MICROPROCESSOR-BASED THERMAL CONTROL UNIT

A Thesis

Presented to

the Faculty of the

Department of Electrical Engineering

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

by

Kam Kong Pang

December, 1978

ABSTRACT

This thesis describes all conceptual aspects related to the design of a microprocessor based data acquisition and control system built for use in a thermal energy conservation study at the University of Houston. A comprehensive description of the system requirements, functional design, software organization, and hardware architecture is presented. In addition, a system performance evaluation and improvement recommendations based on three months of continuous operation are also discussed.

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (Cont.)

CHAPTER 1

INTRODUCTION

This thesis presents all conceptual aspects related to the design of a microprocessor-based data acquisition and control system built for use in a thermal energy conservation study conducted at the University of Houston. The focus of the presentation will be on the system rather than on the study for which it was designed.

Since the occurrence of an energy crisis, there has been an increasing awareness of the importance of energy conservation. Figure 1-1 shows a distribution of energy allocation for various purposes[1]. From the graph, it can be seen that energy used in buildings -for space heating and cooling, water heating, and lighting - accounts for 30 percents of the total national energy consumption.

Recently, considerable amount of work had been done in the area of energy conservation by computerized automation of existing heating, ventilation and air-conditioning (HVAC) systems,especially in the northern part of the country where reduction of thermal energy consumption is the main concern. Snively[2] suggested several possible energy saving applications for computerized automation in order to bring new awareness and understanding to the problem. Spethmann[3] summarized some specific energy conservation methods; such as running equipment only when needed, sequencing heating and cooling as much as possible, and using load reset to

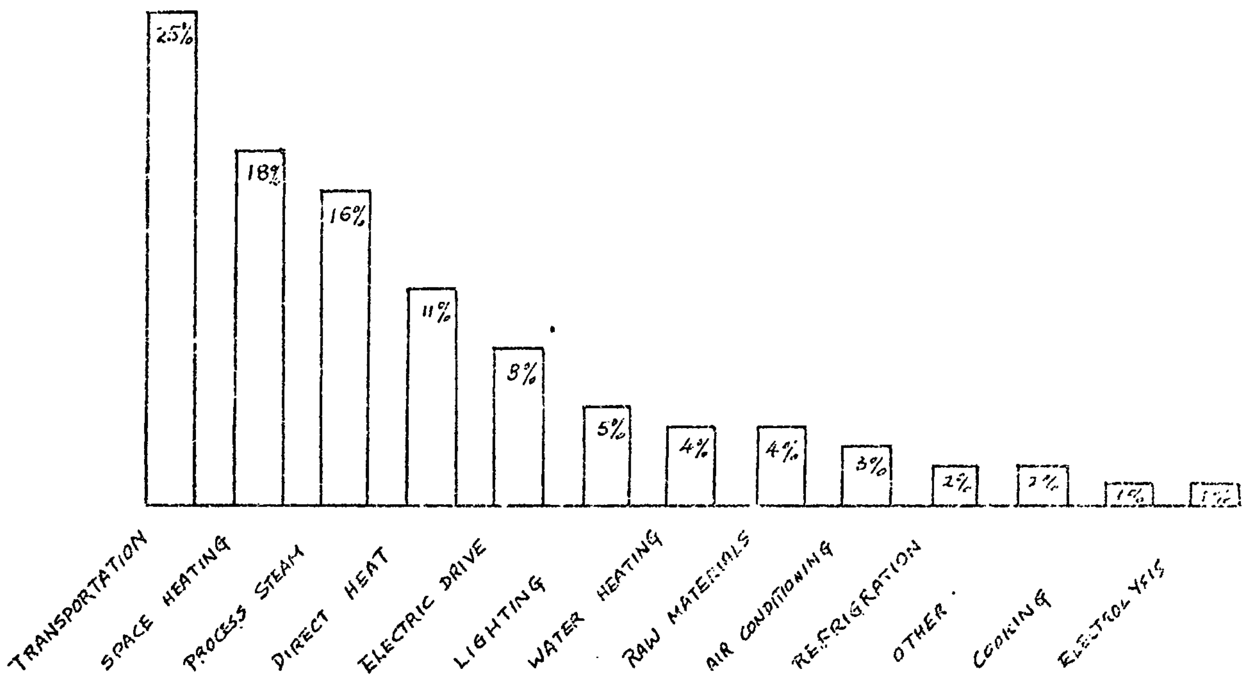


Figure 1-1 Energy allocation distribution in the U.S.A.

improve efficiency of all heating and cooling supply system, etc.; as discussed at a number of ASHRAE Forums (e.g. Louisville July, 1973; Montreal July, 1974). Shavit[4] combined the energy conservation methods as suggested by Snively[5] and Spethmann[6] with relaxing space temperature by widening the thermostat throttling range, and reported significant saving on computer simulation.

