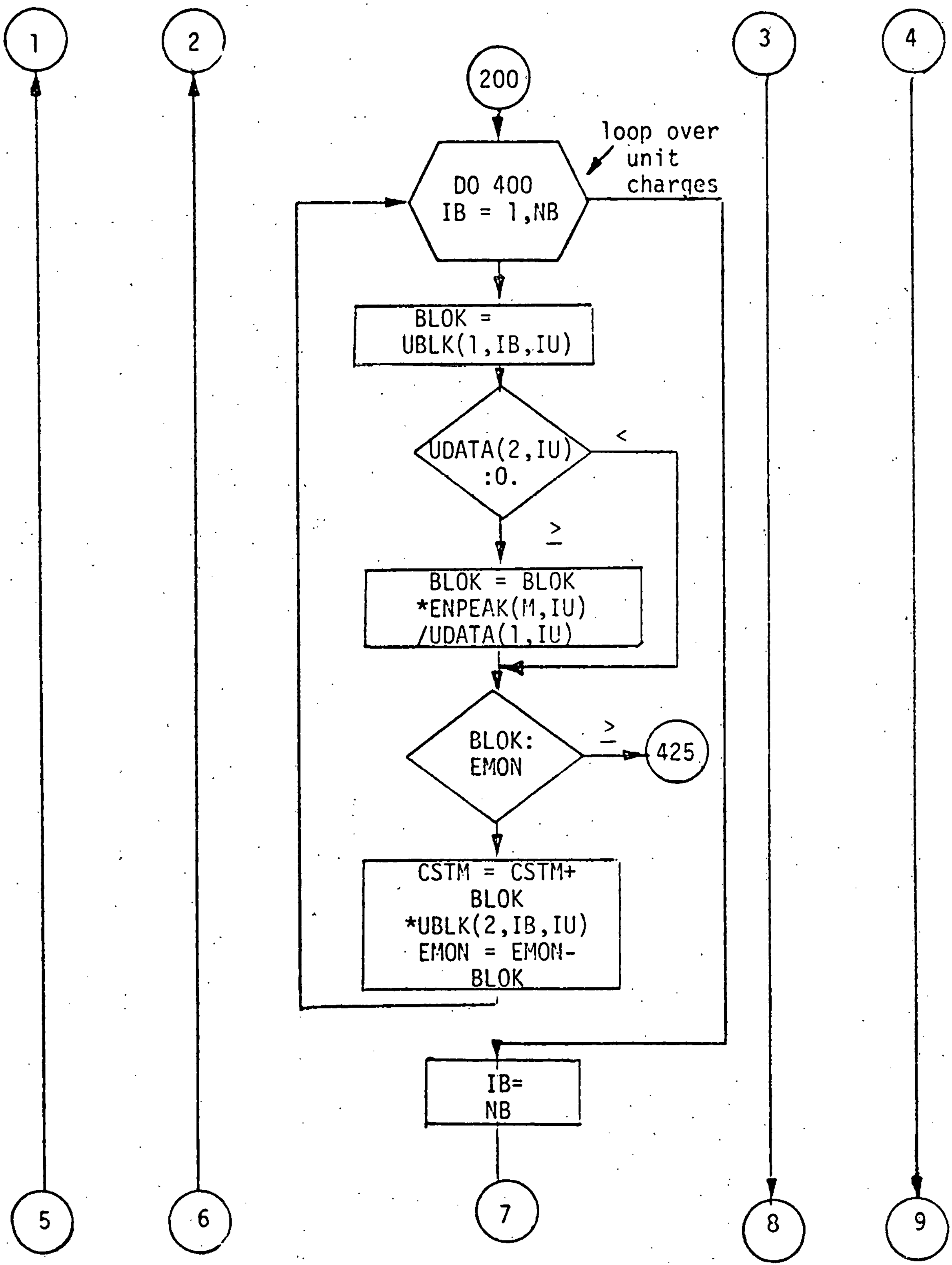
```
CST = CST + AMAX1(UDATA(4,IU) CSTM)
```

11. Total escalated annual charge is calculated as a present value factor times an escalation factor (UDATA(3,IU)) times annual cost.

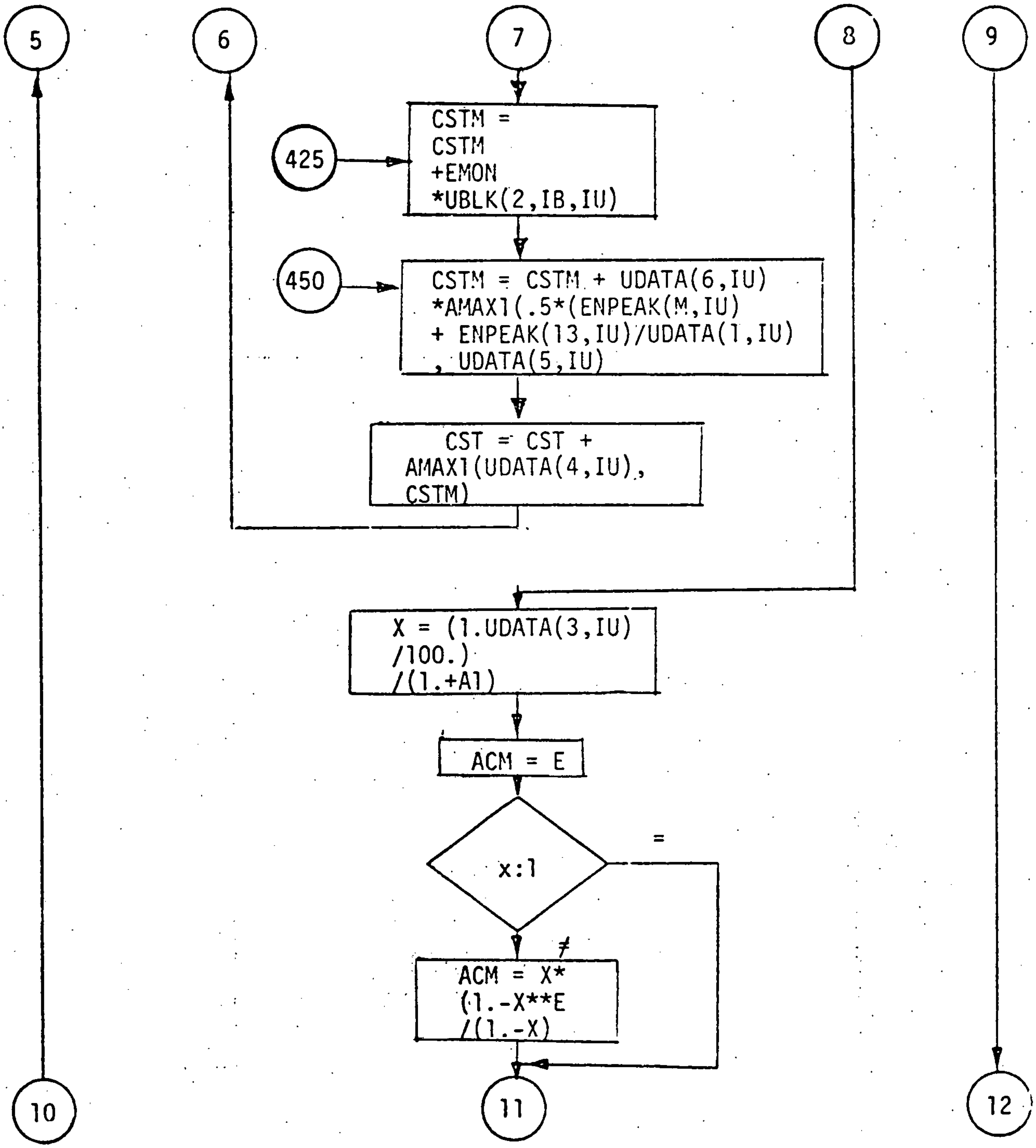
```
X = (1.+UDATA(3,IU)/(100.)/(1.+A1)
ACM = E
IF(X.NE.1.)ACM = X * (1.-X**E)/(1.-X)
ENCOST(IU) = ACM * CST
```

Subroutine COSTEN Flow Chart

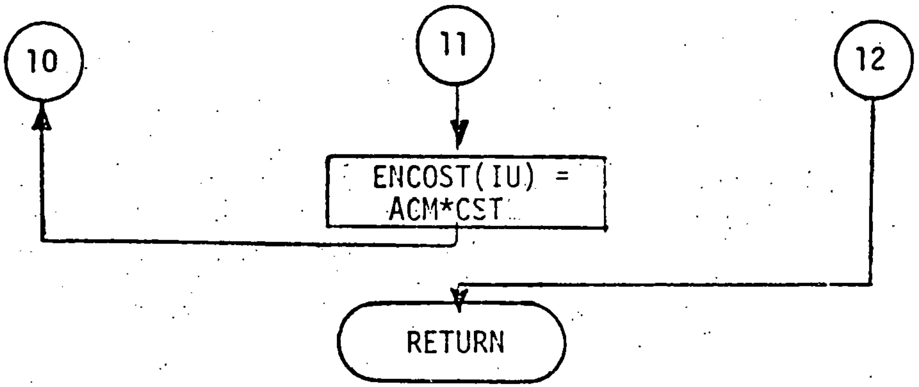



1
2
200
3
4
loop over unit charges
DO 400 IB = 1,NB
BLOK = UBLK(1,IB,IU)
UDATA(2,IU) :0.
<
≥
BLOK = BLOK *ENPEAK(M,IU) /UDATA(1,IU)
BLOK: EMON
≥
425
CSTM = CSTM+ BLOK *UBLK(2,IB,IU) EMON = EMON- BLOK
IB= NB
5
6
7
8
9

Subroutine COSTEN Flow Chart - Continued



Subroutine COSTEN Flow Chart - Continued



2. Subroutine COSTEQ*

Descriotion

Subroutine COSTEQ calculates equipment costs using seven life-cycle parameters, given input data and specific equipment entries on reference equipment.

Subprograms Calling This Routine

SUBROUTINE R4PRNT

Subprograms Called by This Routine

FUNCTION CYC

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/LFCYCD/*	ALFCYC array	None
/ECDATA/*	EQCOSR array	EQCHT array, EQCOST array
/EDATA/*	KINS array, KAV array, NEDATA, NEQSIZE array EQCOSD array	
/STATD/*	IOPRHR array	

Declarations

None

Input

Name	Description
ALFCYC(1)	Discount rate (%)
ALFCYC(2)	Labor inflation rate over general inflation (%)
ALFCYC(3)	Material inflation rate over general inflation (%)
ALFCYC(4)	Energy inflation rate over general inflation (%)
ALFCYC(5)	Project life (yrs)
ALFCYC(6)	Labor cost ($/hr)
ALFCYC(7)	Site cost factor
EQCOSR(1,I)	Reference data equipment size (kW)
EQCOSR(2,I)	Reference data equipment unit cost ($)
EQCOSR(3,I)	Reference data equipment installed cost factor
EQCOSR(4,I)	Reference data equipment consumables ($/hr)
EQCOSR(5,I)	Reference data equipment maintenance
EQCOSR(6,I)	Reference data equipment life (hrs/yr)
EQCOSR(7,I)	Reference data equipment hours to minor overhaul (hrs)

*In the October 1, 1977, version of Cal-ERDA, subroutine COSTEQ is found in the PLANT program. In subsequent versions it will be moved to the ECON program, and all required data from PLANT will be sent in the plant costs file to ECON.

EQCOSR(8,I)	Reference data equipment minor overhaul ($)
EQCOSR(9,I)	Reference data equipment hours to major overhaul (hrs)
EQCOSR(10,I)	Reference data equipment major overhaul cost ($)
EQCOSD(1-10,J,I)	Similar to EQCOSR(1-10), except that data is for given equipment I and size J
IOPRHR(J,I)	Operating hrs/yr for equipment I, size J
KINS	Number of units installed
KAV	Number of units available
NEDATA	Number of different equipment types
NEQSIZE	Number of different sizes of each equipment type

Output

Name	Description
EQCHT(1,I)	Total first cost of equipment index I ($)
EQCHT(2,I)	Total annual cost of equipment index I ($)
EQCHT(3,I)	Total cyclical cost of equipment index I ($)
EQCHT(4,I)	Unused
EQCHT(5,I)	Total cost of equipment index I ($)

EQCOST(K,J,I)	Costs for equipment index I ($)
	K = 1, first costs; K = 2, annual costs;
	K = 3, cyclical costs; K = 4, unused; K = 5, sums of costs

Calculation Procedure

1. If the site-cost factor (ALFCYC(7)) is less than 1, set it equal to 1.

   Initialize

   ```
   A1  = ALFCYC(1)/100., interest rate/100;
   A2  = ALFCYC(2)/100., labor inflation rate/100;
   R   = ALFCYC(3)/100., material inflation rate/100;
   E   = ALFCYC(5), project life (yrs); and
   FCF = 1.0, first cost factor.
   ```

2. Calculate annual cost inflation multiplier.

   ```
   X = (1.+A2)/(1.+A1)
   ACM = E
   IF (X.NE.1) ACM = X * (1.-X**E)/(1.-X)
   ```

3. Enter a DO loop, performing the following calculations for each equipment type I.

4. Initialize output variables.

   ```
   EQCHT(1,I) = EQCHT(2,I) = EQCHT(3,I) = EQCHT(4,I) = EQCHT(5,I) = 0
   ```

   and check to see if we have equipment data for this type; if not, go to next I.

   ```
   NS = NEQSIZE(I)
   IF(NS.LE.0) GO TO 400
   ```

5. Enter a DO loop, looping on each size, J, of equipment type.

   a. Calculate first costs = number of units installed * uniform cost * installed cost factor * first cost factor.

   ```
   EQCOST(1,J,I) = KINS(J,I) * EQCOSD(2,K,I) * EQCOSD(3,J,I) * FCF
   ```

   b. Calculate annual cost = ACM * (consumables cost * operating hours/year + number of units installed * maintenance hour/year * labor cost/hour).

   ```
   EQCOST(2,J,I) = (EQCOSD(4,J,I) * IOPRHR(J,I) + KINS(J,I) * EQCOSD(5,J,I)
   * ALFCYC(6) * ACM
   ```

   c. Calculate cyclical costs of equipment replacement, using FUNCTION CYC.

   ```
   RM = FLOAT(IOPRHR(J,I)) * E/FLOAT(KINS(J,I))
   EQCOST(3,J,I) = CYC(EQCOSD(6,J,I), E, R, RH)
     * KINS(J,I) * EQCOSD(2,J,I) * EQCOSD(3,J,I)
   ```

   d. Calculate cyclical minor overhaul cost.

   ```
   EQCOST(3,J,I) = EQCOST(3,J,I) + CYC(EQCOSD(7,J,I),E,R,RH) * KINS(J,I)
       * EQCOSD(8,J,I)
   ```

   e. Calculate major overhaul cost.

   ```
   EQCOST(3,J,I) = EQCOST(3,J,I) + CYC(EQCOSD(9,J,I),E,R,RH) * KINS(J,I)
     * EQCOSD(10,J,I)
   ```

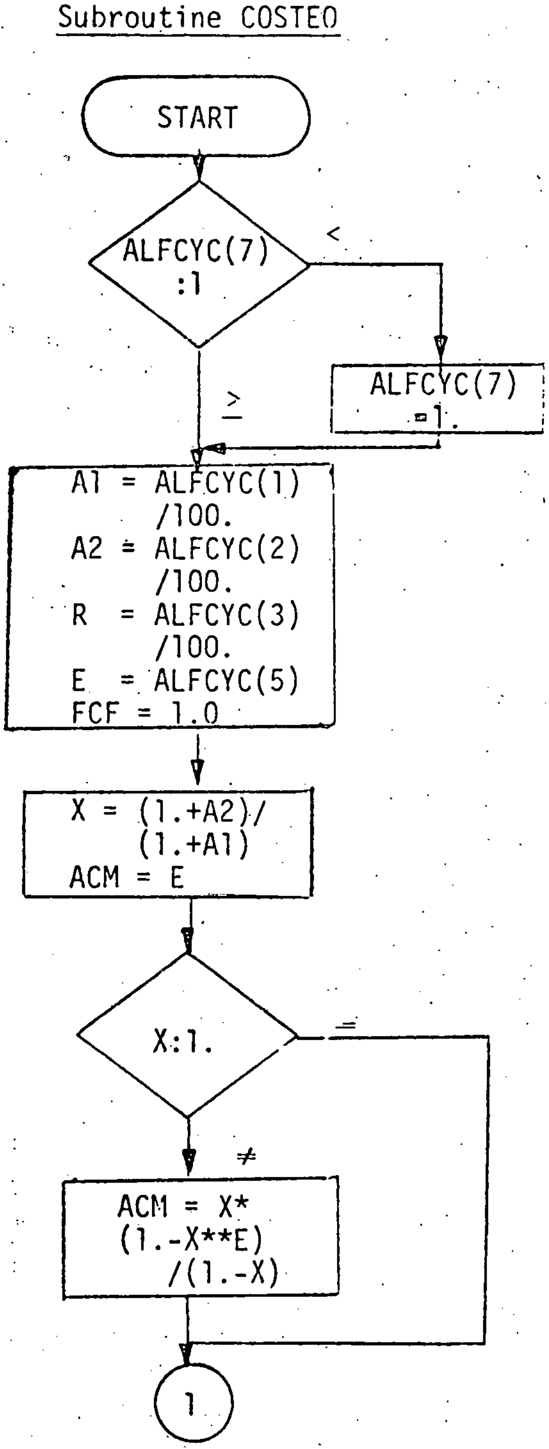
   f. Sum first, annual, and cyclical costs.

   ```
   EQCOST(5,J,I) = EQCOST(1,J,I,) + EQCOST(2,J,I) + EQCOST(3,J,I)
   ```

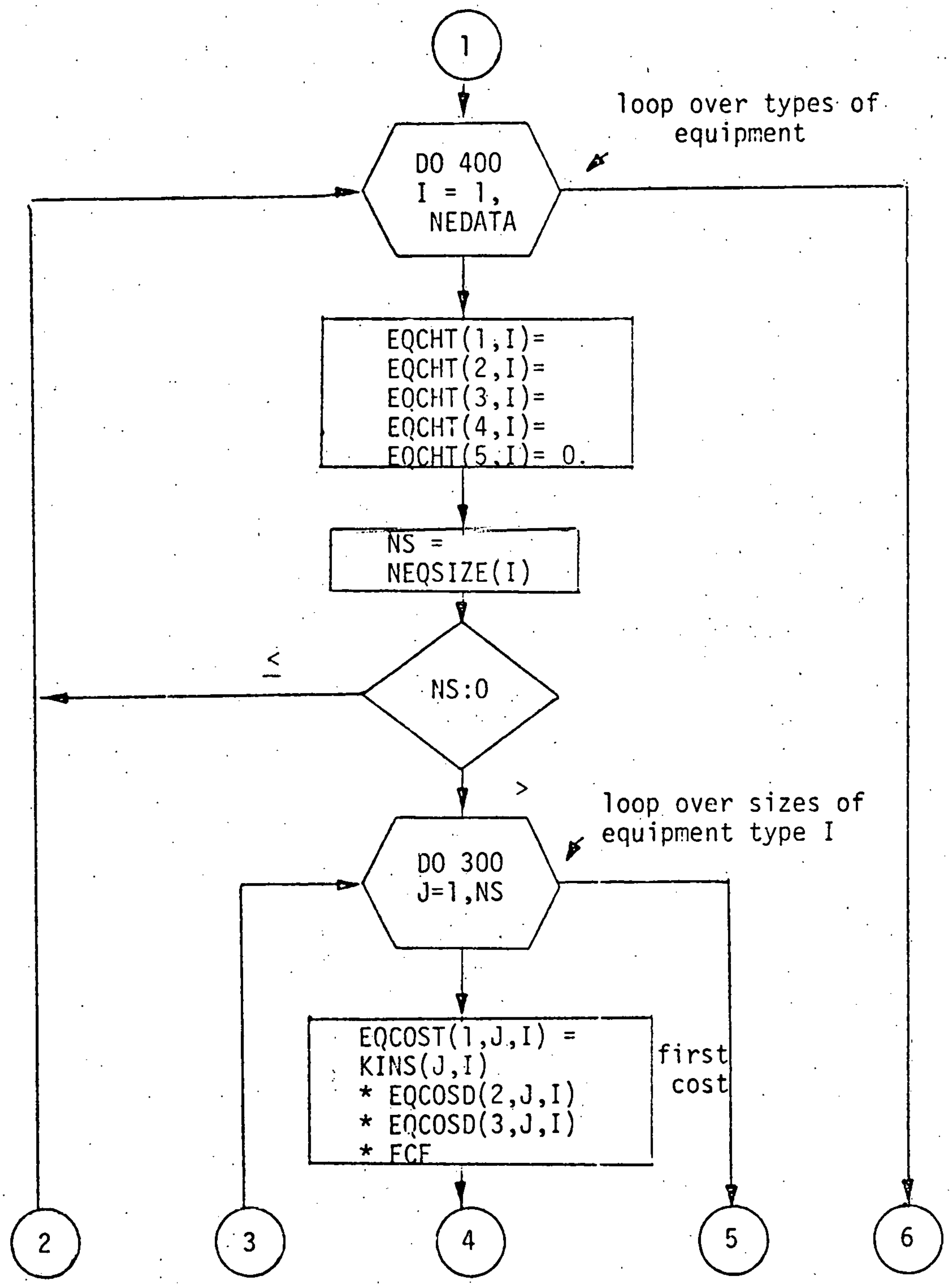
6. For each equipment type I, keep a running total for all sizes, J, of that type.

```
EQCHT(1,I) = EQCHT(1,I) + EQCOST(1,J,I)
EQCHT(2,I) = EQCHT(2,I) + EQCOST(2,J,I)
EQCHT(3,I) = EQCHT(3,I) + EQCOST(3,J,I)
EQCHT(5,I) = EQCHT(5,I) + EQCOST(5,J,I)
```

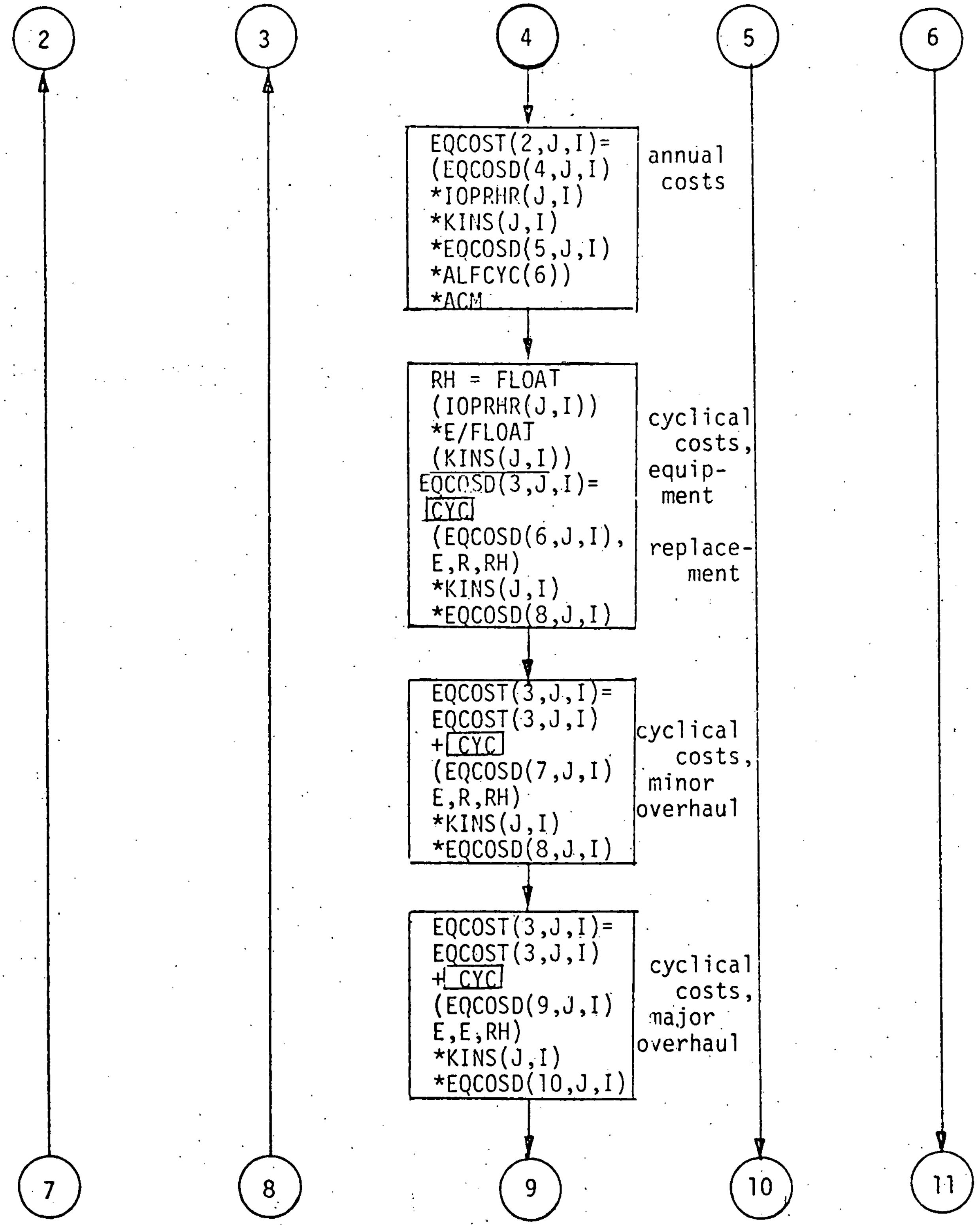
Subroutine COSTEQ Flow Chart



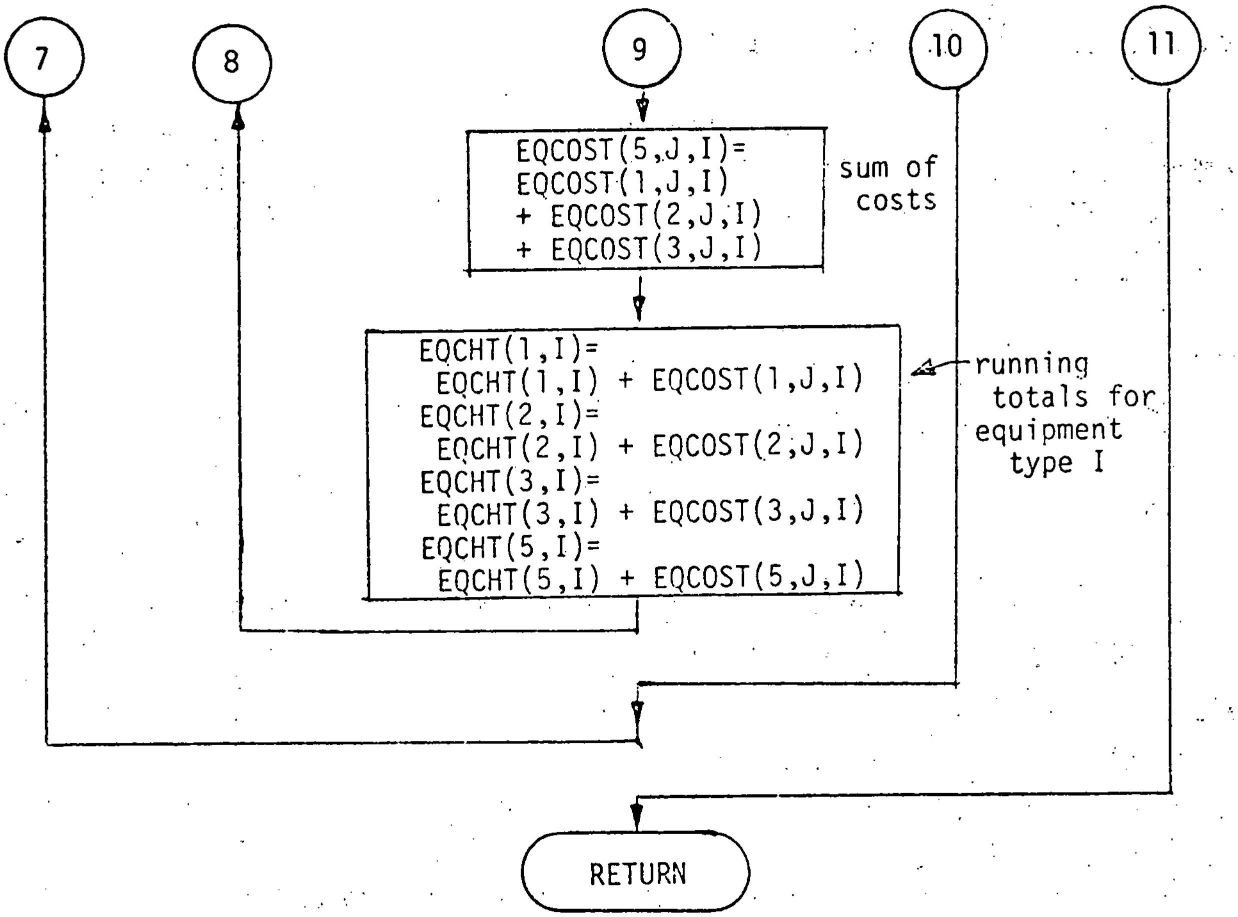
Subroutine COSTEQ Flow Chart - Continued



Subroutine COSTEQ Flow Chart - Continued



Subroutine COSTEQ Flow Chart - Continued


7
8
9
10
11
EQCOST(5,J,I)=
EQCOST(1,J,I)
+ EQCOST(2,J,I)
+ EQCOST(3,J,I)
sum of costs
EQCHT(1,I)=
EQCHT(1,I) + EQCOST(1,J,I)
EQCHT(2,I)=
EQCHT(2,I) + EQCOST(2,J,I)
EQCHT(3,I)=
EQCHT(3,I) + EQCOST(3,J,I)
EQCHT(5,I)=
EQCHT(5,I) + EQCOST(5,J,I)
running totals for equipment type I
RETURN

3. Function CYC*.

FUNCTION CYC (HO, CN, R, RH)

Description

Function CYC calculates a cyclical cost coefficient that includes an inflation factor.

Subprograms Calling This Routine

SUBROUTINE COSTEQ*

Subprograms Calling by This Routine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/LFCYCD/*	ALFCYC(1)	None

Declarations

None

Input

Name	Description
HO	Hours to overhaul or replacement (hrs)
CN	Project life (yrs)
R	Material inflation rate (fraction)
RH	Equipment operation hours during project life
ALFCYC(1)	Interest rate (%)

Output

Name	Description
CYC	Cyclical cost coefficient

Calculation Procedure

1. Check input variables; if unsuitable, set CY to zero and return.

```
IF (HO.EQ.0).OR.(RH.EQ.0.)) GO TO 700
:
700 CYC = 0.
```

* In the October 1, 1977, version of Cal-ERDA, function CYC and subroutine COSTEQ are found in the PLANT program. In subsequent versions, they will be moved to the ECON Program, and all required data from PLANT will be sent in the plant costs file to ECON.