Several commercial energy management systems are available for conducting the study, e.g. Power Co.'s S170, Honeywell's TDC-2000, IBM's system/7[7], Johnson Controls' JC/80, Barber Colman Co.'s ECON-VI, and Robertshaw's digital management system 2400, ect. However, most of these systems are mini-computer based which makes them costly, large in size, and unreliable as compared to distributed systems.

The design of microprocessor-based data acquisition and control systems has become the topic of discussion in many recent papers. Weissberger[8] mentioned its advantage of cutting cost, cutting size, increasing reliability, and increasing flexibility. Reed and Mergler[9] described the architecture of a general purpose microprocessor based industrial process controller. Shih[10] suggested a way of cutting cost of a distributed automation system by employing addressable sensing devices. The system built for the University of Houston study is a microprocessor-based, stand-alone intelligent remote terminal that has the advantages of a microprocessor based system and provides all the capabilities of presently available minicomputer-based automation systems.

This thesis emphasizes the description of the overall design concepts of the system. Chapter 2 defines the automation system requirements as design requirements and operational requirements. Chapter 3 describes the functional design of the system as specified by the operational requirements, as well as a user language written for operator communication. Chapter 4 implements the functional design by a number of user tasks which are co-ordinated by a real-time operating system, the inter-task communication is also explained. Chapter 5 completes the description of the functional design by presenting the hardware architecture of the system which supports the software tasks, the communication between the software tasks and the hardware modules is also included. Chapter 6 evaluates the system operation, discusses system improvement recommendations, and provides the conclusion of the thesis.

A complete description of the system operation, the hardware modules, the hardware field equipments, and the software modules is presented in the Thermal Control Unit (TCU) manuals[11-15].

CHAPTER 2

SYSTEM REQUIREMENTS

A system design must meet two basic sets of requirement: design requirements which relate to the design approach and operational requirements which relate to the system performance.

2.1 Design Requirements

The basic design requirement for the Thermal Control Unit(TCU) is that it should reflect the state of the art, be reliable, be modular in both software and hardware, and have simple implementation so that other schools may use the design for their own implementations.

2.1.1 Reliability

The TCU should be designed in a manner that it may be used as an element of a distributed system under the control of a master processor. In addition, stand-alone capability should be incorporated into the design so that it will carry out the pre-programmed activities even in the event of a failure of the Central Control Unit (CCU). The TCU should also operate under the supervision of a terminal such as a teletype. Raw information acquired by the TCU should be processed locally so that only reduced data is transmitted to the CCU. This approach will decrease the communication traffic between the TCU and the CCU, and thus improve its reliability, and decrease the processing load of the CCU.

Programs for the TCU should be stored in firmware that is unaffected by loss of power, and the system should be capable of self-initialization when power is restored. Lastly, default operation of all system functions should be established so that normal operation of the HVAC system is possible when the TCU is isolated from the rest of the system.

2.1.2 Modularity

The software and hardware designs of the TCU should be implemented as functional modules, i.e. functionally oriented tasks and functionally oriented cards. This design approach simplifies system expansion, testing, and integration. Also, the parameters of all the tasks should be stored in a common memory area so that system parameters may be changed without affecting the system programs.

2.1.3 Reproducibility

The TCU should be a simple, inexpensive and well documented system so that it may be easily reproduced by the staff of other schools and universities. In order to accomplish this goal, the TCU should be designed with readily available sub-system components. The sub-systems should have built-in diagnostic routines to simplify testing.

2.2 Operational Requirements

The TCU must provide all data acquisition and control features available in conventional automation systems to support the measurement and control of variables associated with the HVAC system. A block diagram of the system interfaces is shown in Figure 2-1.