2. Find, F = INT(RH/MO), i.e., ratio of project life operating hours to hours to replacement. If less then 1, set

   CYC = 0. and return.

3. Define PERIOD = CN/(RH/HO).

   Set A1 = ALFCYC(1)/100.

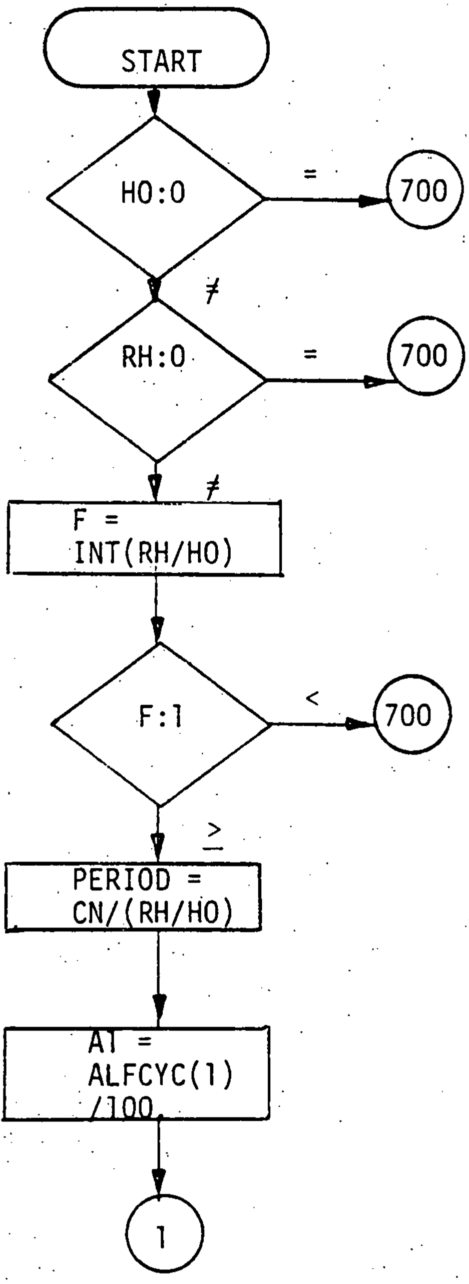
4. Calculate intermediate variable X:

   X = ((1.+R)/(1.+A1))**PERIOD.

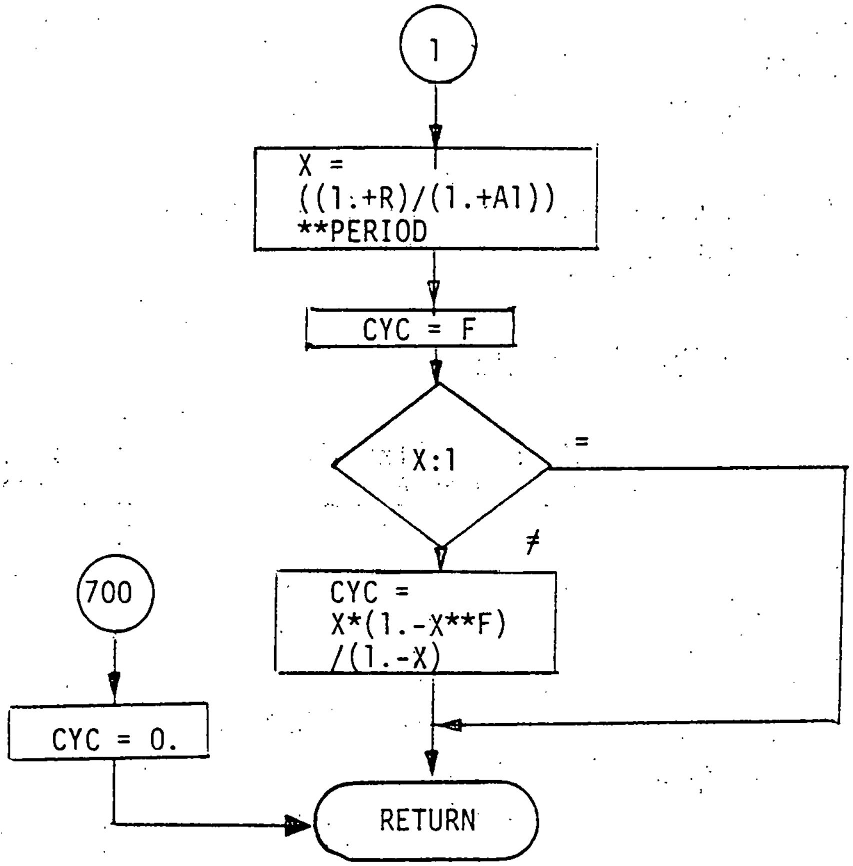
5. Set CYC = F. If X ≠ 1,

   CYC = X*(1.-X**F)/(1.-X).

Function CYC Flow Chart



Function CYC Flow Chart - Continued



4. Subroutine ECOUT.

SUBROUTINE ECOUT - INCOMPLETE

Description

ECOUT writes the output file from the ECON program.

Subprograms Calling This Subroutine

ECON

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/FILES/		None
/COSTS/		

Declarations

None

Input

By Common Block Only

Output

Incomplete

Calculation Procedure

None

5. Subroutine EORPT.

Description

Subroutine EORPT prints an economics life-cycle cost summary report, a report of overall costs, and statistics of life-cycle savings/investment. It also contains an optional debug print of NPC, and the arrays RATES, BSLN, and COST.

Subprograms Calling This Subroutine

ECON

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed In Common Blocks
/FILES/	None	None
/COSTS/	NPC, COST array, CSTSUM array, TOTCST array, BSLN array, SAV array	None

Declarations

None

Input

Common Blocks Only

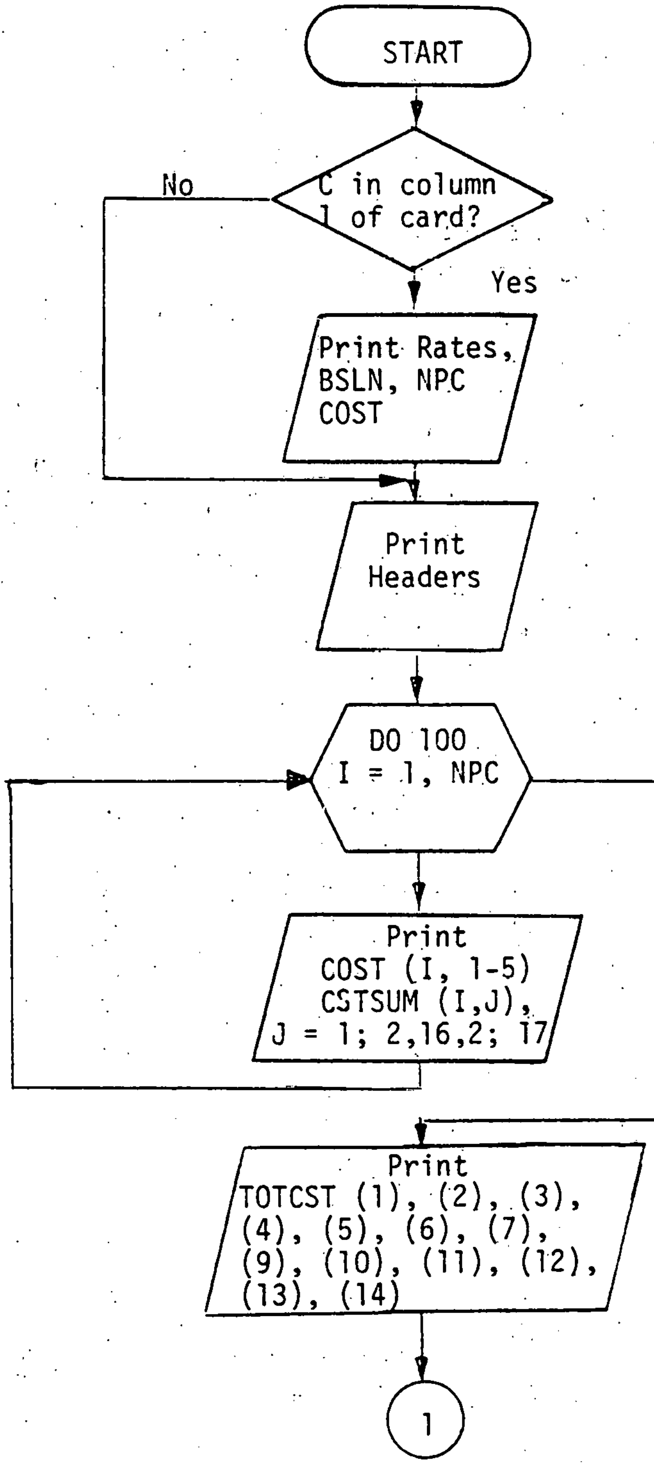
Output

Name	Description
NPC	Number of nonplant costs
COST (NPC, 1-5)	Cost name, unit name (two words each), and number of units for each item
CSTSUM (NPC, I)	I = 1, 2, 4, 6, 8, 10, 12, 14, 16, 17 - for each item, 1 to NPC, first cost, installation cost, present value of annual cost, present value of annual maintenance cost, present value of annual consumables cost, present value of total cyclical cost, present value of minor overhaul cost, present value of major overhaul cost, present value of replacement cost, overall cost of this item
TOTCST (1-17) and (9-14)	Sums of first costs, installation costs, annual costs, cyclical costs, sum of these first 4 costs; sum of equipment costs, sum of utility costs, sum of these; sums of maintenance cost, consumables costs, minor overhaul costs, major overhaul costs, and replacement cost
BSLN (1-3)	Plant life-cycle cost in baseline case, baseline total fuel use, and baseline total fuel cost
SAV (1-6)	Cost savings, investment, savings/investment ratio (SIR), Btu savings, Btu savings investment ratio, discounted payback period

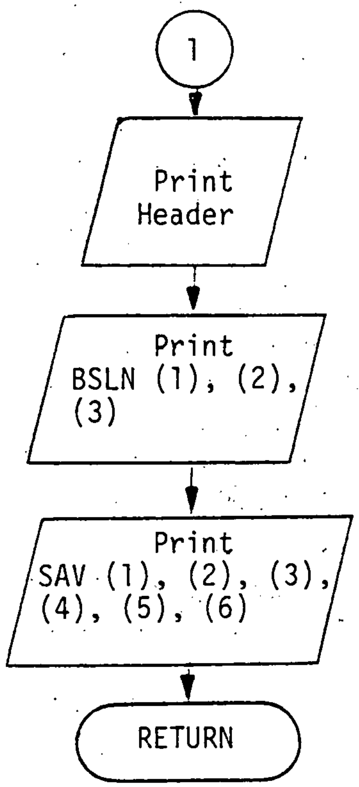
Calculation Procedure

None

Subroutine EORPT Flow Chart


START
C in column 1 of card?
No
Yes
Print Rates, BSLN, NPC COST
Print Headers
DO 100 I = 1, NPC
Print COST (I, 1-5) CSTSUM (I,J), J = 1; 2,16,2; 17
Print TOTCST (1), (2), (3), (4), (5), (6), (7), (9), (10), (11), (12), (13), (14)
1

Subroutine EORPT Flow Chart - Continued



6. Subroutine EVRPT.

Description

EVRPT prints an echo of certain nonplant cost data, both input and calculated, for verification.

Subprograms Calling This Subroutine

SUBROUTINE NPCOST

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/FILES/	None	None
/COSTS/	RATES array, NPC, COST array, CSTSUM array	

Declarations

None

Input

Common Blocks Only

Output

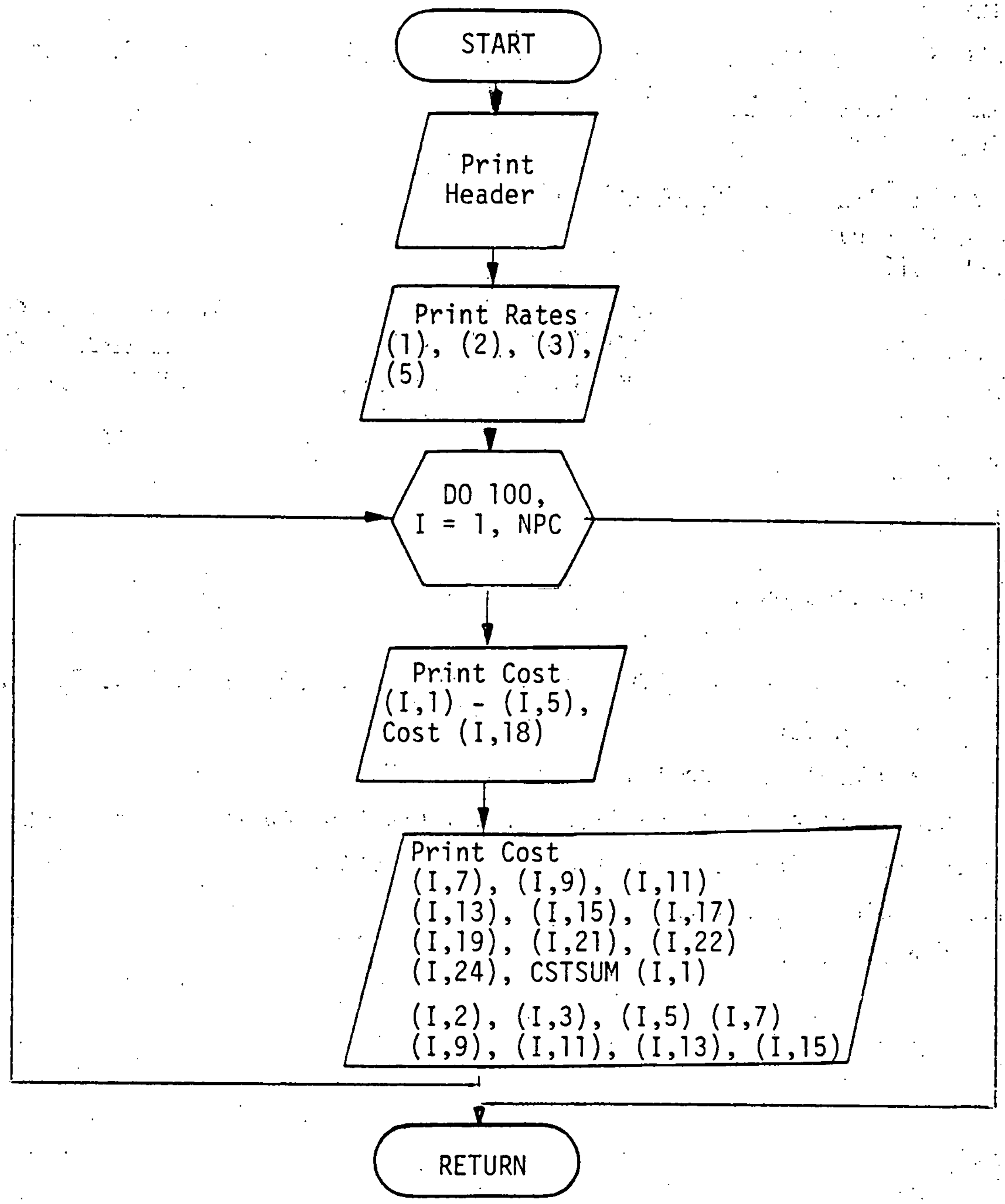
To printer:

Name	Description
RATES(1), (2) (3), (5)	Discount rate (DR), labor inflation rate (RL), material inflation rate (RM), project life (Y)
COST(NPC,I)	I = 1-5, 7, 9, 11, 13, 15, 17-19, 21, 22, 24; cost name (two words), unit name (two words), number of units (U), first cost/unit, installation cost/unit, annual cost/unit (ACPU), cyclical cost/unit, (CYCPU), maintenance cost/unit, consumables cost/unit, equipment life (EQLIFE), minor overhaul interval, minor overhaul cost/unit, major overhaul interval, major overhaul cost/unit
CSTSUM(NPC,I)	I = 1-3, 5, 7, 9, 11, 13, 15; first cost (FC), installation cost (IC), annual cost (AC), maintenance cost (CM), consumables cost (CC), cyclical cost (CYC), minor overhaul cost (CMIN), major overhaul cost (CMAJ), replacement cost (CREP)

Calculation Procedure

None

Subroutine EVRPT Flow Chart


START
Print Header
Print Rates (1), (2), (3), (5)
DO 100, I = 1, NPC
Print Cost (I,1) - (I,5), Cost (I,18)
Print Cost (I,7), (I,9), (I,11) (I,13), (I,15), (I,17) (I,19), (I,21), (I,22) (I,24), CSTSUM (I,1) (I,2), (I,3), (I,5) (I,7) (I,9), (I,11), (I,13), (I,15)
RETURN

7. Subroutine NPCOST.

Description

NPCOST calculates present-value costs associated with nonplant items.