2.2.1 Data Acquisition Capability Requirements

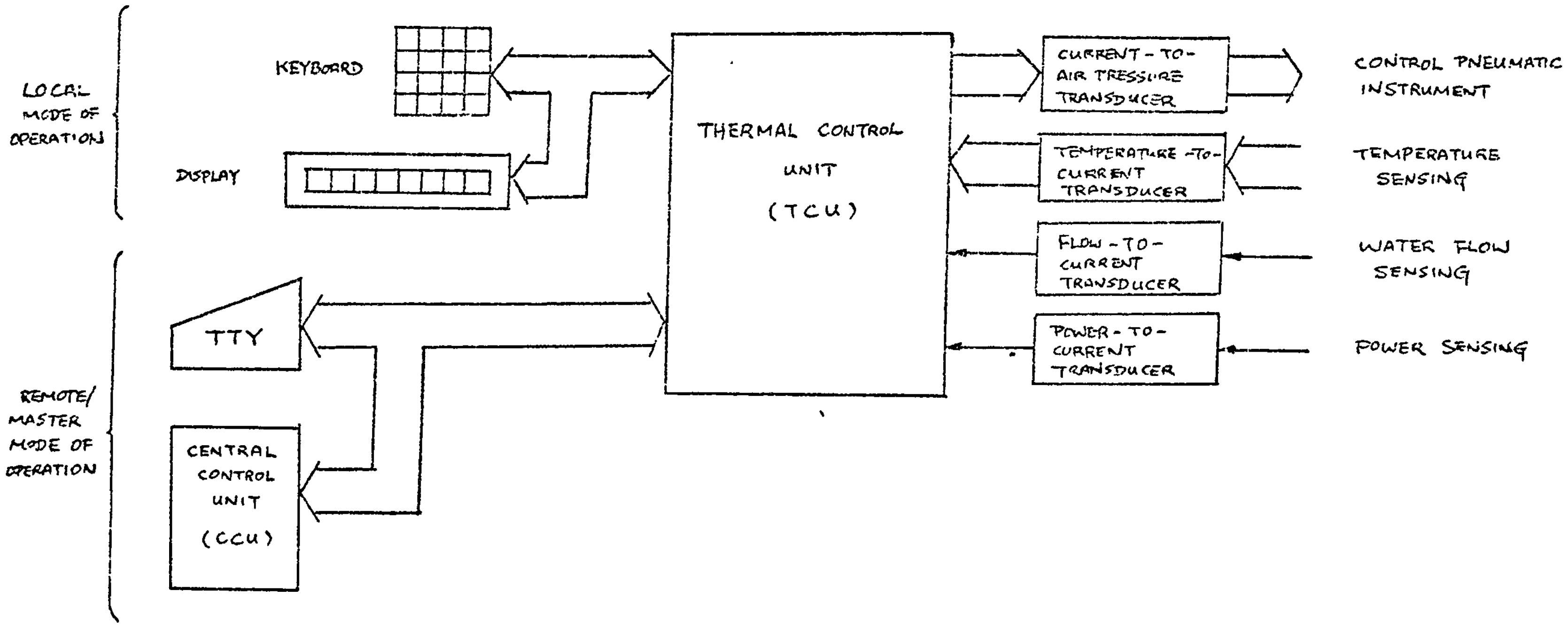
The system should be able to acquire the following variables: (1) temperature of air, (2) temperature of water, (3) flow rate of water, and (4) electrical power consumption. The temperature of air and the temperature of water should be acquired once every five seconds with an accuracy of $\pm 1^{\circ}F$. The temperatures of supply and return chilled water which are used for calculation of cooling energy consumption should be acquired every second with an accuracy of $\pm 0.5^{\circ}F$. The flow rate of water and the electrical power consumption should be acquired once every five seconds with an accuracy of ±1%.

2.2.2 Control Capability Requirements

The operator should have access to the control of all set point temperatures of the air handling unit decks and the hot water converters with an accuracy of ±1%.

2.2.3 System Capability Requirements

The TCU should be designed to be operated by non-technical personnel. It should have a simple and easy to use set of commands which may be entered through a terminal such as teletype or a front panel keyboard. The acquired data and internally updated variables should be presented to the operator in an easy to understand format. The system should provide reports presenting system variables through a printer



LOCAL MODE OF OPERATION
KEYBOARD
DISPLAY
REMOTE/ MASTER MODE OF OPERATION
TTY
CENTRAL CONTROL UNIT (CCU)
THERMAL CONTROL UNIT (TCU)
CURRENT-TO-AIR PRESSURE TRANSDUCER
TEMPERATURE-TO-CURRENT TRANSDUCER
FLOW-TO-CURRENT TRANSDUCER
POWER-TO-CURRENT TRANSDUCER
CONTROL PNEUMATIC INSTRUMENT
TEMPERATURE SENSING
WATER FLOW SENSING
POWER SENSING

Figure 2-1 System interfaces block diagram

or on an LED display. The TCU should self-initialize all programs when power is turned on. It should report any system malfunction in the form of alarm notification to the operator.

CHAPTER 3

OPERATION

This chapter describes the functional design of the system, i.e., the translation of the operational requirements into well defined functions that can be implemented in terms of hardware and software modules. Also described in this chapter is the interaction between functions, and the syntax of the commands used for man/machine interface. An overview of the system operation is also presented.

3.1 Functional Description

A characterization of the Thermal Control Unit (TCU) is best accomplished in terms of a description of the functions that it performs as an integral part of the heating, ventilating and air conditioning system. In order to satisfy all the operational requirements, the TCU should be able to perform nine distinct functions. These functions are listed below and described in detail in the following sections.

- Data Acquisition Function
- Limit Check Function
- Set Point Control Function
- Time Update Function
- Periodic Log Function
- System Alarm Function
- Master Processor Communication Function

- Operator Interaction Function
- Real-time Operation and Initialization Function

3.1.1 Data Acquisition Function

The data acquisition function includes the measurement of temperatures, flow of water, electric power, and presentation of these values in proper engineering units. This function should be performed periodically every five seconds. The energy consumption for cooling a building should be calculated from the value of chilled water flow and the difference in temperatures between the supply and return water. The consumption of thermal energy for heating a building should be established from the flow of steam condensate.

3.1.2 Limit Check Function

The limit check function allows the identification of abnormal system operation by monitoring the value of each system variable and checking if it is within an allowable range. The boundaries of the allowable range are identified by a high limit and a low limit. Occurrence of an out of range condition creates a data alarm condition that should be reported by the alarm function. The operator should have control of whether or not a variable is limit checked. In addition, the limit check function allows the operator to modify the limit of any variable.

3.1.3 Set Point Control Function

The purpose of the set point control function is to provide remote control of the set points of the HVAC system. The set point variables controlled by the system correspond to: (1) hot deck temperatures, (2) cold deck temperatures, (3) temperatures of hot water supply to air handling units, and (4) domestic hot water temperature.

The set point control function allows the set point to be either controlled locally by manual adjustment to the pneumatic controller or to be under an adjustment specified by the system. The set point function provides the operator with a report of the setting of all system set points on request.

The set point control function should include monitoring the controlled variable to detect a possible loss of control.

3.1.4 Time Update Function

The time update function updates the time of the day within the system. This function is required to support other functions which are time dependent (e.g. periodic log function), and also to identify the time when critical system events happen. This function is active at all times.

3.1.5 Periodic Log Function

The purpose of the periodic log function is to tabulate all required data at specified time intervals. This function allows the operator to select specific channels to be tabulated. It is active only under operator demand.

3.1.6 System Alarm Function

The system alarm function warns the system operator of the existence of an abnormal condition in system operation. There are two types of alarm condition: system alarm and data alarm. The first type of alarm is associated with the failure of a system component. A data alarm is associated with a system variable being outside the allowable range specified by its high and low limits. Annunciation of an alarm is triggered by the occurrence of the event. A code number is assigned to each type of alarm to simplify the identification of the cause. A report is provided upon demand by the alarm function to summarize all existing alarms at any given time.

3.1.7 Master Processor Communication Function

The master processor communication function allows the TCU to be controlled by a master processor. This function provides an efficient means for transferring commands to the TCU and receiving data from it. A well defined protocol for communication between the TCU and a master processor is part of this function.