Subprograms Calling This Subroutine

ECON

Subprograms Called by This Subroutine

FUNCTION PVF
SUBROUTINE EVRPT

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/COSTS/	NPC, RATES array, BSLN array, COST array	CSTSUM array

Declarations

None

Input

Common Blocks Only

Output

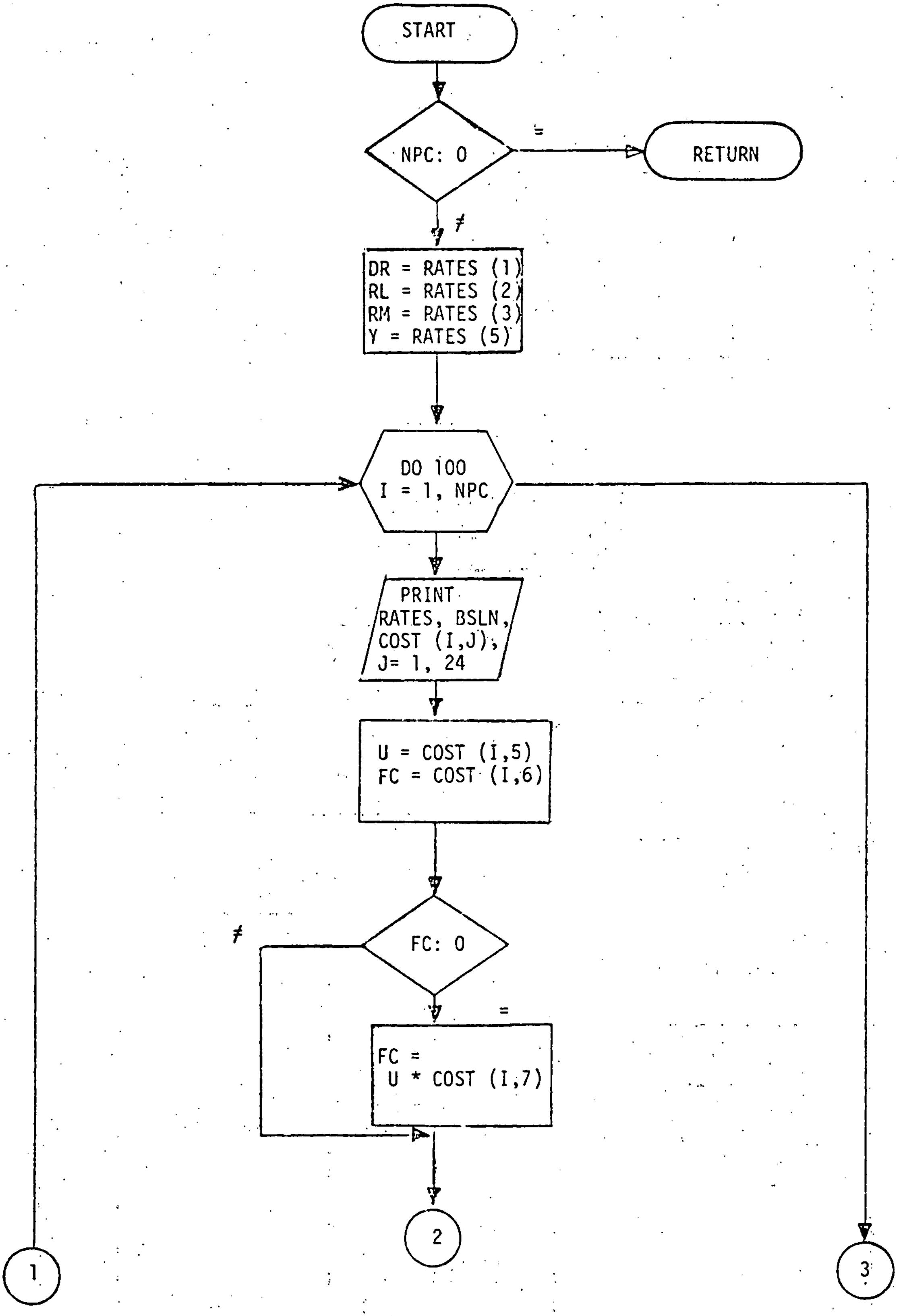
To printer: echo of RATES array, BSLN array, COST array, CSTSUM array

Calculation Procedure

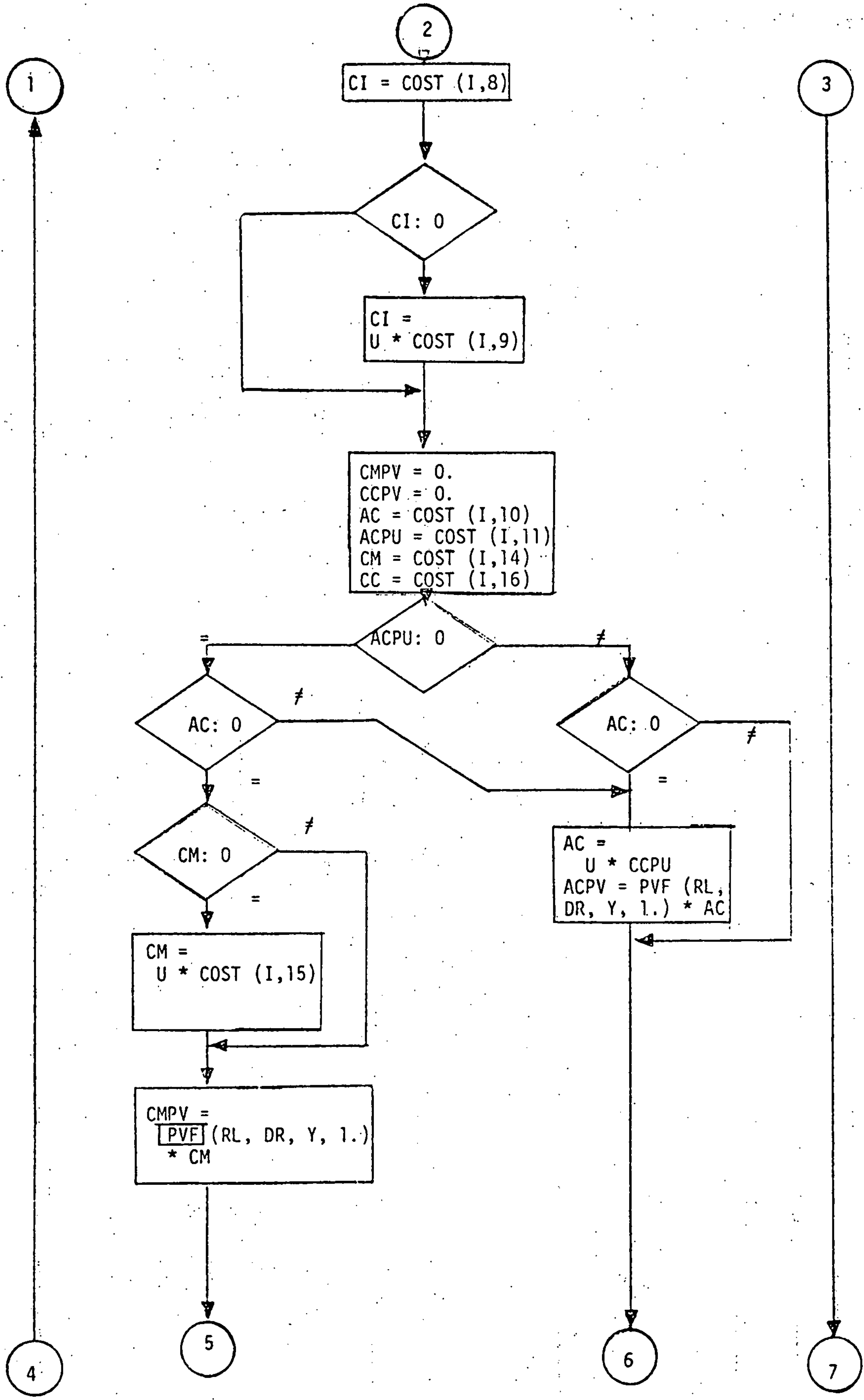
For each nonplant cost item, I = 1 to NPC:

1. Calculate annual cost, cyclical costs, overhaul costs, replacement costs.
2. Calculate yearly costs and present value costs.
3. Assign values to CSTSUM array.
4. Print results in CSTSUM array.