3.1.8 Operator Interaction Function

This function enables the operator to interact with the TCU functions through commands and responses. It accepts commands, analyzes the command syntax, protects the system from operator errors, informs the operator when a wrong entry is made, and presents information in an easily understandable format. Command mnemonics are used to simplify the system use by non-technical personnel.

3.1.9 Real-time Operation and Initialization Function

Insofar as the operator's perception, all functions are performed simultaneously. The system is controlled by either the operator or the master processor. The front panel keyboard and display allows the monitoring of system variables while the system is under control of a remote operator or master processor. Since the data acquisition function generates the data evaluated by the limit check function, the data acquisition function should be performed prior to the limit check function.

When the system is initially powered, default options are specified for each function. These options are as follows: (1) time is initialized to zero, (2) power up alarms and system alarms are reported, and the operator interaction function becomes ready to accept command, (3) data acquisition is performed periodically, (4) all set points are set to manual mode, and (5) logging function is not performed.

3.2 User Language and Responses

In order for the operator to communicate with the system, a simple language was developed that allows the operator to enter system commands which are simple and easy to remember. Through the use of these commands, the user can start, modify, and stop system functions. Appropriate responses presented in an easy to understand format are given to the operator. The system is designed to accept commands from three distinct types of devices; a keyboard on the TCU front panel, a terminal such as a teletype, or a supervisory master processor. The command entry and responses are designed to take maximum advantages of the features provided by each input/output device.

3.2.1 Command Syntax

On the TCU front panel keyboard, special keys are used to enter each permissible command. The numerical keys are used to enter data associated with each command. Figure 3-1 shows the data which must be entered following a command key stroke.

The command syntax for the terminal and the master processor consists of three elements: instruction, data set, and delimiters. An instruction is a valid combination of two of the following alphabetic characters:

A,B,C,E,H,L,M,N,P,R,S,T,V

Data consists of a combination of the digits 0 through 9; or just "A" by itself, which represents all channels. The length of data varies from one to eight digits, and the number of data for a data set varies from zero to six. There are two delimiters: one is used to separate data from commands or other data, the other identifies the end of a command. A delete character is provided to allow an incorrectly entered command to be cancelled. To simplify the description of commands, the data separation delimiter will be denoted by "-", the end of command delimiter by "@", and the delete character by "$". The keys corresponding to each delimiter and the delete character are presented in Table 3-1.

A repetoire of sixteen commands was developed to control the system functions. A complete list of the TCU commands is presented in Table 3-2. A command starts with an instruction which may be followed by a data set as shown in Figure 3-2.

As an example of a command, consider setting the system time to 12 hours, 15 minutes, and 30 seconds. The following key strokes should be entered on a terminal to perform this action:

```
TS-12-15-30@
```

If an error is made while entering the instruction or data, the partially entered command may be deleted by using the delete key. For example, if an error is made while keying-in the minutes data, the whole command may be deleted as follows:

```
TS-12-16$
```

Upon recognition of the delete key depression, the system will be ready to accept a new command.

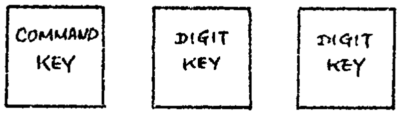
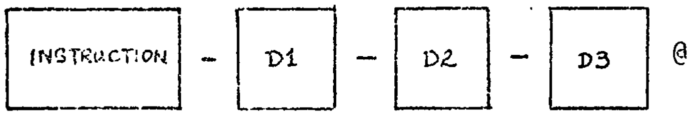


Figure 3-1 Command syntax for front panel keyboard entries

Table 3-1 Delimiter/Delete character description

DELIMITER/DELETE KEY	KEY	FUNCTION
-	Space	Separation
@	Carriage Return	End of Command
$	Rubout	Delete



where D1 is data 1

D2 is data 2

D3 is data 3

D1, D2, and D3 constitutes a data set

Figure 3-2 Command syntax for master processor or terminal entries

Table 3-2 TCU user commands and description

FUNCTION	COMMAND	DESCRIPTION
Data Acquisition	CR	Report Cooling Energy Usage
	CS	Set Cooling Ton-hours
	HR	Report Heating Energy Usage
	HS	Set Pound of Condensate
	VR	Variable (Channel) Report
Limit Check	LL	Limit Load
	LC	Clear Limit Check
	LE	Limit Edit
Set Point Control	SM	Set to Manual Mode
	SA	Set Point Adjust
	SR	Set Point Report
Time Update	TS	Time Set
	TM	Time Report
Periodic Log	PB	Begin Periodic Report
	PN	End Periodic Report
System Alarm	AR	Alarm Report

3.2.2 Responses

The format of the system response has been designed to maximize the utilization of the available printing or display capabilities.

The TCU front panel has eight seven-segment LED elements which are used to display the responses to command entered through the front panel keyboard.

Commands entered through a terminal keyboard will cause the system to provide a complete alphanumeric response which is more detailed than that of the front panel display. Prior to the entry of a command, the system will print a character ">" on the first column which is used to indicate that the system is ready to accept a command. If there is an error in the command entered, the system will respond with "WHAT?". If the number of character entered is greater than twenty-two, or if a delete character is entered, the system will respond with "CANCELLED".

CHAPTER 4

SOFTWARE ORGANIZATION

In this chapter, the conceptual functions of the system described in chapter 3 are described as system tasks. The commands and responses required for operator interaction with each function are also implemented.

There are a total of nine tasks as listed below:

- Interpreter Task
- Data Acquisition Task
- Temperature Averaging Task
- Set Point Task
- Keyboard Task
- Display Task
- Debug Task
- Paper Logging Task
- Protocol Task

In order to maintain a real-time system performance, each task is given control of the CPU for short period of time. A real-time executive is used to support the control of task execution. The executive program works in conjunction with the update task to maintain task status and system time. A service interrupt subroutine is used to accumulate pulses from a hardware clock to provide the system time. The TTY input task and the TTY output task provide the input and output services for the executive. Figure 4-1 illustrates the software organization.