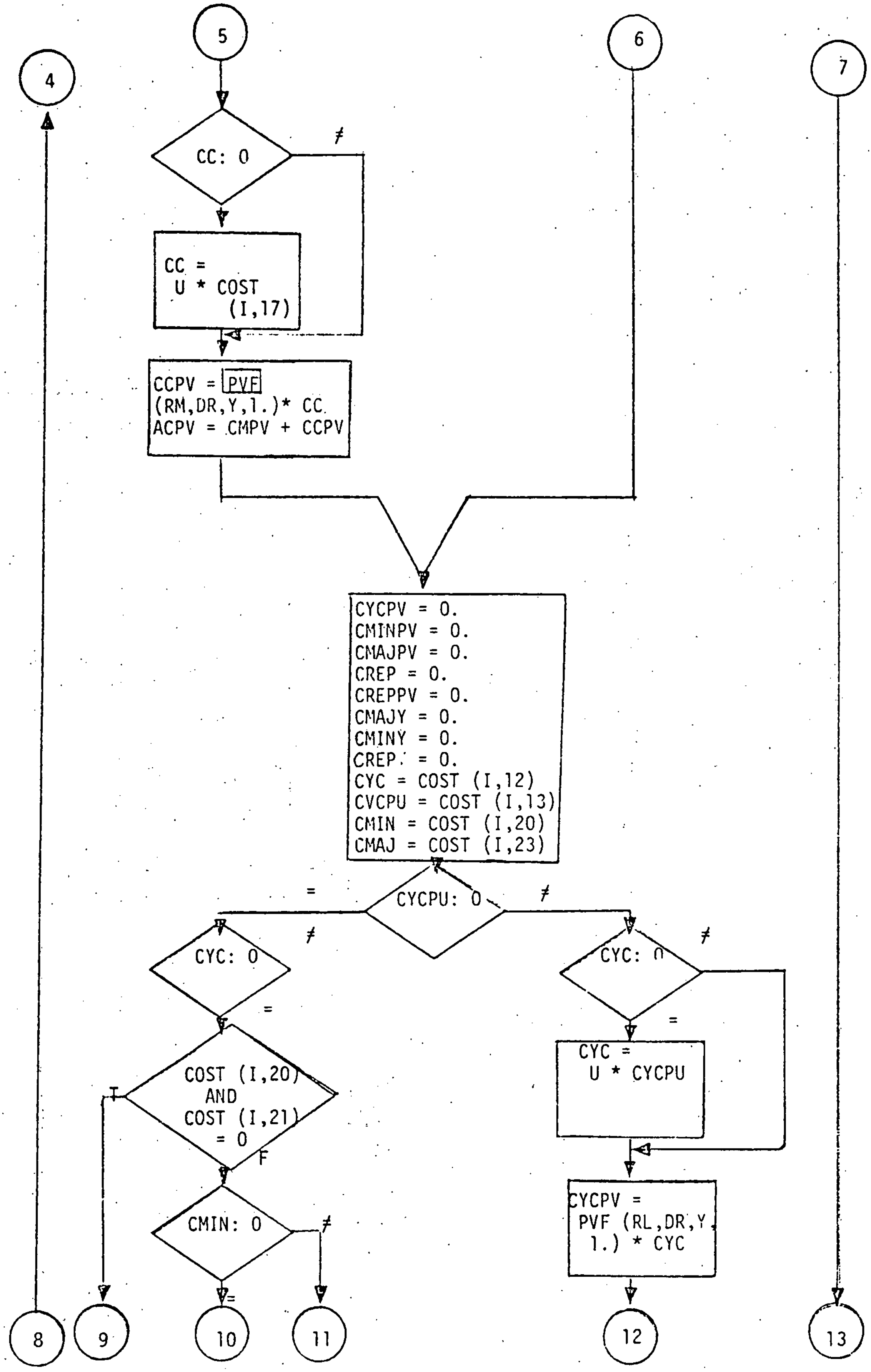
Subroutine NPCOST Flow Chart



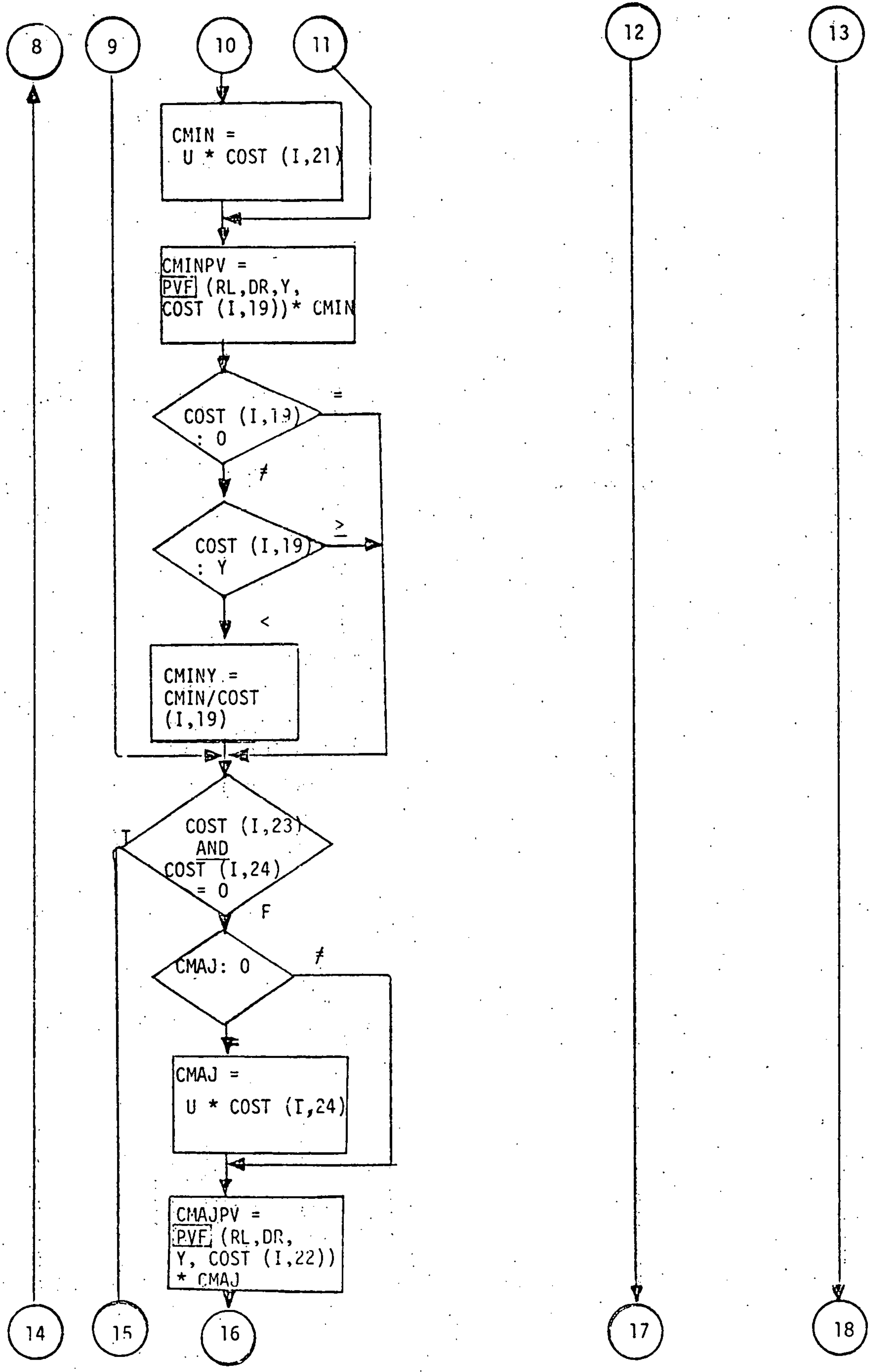
Subroutine NPCOST - Continued



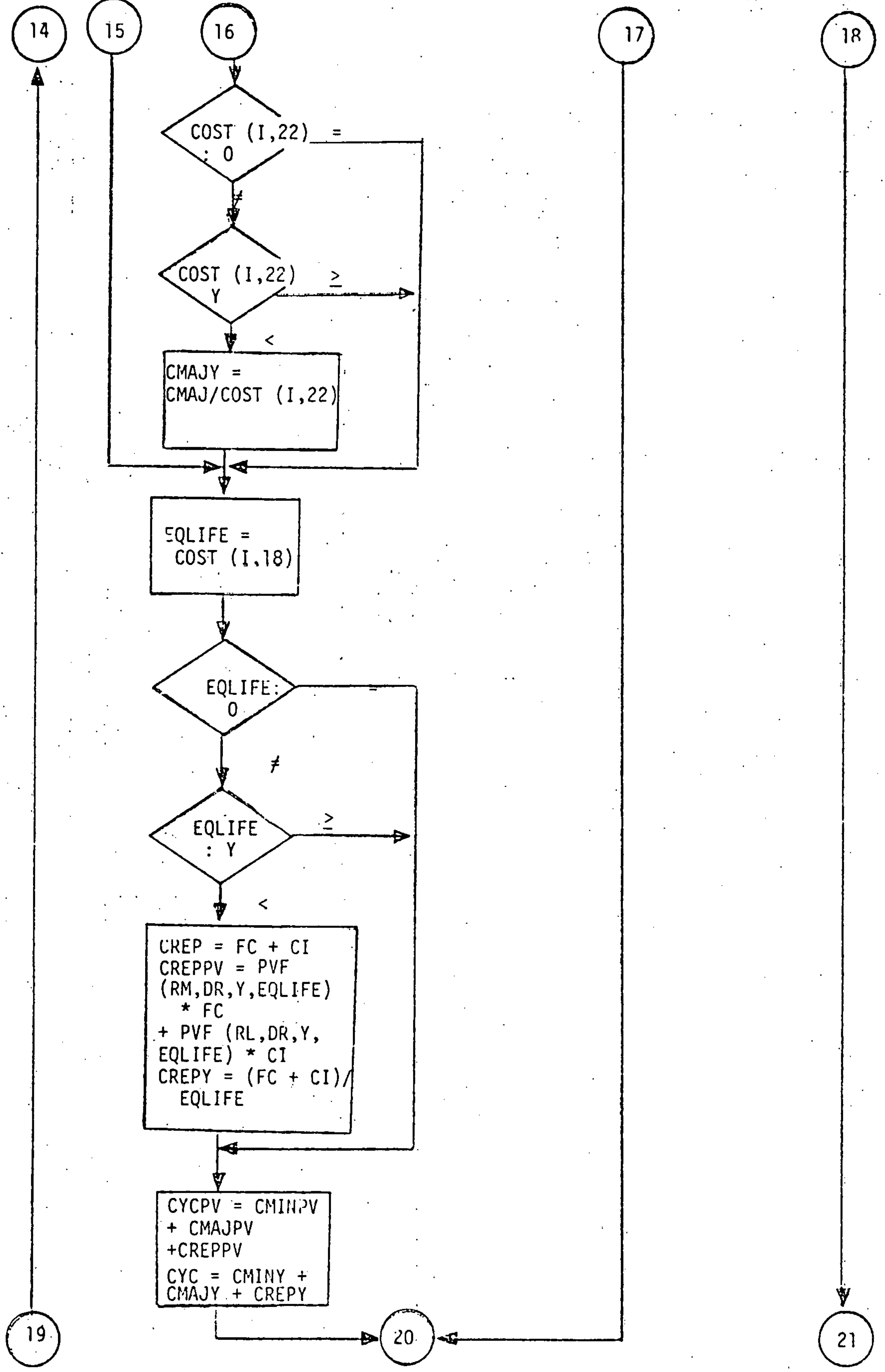
Subroutine NPCOST - Continued



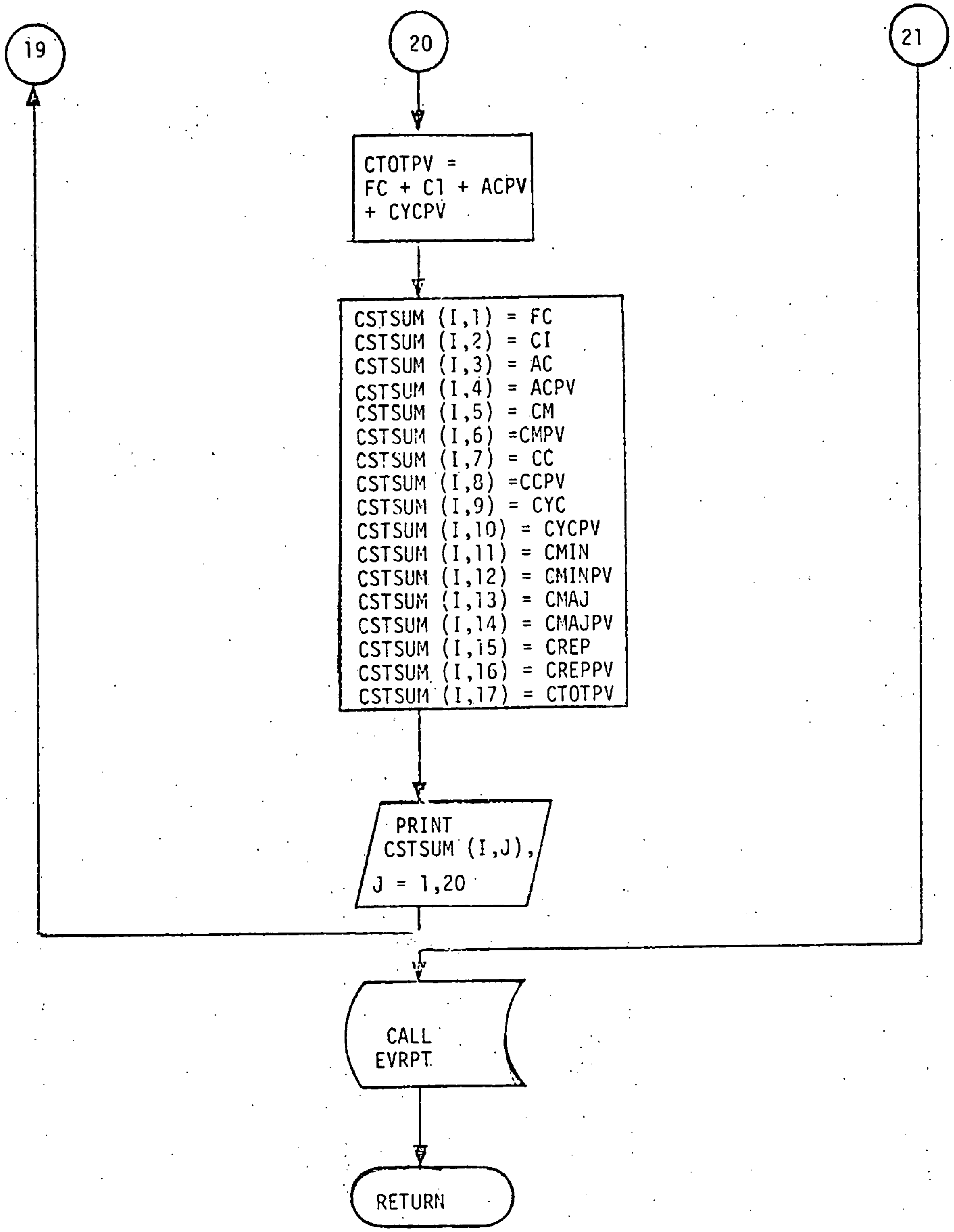
Subrourine NPCOST - Continued



Subroutine NPCOST - Continued


14
15
16
17
18
COST (I,22) : 0
=
≠
COST (I,22) : Y
≥
<
CMAJY = CMAJ/COST (I,22)
EQLIFE = COST (I,18)
EQLIFE : 0
=
≠
EQLIFE : Y
≥
<
CREP = FC + CI
CREPPV = PVF (RM,DR,Y,EQLIFE) * FC + PVF (RL,DR,Y, EQLIFE) * CI
CREPY = (FC + CI)/ EQLIFE
CYCPV = CMINPV + CMAJPV +CREPPV
CYC = CMINY + CMAJY + CREPY
19
20
21

Subroutine NPCOST - Continued



8. Subroutine PORPT.

SUBROUTINE PORPT - INCOMPLETE

Description

This subroutine prints a report of plant costs data.

Subprograms Calling This Subroutine

None

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/FILES/		None
/COSTS/		

Declarations

None

Input

Common Blocks Only

Output

Incomplete

Calculation Procedure

Incomplete

9. Function PVF.

Description

PVF calculates the present value factor given an escalation rate (F), a real discount rate (D), number of years (Y), and compounding interval (T).

Subprograms Calling This Function

SUBROUTINE NPCOST

Subprograms Called by This Function

None

Common Blocks

None

Declarations

None

Input

Name	Description
R	Escalation rate
D	Discount rate
Y	Number of years
T	Compounding interval

Output

Name	Description
PVF	Present value factor returned to calling program

Calculation Procedure

1. Set PVF to zero.
2. Test input variables; if unacceptable, return PVF = 0.0.
3. Calculate intermediate variables NMAX and X.

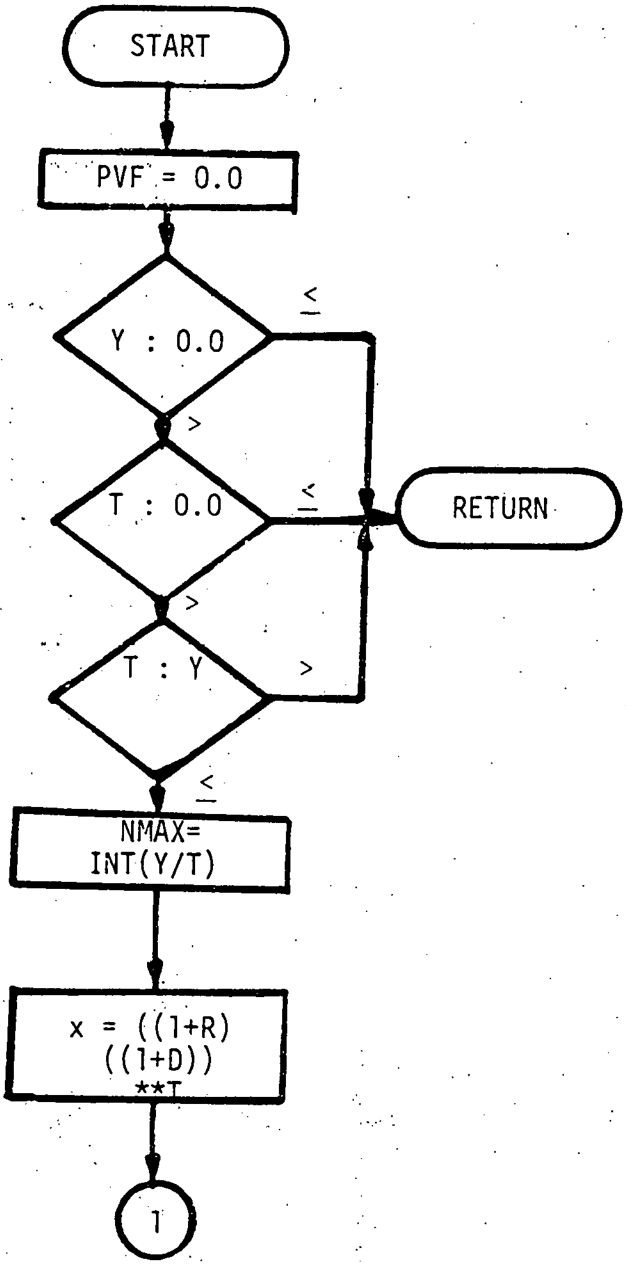
```
NMAX = INT(Y/T)
X = ((1+R)/(1+D))**T
```

4. Calculate present value factor,

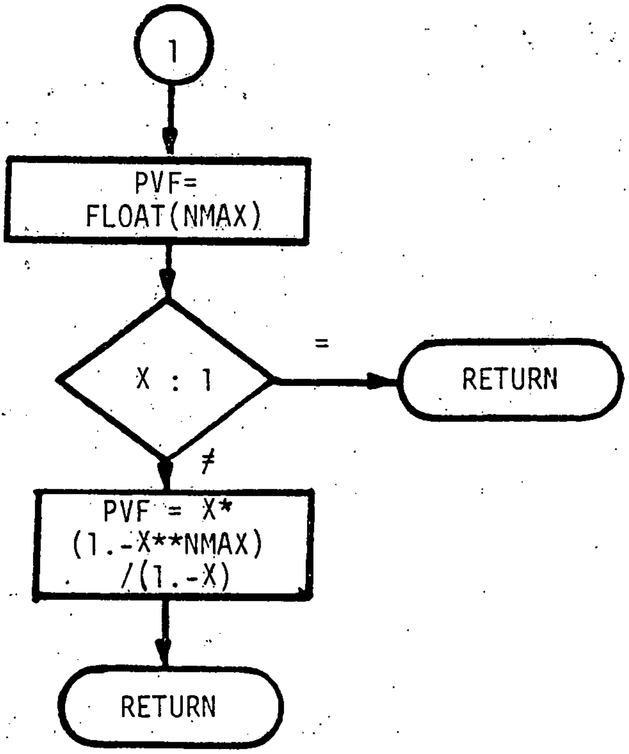
```
PVF = FLOAT (NMAX)
IF (X.EQ.1) RETURN
PVF = X*(1.-X**NMAX)/(1.-X)
```

and return.