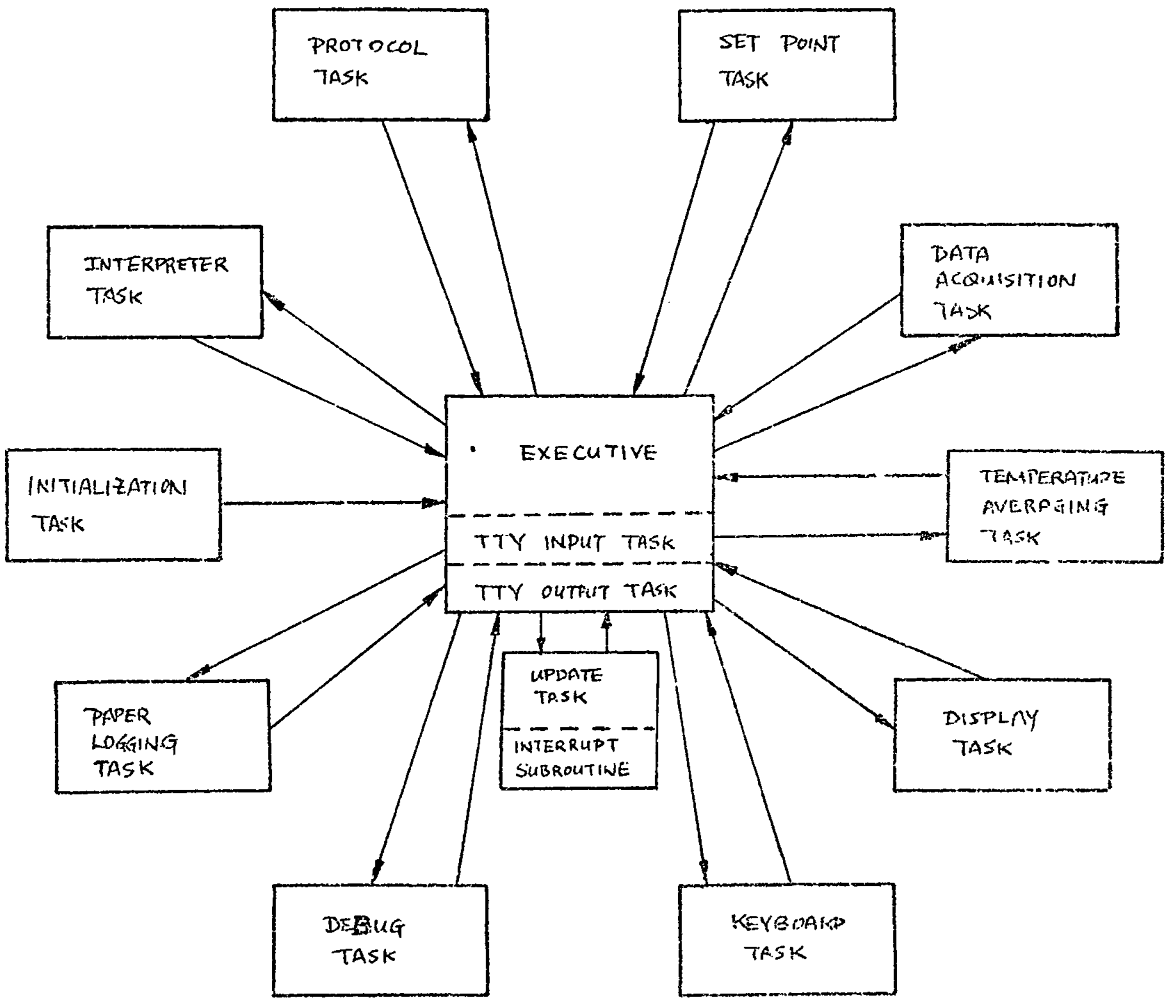


Figure 4-1 Software organization

4.1 User Task Description

In this section, user tasks are described in terms of the system functions they implement. A detailed description of each task's conceptual algorithm is presented. The interaction between tasks and between a task and its related peripheral hardware is also described.

4.1.1 Interpreter Task

The interpreter task contributes to the implementation of the following conceptual functions: limit check function, set point control function, time update function, periodic log function, system alarm function, master processor communication function, and operator communication function. A block diagram showing the interfaces of the interpreter task is presented in Figure 4-2.

The interpreter task is designed to have the following capabilities:

- Self-initializes when power is turned on.
- Announcement of power up alarm.
- Checks the mode of system operation periodically.
- Notifies the operator with a prompt sign when it is ready to accept a new command.
- Checks whether the input command is ready for interpreting.
- Checks whether the rubout bit is set, and outputs a "CANCELLED" message.