Function PVF Flow Chart



Function PVF - Continued



10. Subroutine RDESF.

Description

RDESF reads the nonplant costs standard file and multiplies certain input values by required factors.

Subprograms Calling This Subroutine

ECON

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/FILES/	IESF	None
/COSTS/	None	RATES array, BSLN array, NPC, COST array

Declarations

None

Input

Name	Description
RATES(1) - RATES(10)	Discount rate (DR), labor inflation rate (RL), material inflation rate (RM), energy inflation rate, project life (Y); RATES(6) - BSLN(10) unused
BSLN(1) - BSLN(10)	Plant (nonplant) life-cycle cost, total fuel use, total fuel cost; BSLN(4) - BSLN(10) unused
NPC	Number of nonplant costs
((COST (I,J), I = 1,NPC), J = 1,24)	For each unit (I), cost name (two words), unit name (two words), number of units, first cost, first cost per unit, annual cost, annual cost per unit, cyclical cost, cyclical cost per unit, maintenance cost, maintenance cost per unit, consumables cost, consumables cost per unit, equipment life (yrs), minor overhaul interval (yrs), minor overhaul cost, minor overhaul cost per unit, major overhaul interval (yrs), major overhaul cost, major overhaul cost per unit

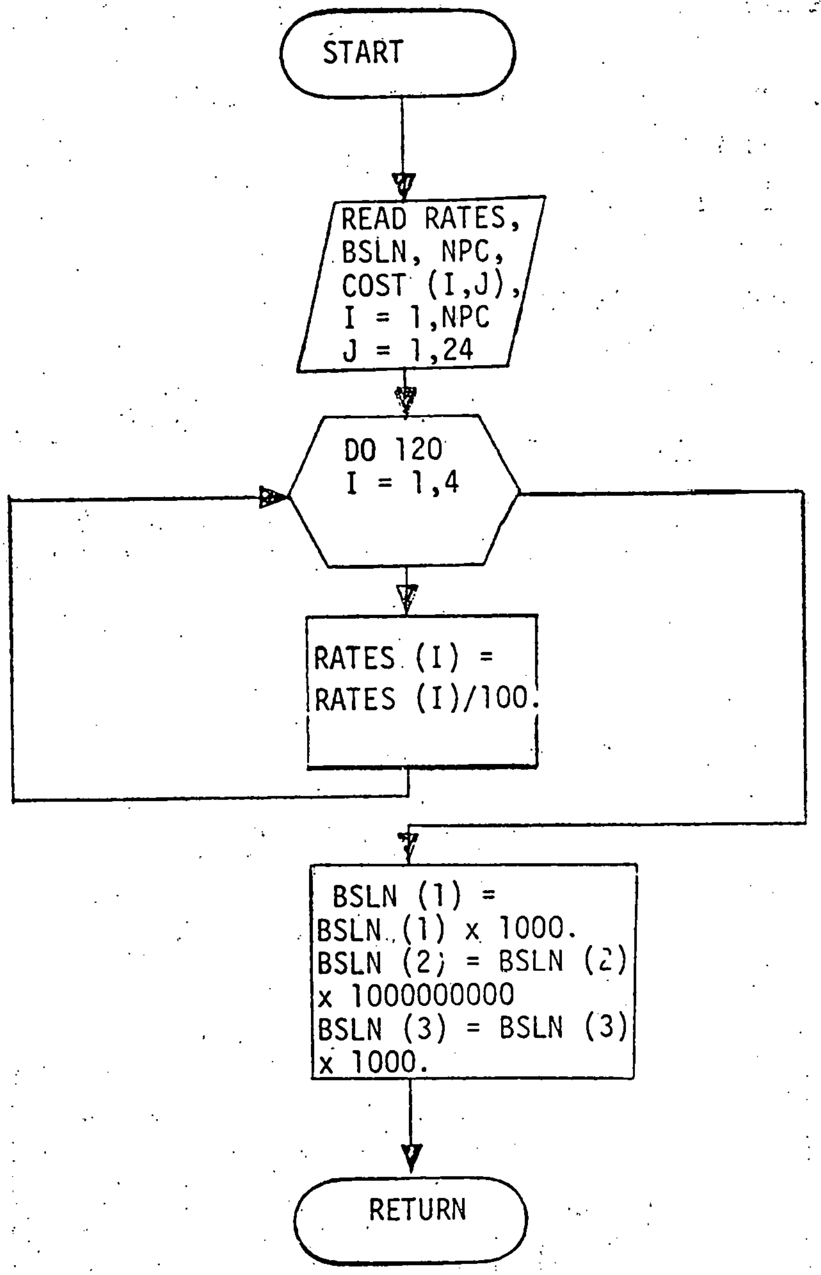
Output

By Common Block Only

Calculation Procedure

1. Read from nonplant costs file RATES, BSLN, NPC, COST.
2. Divide RATES (I) by 100, I = 1,4.
   Multiply BSLN (1) and BSLN (3) by 1000, and BSLN (2) by 10^9.

Subroutine RDESF Flow Chart


START
READ RATES, BSLN, NPC, COST (I,J), I = 1,NPC J = 1,24
DO 120 I = 1,4
RATES (I) = RATES (I)/100.
BSLN (1) = BSLN (1) x 1000.
BSLN (2) = BSLN (2) x 1000000000
BSLN (3) = BSLN (3) x 1000.
RETURN

11. Subroutine RDPCF.

Description

RDPCF reads and echos data from the plant costs standard file and calculates totals of life-cycle equipment costs, utilities, and energy use.

Subprograms Calling This Subroutine

ECON

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/FILES/	IPCF	None
/COSTS/	RATES(5)	TOTECS, TOTUCS, TOTEN
/PLCOST/		NEDATA, NUTLTY, EQHT array, NEQSIZE array, ENUSE array, ENCOST array, UDATA array, ALFCYC array

Declarations

None

Input

Name	Description
NEDATA	Number of energy plant data
NUTLTY	Number of utilities
EQHT array	Equipment costs for each equipment type
NEQSIZE array	Number of sizes of each equipment type
ENUSE array	Utility energy use
ENCOST (1) -ENCOST(10)	Energy cost/utility
UDATA array	Parameters of utility cost
ALFCYC array	Life-cycle parameters

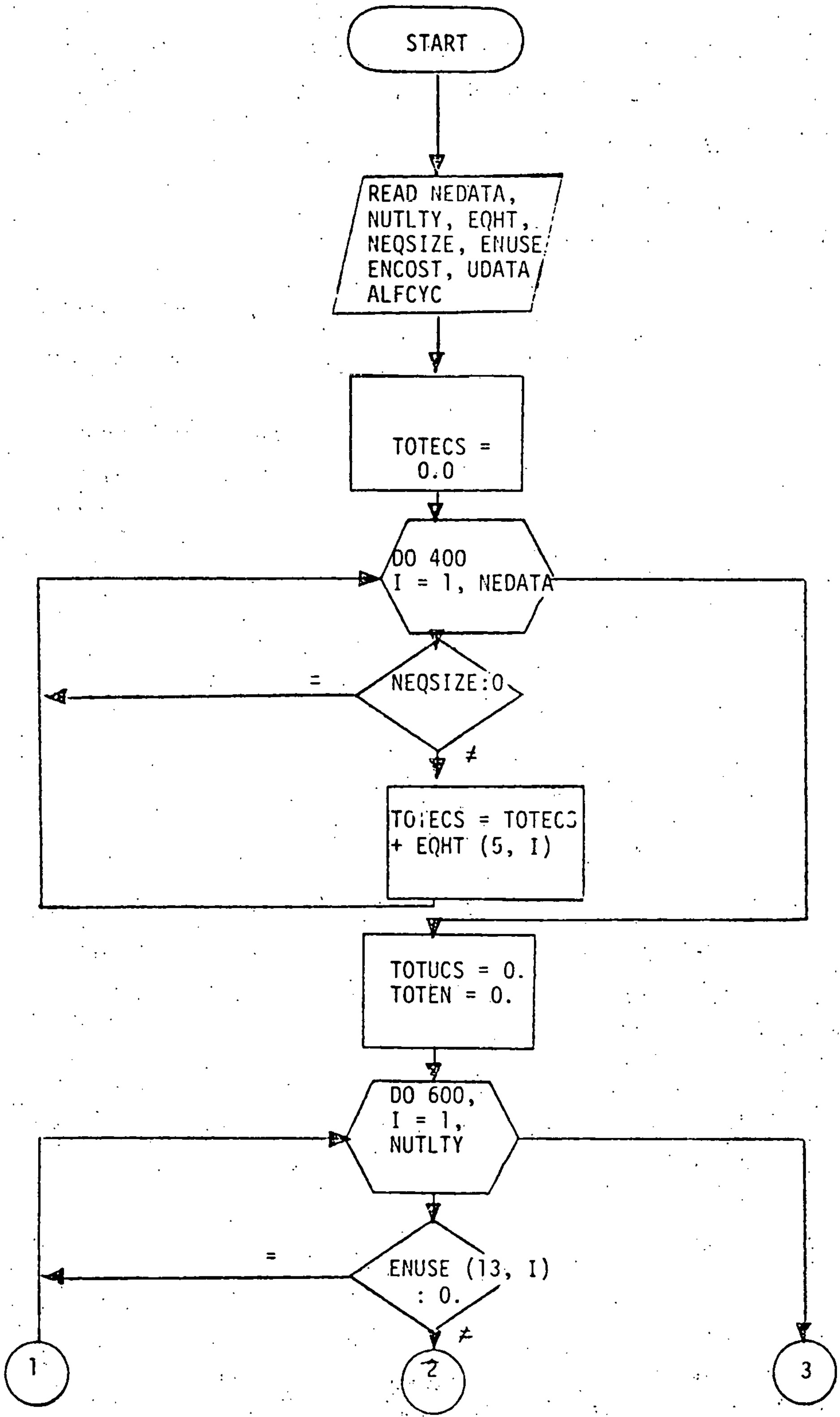
Output

By Common Block Only

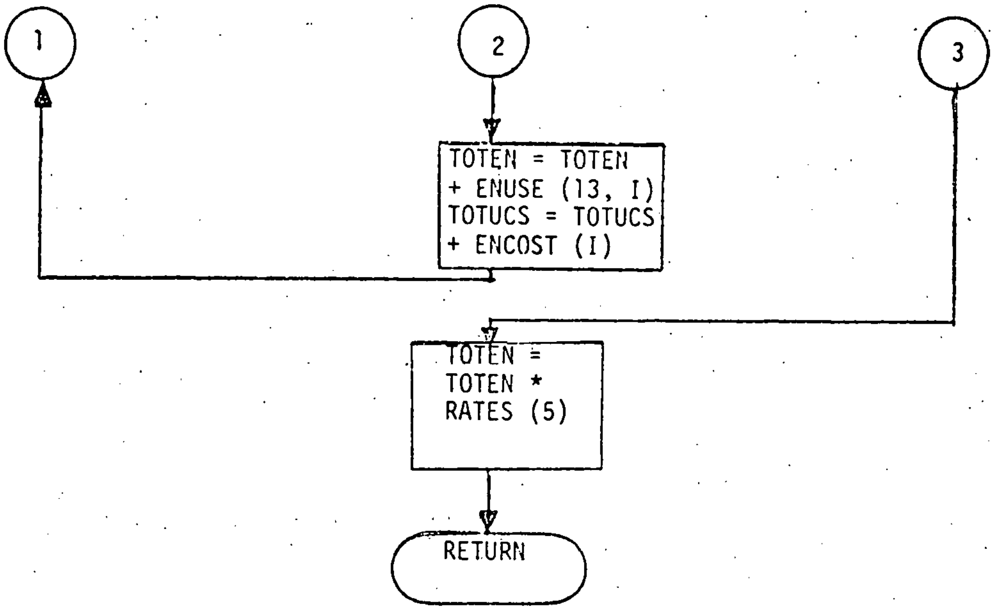
Calculation Procedure

1. Read data from plant-cost file and echo print the data.
2. Calculate TOTECS = the sum of EQHT (5,I), I = 1, NEDATA
   TOTUCS = the sum of ENCOST (I), I = 1, NUTLTY
   TOTEN = RATES (5) * the sum of ENUSE (13, I), I = 1, NUTLTY

Subroutine RDPCF Flow Chart



Subroutine RDPCF - Continued



12. Subroutine R4PRNT*.

Description

The essential purpose of Subroutine R4PRNT is to print a report of life-cycle cost data. R4PRNT also calls two subroutines, COSTEN and COSTEQ, that calculate some cost data for printing.

Subprograms Calling This Routine

ECON*

Subprograms Called by This Routine

Subroutine COSTEN*
Subroutine COSTEQ*
Subroutine RTPRNT*

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/CDPR/*	IPRT	None
/ECDATA/*	EQCHT array, EQCOST array	None
/EDATA/*	NEDATA, NEQSIZE array, IENAME array, CNOM array, KINS array	None
/LFCYCD/*	ALFCYC(5)	None
/STATD/*	ENUSE array, ENPEAK array KENPDY array, KENPHR array	None
/UCOSTD/*	IUNAM array, ENCOST array, UDATA array	None

Declarations

None

Input

Name	Description
IPRT	Unit number for printer output

For a description of the other input variables, EQCHT, EQCOST, NEDATA, NEQSIZE, IENAME, CNOM, KINS, ALFCYC, ENUSE, ENPEAK, KENPDY, KENPHR, IUNAM, ENCOST, and UDATA, see Chap. V.E.2, CCBTEPS.

* In the October 1, 1977, version of Cal-ERDA, subroutines R4PRNT, COSTEN, COSTEQ, and RTPRNT are found in the PLANT Program. In subsequent versions, however, the first three and a copy of the fourth will be moved to the ECON program, and all required data from PLANT will be sent in the plant costs file to ECON.

Output

The input variables are output to the printer. In addition, three variables are calculated internally and are printed.

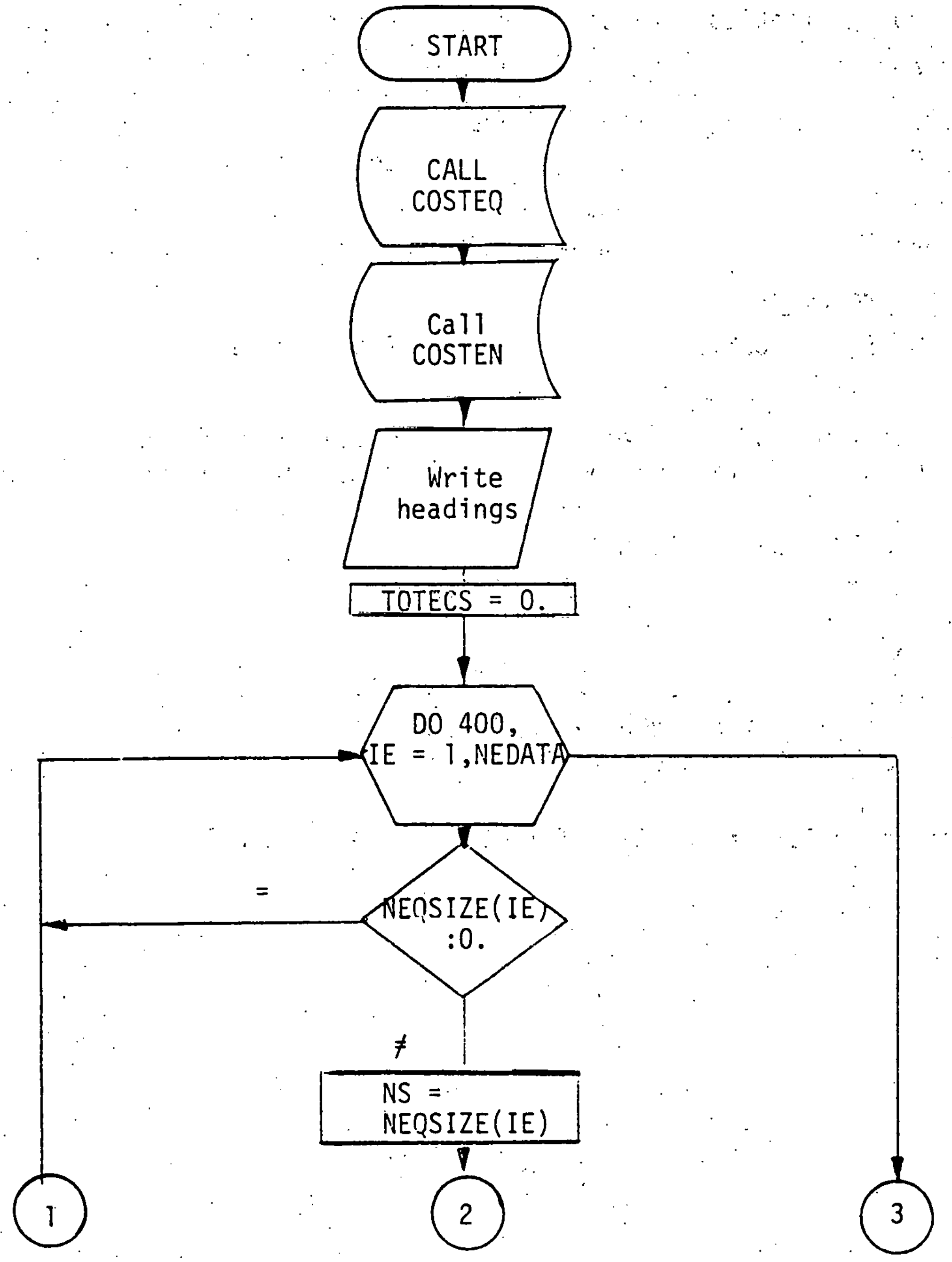
Name	Description
TOTECS	Total equipment cost
TOTUCS	Total cost of energy
TOTCOST	Sum of TOTECS and TOTUCS

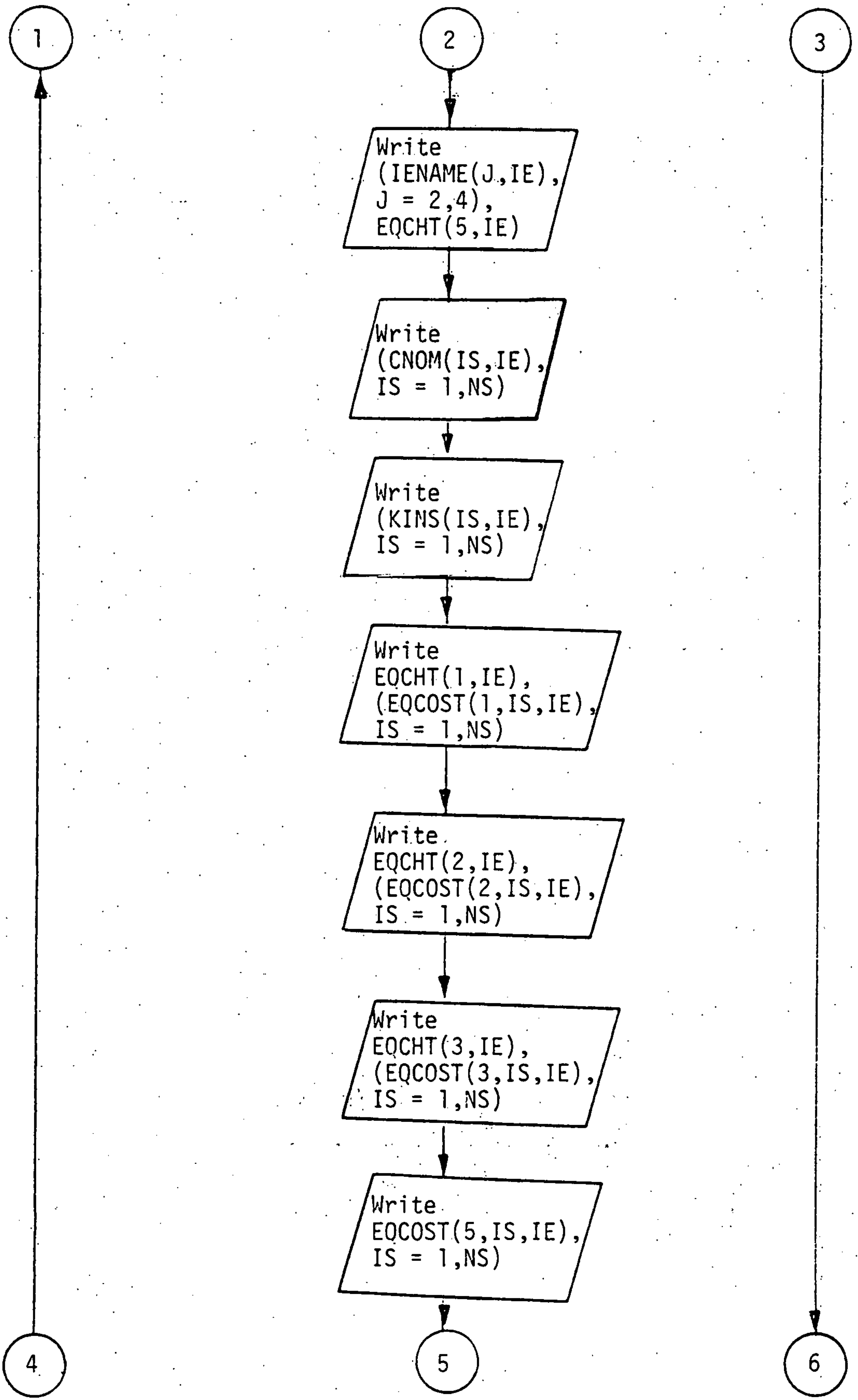
Calculation Procedure

1. Call COSTEN and COSTEQ for calculation of cost data.
2. Write headings. Set TOTECS to zero.
3. Loop over types of equipment, I, printing IENAME and EQCHT(5,I), and for each size, J, of each type of equipment, CNOM, KINS, EQCHT, and ENCOST.

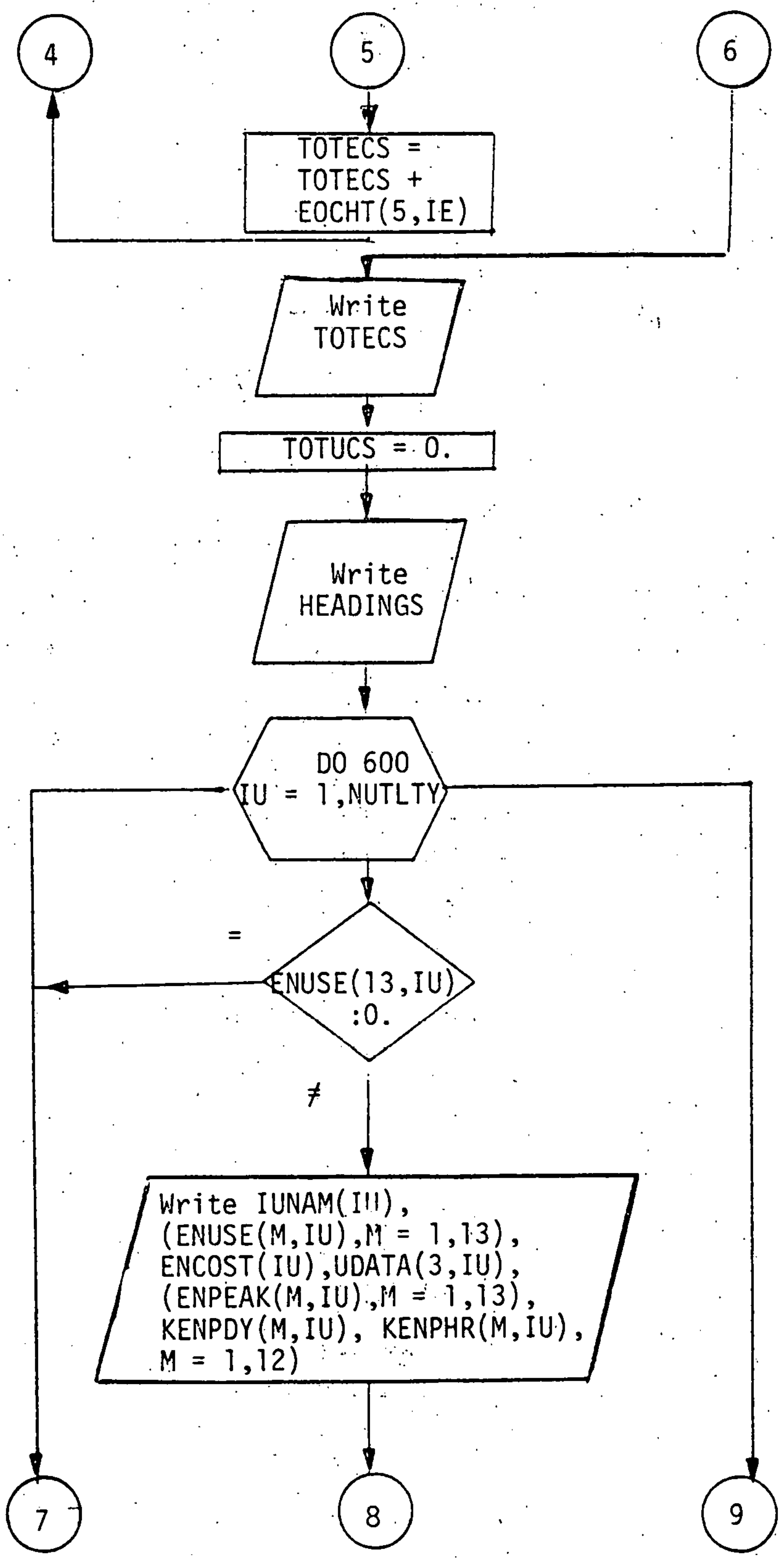
   Calculate TOTECS = TOTECS + EQCHT(5,I)
4. Write TOTECS.
   Set TOTUCS to zero.
   Print headings.
5. For each utility, print IUNAM and monthly and annual values of ENUSE, ENCOST, UDATA, ENPEAK, KENPDY, KENPHR.
   Calculate TOTUCS = TOTUCS + ENCOST (NUTLTY).
6. Write TOTUCS.
   Calculate TOTCOST = TOTECS + TOTUCS and IY = ALFCYC(5).
   Write IY and TOTCOST.
7. Call RTPRNT; then return.

Subroutine R4PRNT Flow Chart

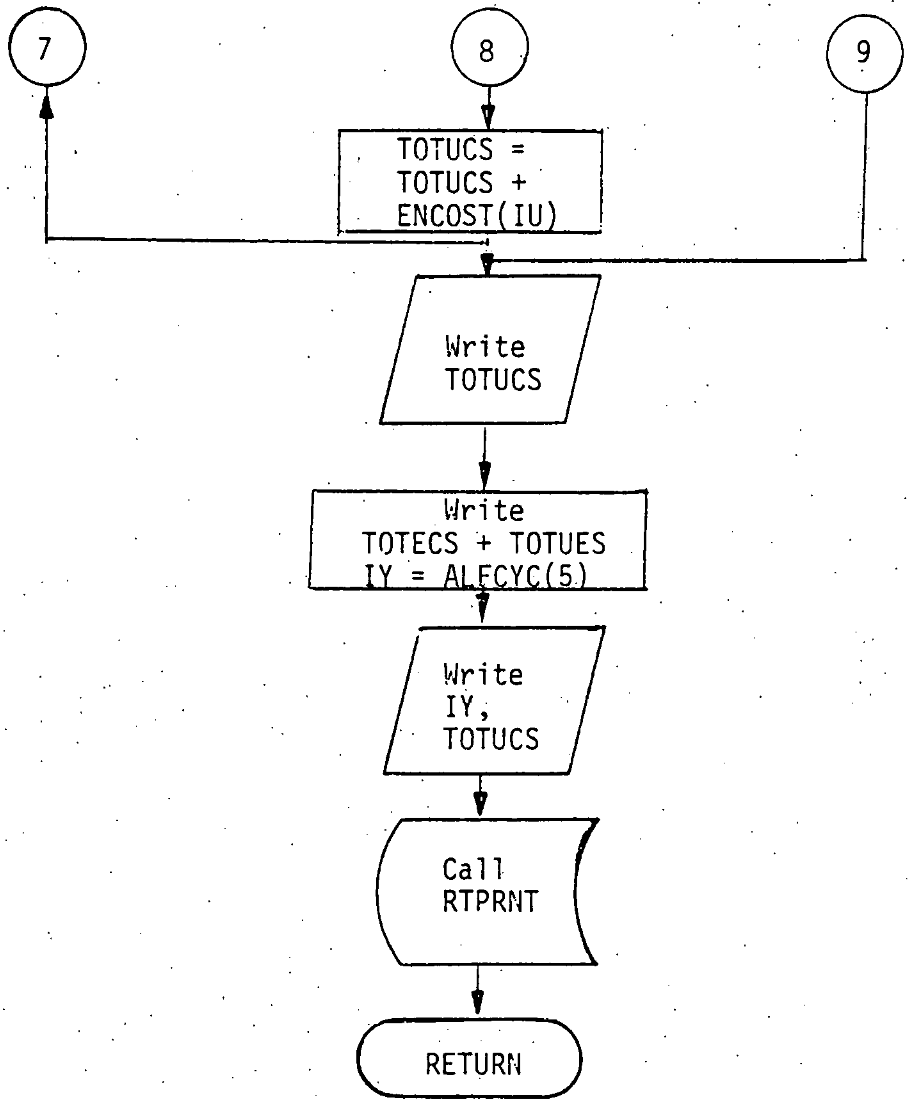



1
2
3
Write
(IENAME(J,IE),
J = 2,4),
EQCHT(5,IE)
Write
(CNOM(IS,IE),
IS = 1,NS)
Write
(KINS(IS,IE),
IS = 1,NS)
Write
EQCHT(1,IE),
(EQCOST(1,IS,IE),
IS = 1,NS)
Write
EQCHT(2,IE),
(EQCOST(2,IS,IE),
IS = 1,NS)
Write
EQCHT(3,IE),
(EQCOST(3,IS,IE),
IS = 1,NS)
Write
EQCOST(5,IS,IE),
IS = 1,NS)
4
5
6

Subroutine R4PRNT - Continued



Subroutine R4PRNT - Continued



7
8
9
TOTUCS = TOTUCS + ENCOST(IU)
Write TOTUCS
Write TOTECS + TOTUES IY = ALECYC(5)
Write IY, TOTUCS
Call RTPRNT
RETURN

13. Subroutine SCOST.

Description

SCOST calculates cost savings and fuel savings over the baseline case.

Subprograms Calling This Subroutine

ECON

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/COSTS/	BSLN array, TOTCST array, RATES array	SAV array

Declarations

None

Input

Common Block Only

Output

Name	Description
TOTCST (1) - TOTCST (20)	Sums of first costs, installation costs, present value annual costs, present value cyclical costs, sum of these four; sums of equipment cost, utilities cost, energy use, sum of these three; sums of present value annual maintenance costs, present value annual consumables costs, present value minor overhaul costs, present value major overhaul costs, replacement costs; the last six elements are unused
BSLN (1) - BSLN (10)	Baseline plant life-cycle cost, total fuel use, total fuel cost; the last seven elements in the array are unused
SAV (1) - SAV (6)	Cost savings, investment, savings/investment ratio, Btu savings, Btu savings/investment ratio, discounted payback period

Calculation Procedure

1. Calculate cost savings and investment.

   SAV (1) = BSLN (1) - TOTCST (6) + BSLN (3) - TOTCST (7)
   SAV (2) = TOTCST (5)

2. Calculate savings to investment ratio (SIR).

   SAV (3) = SAV (1)/SAV (2)

3. Calculate the Btu savings.

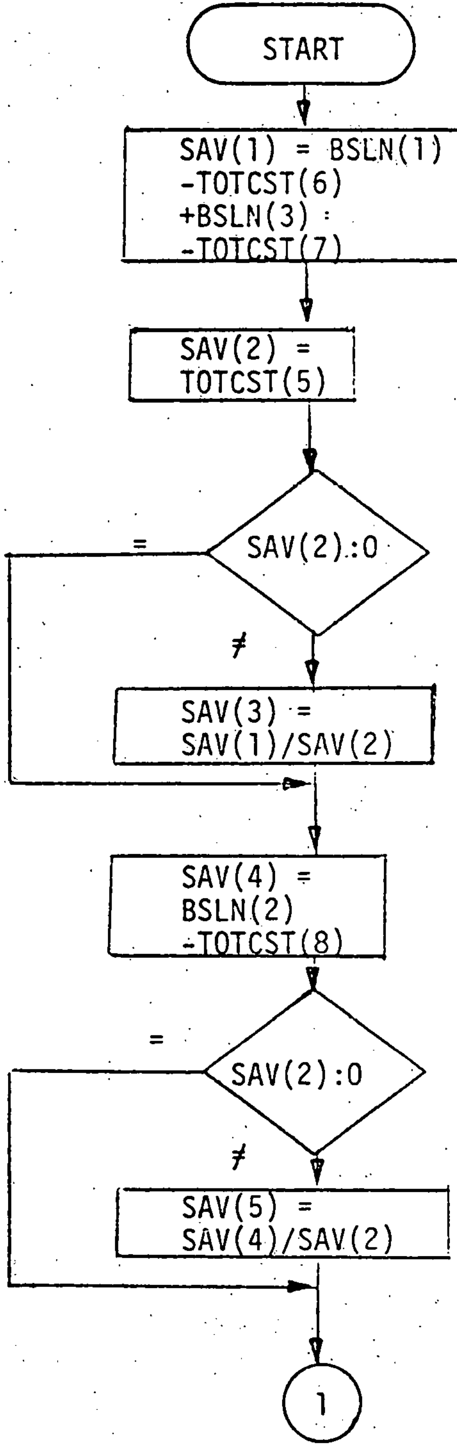
   SAV (4) = BSLN (2) - TOTCST (8)

4. Calculate Btu savings to investment ratio.

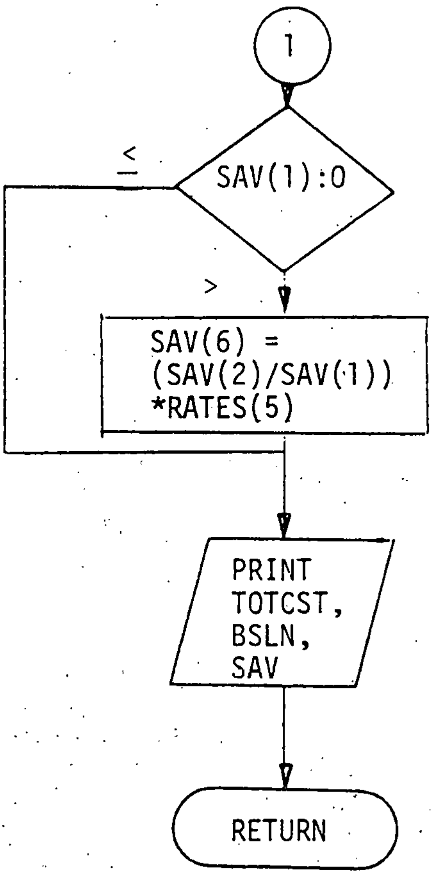
   SAV (5) = SAV (4)/SAV (2)

5. Calculate discounted payback period (number of years for saving to equal investment). If SAV (1), cost savings, = 0, skip the calculation. Otherwise, SAV (6) = (SAV (2)/SAV (1)) * RATES (5).

Subroutine SCOST Flow Chart



Subroutine SCOST - Continued



14. Subroutine TCOST.

Description

Total plant and nonplant costs are calculated by TCOST.

Subprograms Calling This Subroutine

ECON

Subprograms Called by This Subroutine

None

Common Blocks	Variables Obtained from Common Blocks	Variables Placed in Common Blocks
/COSTS/	CSTSUM array, TOTECS, TOTUCS, TOTEN	TOTCST array

Input

Common Block Only

Output

Name	Description
To printer, TOTCST (1) - TOTCST (20)	Sums of first costs, installation costs, present value annual costs, present value cyclical costs, sum of these four; sums of equipment cost, utilities cost, energy use, sum of these three; sums of present value annual maintenance costs, present value annual consumable costs, present value minor overhaul costs, present value major overhaul costs, replacement costs; the last six elements are unused

Calculation Procedure

1. If NPC = 0, i.e., there are no nonplant costs, skip to Step 4.

2. For I = 1 to NPC

TOTCST (1) = the sum of CSTSUM (I,1), first costs

TOTCST (2) = the sum of CSTSUM (I,2), present value of annual costs

TOTCST (3) = the sum CSTSUM (I,4), present value of annual costs

TOTCST (4) = the sum of CSTSUM (I,10), present value of cyclical costs,

TOTCST (10) = the sum of CSTSUM (I,6), present value of annual maintenance costs

TOTCST (11) = the sum of CSTSUM (I,8), present value of annual consumables costs

TOTCST (12) = the sum of CSTSUM (I,12), present value of minor overhaul costs

TOTCST (13) = the sum of CSTSUM (I,14), present value of major overhaul costs

TOTCST (14) = the sum of CSTSUM (I,16), present value of replacement costs

3. TOTCST (5) = is the sum of TOTCST (1) - TOTCST (4).

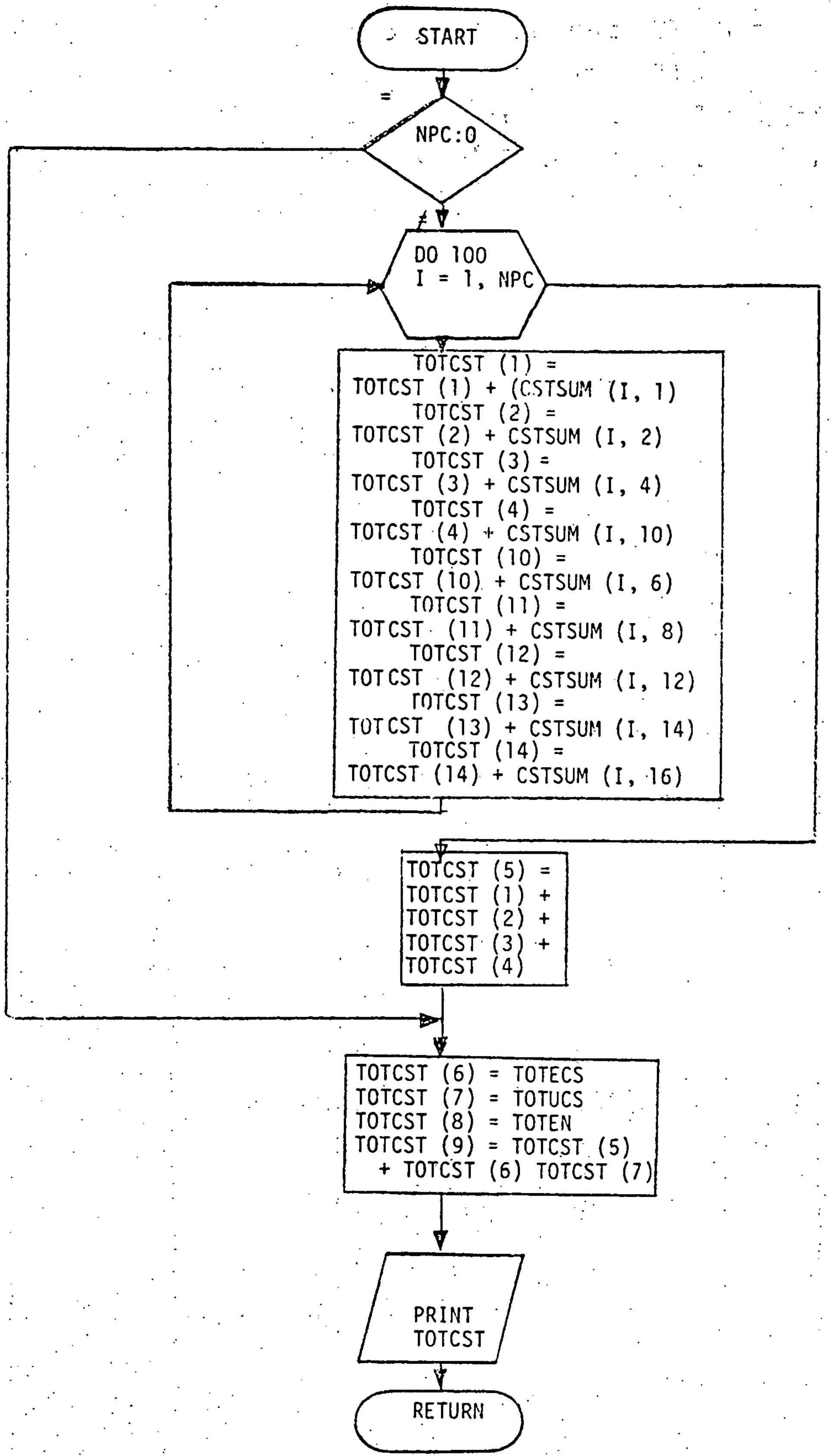
4. TOTCST (6) = TOTECS, total equipment cost

   TOTCST (7) = TOTUCS, total utilities cost

   TOTCST (8) = TOTEN, total energy cost

   TOTCST (9) = the sum of TOTCST (5) - TOTCST (7)

Subroutine TCOST Flow Chart



VII. REPORT PROGRAM

This write-up will be provided in later editions of this manual.

VIII. WEATHER PROGRAMS

The constants and format of both the set of Cal-ERDA weather data processing programs and the weather file library are described in the Cal-ERDA Users Manual. Details of the file and data structures used, the calling sequence, and subroutine descriptions will be provided in later editions of the Program Manual.

IX. UTILITY PROGRAMS

A. EXECUTIVE Programs

The EXECUTIVE Program provides the interface between the Cal-ERDA Program and the computer. Figure I.2, Chap. I, shows this relationship in the Cal-ERDA configuration chart. EXECUTIVE sets up the control cards in proper sequence for calling the several Cal-ERDA Programs. It also attaches, saves, catalogs, and rewinds permanent files, executes termination, controls input/output, processes card images, and reads file directory tables. Because it is machine specific, both a CDC and IBM version exist.

1. CDC Executive. Since the CDC EXECUTIVE has not been implemented on the LBL CDC 7600 as of October 1, 1977, a write-up is not included here. Such a write-up will be provided in later editions of the manual.

2. IBM EXECUTIVE. ANL is converting Cal-ERDA so that it can be executed on IBM computers. ANL will provide the IBM EXECUTIVE write-up at a later date; it will be included in subsequent editions of this manual.

X. LIBRARY DATA*

A. Building Description (Loads) Data

B. Systems Data

C. Plant Data

D. Economics Data

E. Format, Structure, and Access to Each File

[*] This write-up will be provided in later editions of this manual.

XI. REFERENCES

1. "Procedure for Determining Heating and Cooling Loads for Computerizing Energy Calculations. Algorithms for Building Heat Transfer Subroutines," Energy Calculations 1, 1975, ASHRAE Task Group on Energy Requirements for Heating and Cooling of Buildings, American Society of Heating, Refrigerating, and Air-Conditioning Engineers, Inc., 345 East 47th Street, New York, NY 10017 (also appearing as a first printing, edited by M. Lokmanhekin, 1971).

2. "Procedure for Simulating the Performance of Components and Systems for Energy Calculations," Energy Calculations 2, edited by W. F. Stoecker, 1975, ASHRAE Task Group on Energy Requirements for Heating and Cooling of Buildings, American Society of Heating, Refrigerating, and Air-Conditioning Engineers, Inc., 345 East 47th Street, New York, NY 10017.

3. "NECAP, NASA's Energy-Cost Analysis Program," Robert H. Henninger, Editor, September 1975, NASA Contractor Report NASA CR-2590, Part I Users Manual and Part II Engineering Manual. Available from the National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161, as Report N76-10751 ($8.50) and N76-10752 ($9.50).

4. "Life Cycle Costing, Emphasizing Energy Conservation," Energy Research and Development Administration Report ERDA 76/130, September 1976 (Revised May 1977). Available from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Road, Springfield, VA 22161 ($6.00 for printed copy).

5. 1972 ASHRAE Handbook of Fundamentals, American Society of Heating, Refrigerating, and Air-Conditioning Engineers, Inc., 345 East 47th Street, New York, NY 10017.

6. "Energy Conservation Design Manual for New Non-Residential Buildings," in preparation by the State of California, Energy Resources Conservation and Development Commission, 1111 Howe Avenue, Sacramento, CA 95825.

7. D. G. Stephenson and G. P. Mitalas, "Cooling Load Calculations by Thermal Response Factor Method," ASHRAE Semiannual Meeting, Detroit, Michigan, January 20-February 2, 1967, Paper No. 2018.

8. G. P. Mitalas and D. G. Stephenson, "Room Thermal Response Factors," ASHRAE Transactions, pp. III 2.1-2.10, Part I (1967).

9. G. P. Mitalas and J. G. Arseneault, "Fortran IV Program to Calculate Heat Flux Response Factors for a Multi-Layer Slab," National Research Council of Canada (June 1967).

10. "Summary of Solar Radiation Observations," Boeing Company Report D2-90577-1 (December 1964).

11. Collected Algorithms from CACM, Association for Computing Machinery, Vol. 1, Algorithm 199.

12. J. Duffie and W. Beckman, Solar Energy Thermal Processes (Wiley-Interscience, New York, 1974).

13. J. L. Threlkeld, Thermal Environmental Engineering, 2nd edition (Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1970).

14. ASHRAE Handbook and Product Directory, Equipment (1975).

15. S. A. Klein, I. I. Cooper, T. L. Freeman, D. M. Beekman, W. A. Beckman, and J. A. Duffie, "A Method of Simulation of Solar Processes and Its Application," Solar Energy 17, pp. 29-37 (1975).

16. B. Y. H. Liu and R. C. Jordan, "The Interrelationship and Characteristic Distribution of Direct, Diffuse, and Total Solar Radiation," Solar Energy 4, pp. 1-19 (1960).

17. "Life Cycle Costing Emphasizing Energy Conservation," ERDA Division of Facilities and Construction Management report ERDA-76/130 (September 1976).