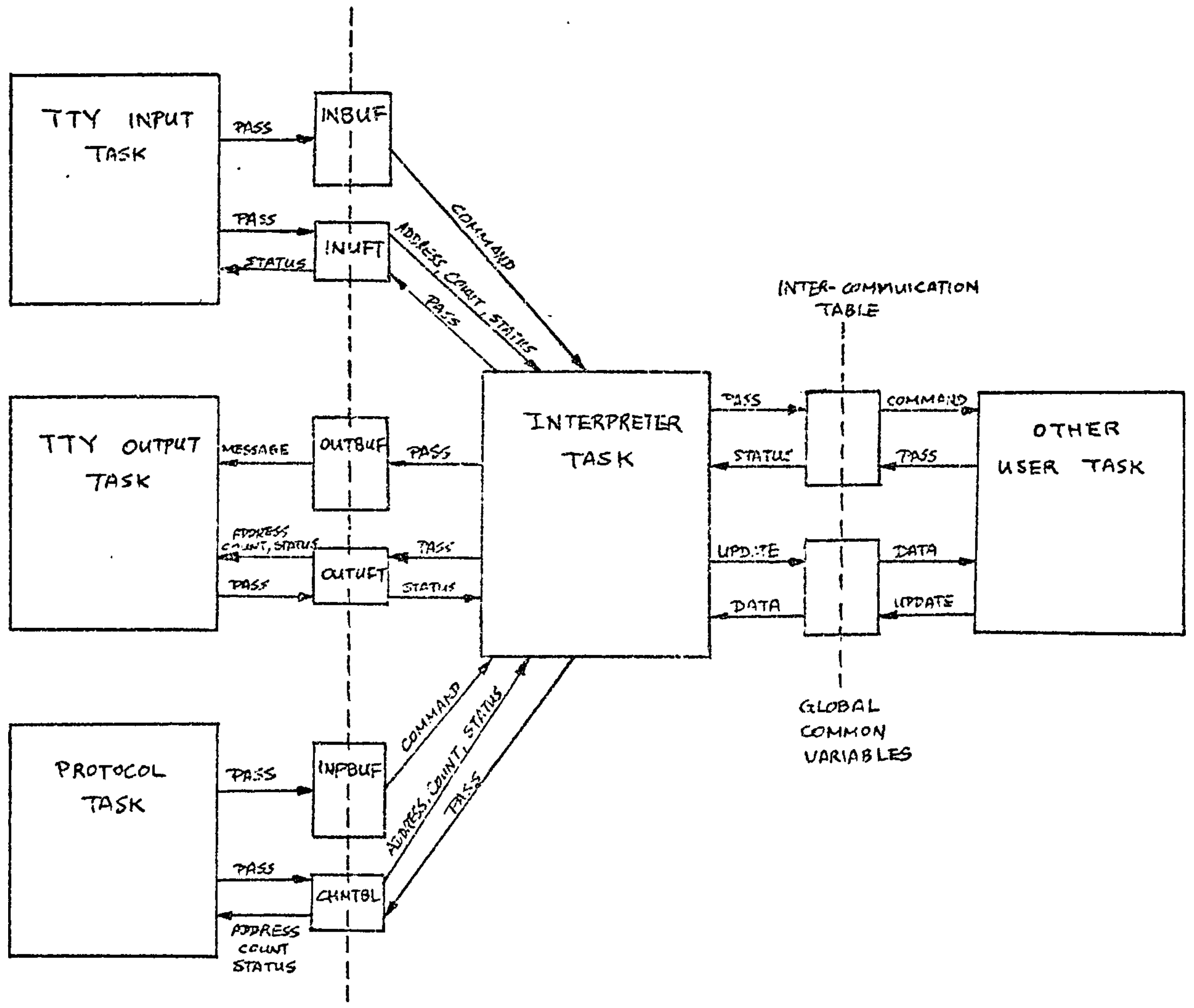


Figure 4-2 Interface block diagram of interpreter task

- Interprets the command and checks for command syntax errors, outputs "WHAT?" when an error is detected.
- Checks the data for an out of range error and outputs "WHAT?" when an error is detected.
- Set time and clears power up alarm.
- Presets the heating and cooling energy totalizers.
- Prints the cooling energy usage, the heating energy usage, channel values, limit values, data alarm conditions, system alarm messages, set point values, mode of operation, and time of the day.
- Modifies the high and the low alarm limits of system variables.
- Loads limits into appropriate memory locations and initiates limit check.
- Terminates limit check of specified channel.
- Commands the set point task to adjust a set point.
- Converts the value of a set point from the range of -50 to 50 to a value between 0 and 100 for use by the set point task.
- Activates the paper logging task through executive service request, and communicates to the task the necessary information through

a block of memory locations.

- Deactives the paper logging task through executive service request.

The interpreter task implements the user language and responses described in chapter 2. It acts as an interface between the user entered command and other user tasks. It directs the activities of the set point task through the communication table SPINCM, the paper logging task through CHNUM, the protocol task through CHNTBL, and the debug task. It retrieves data from the global common locations into which other tasks write. It communicates with the operator through the TTY input and the TTY output tasks. The memory blocks used for this purpose are INBUF, INUFT, OUTBUF, and OUTUFT. It does not directly interact with any hardware except storage locations.

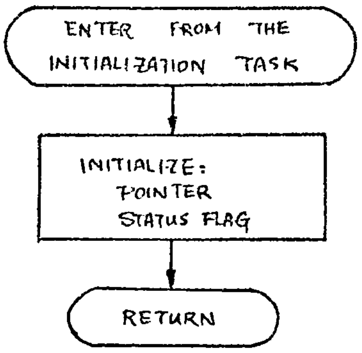
The flow chart of the interpreter task is shown in Figure 4-3.

4.1.2 Data Acquisition Task

Since the data acquisition function and the limit check function have to be executed consecutively, they are implemented together in the data acquisition task.

The data acquisition task is designed to have the following capabilities:

- Reads periodically the channel values in counts from the analog-to-digital card and converts them into engineering units which are stored in


ENTER FROM THE INITIALIZATION TASK
INITIALIZE: POINTER STATUS FLAG
RETURN

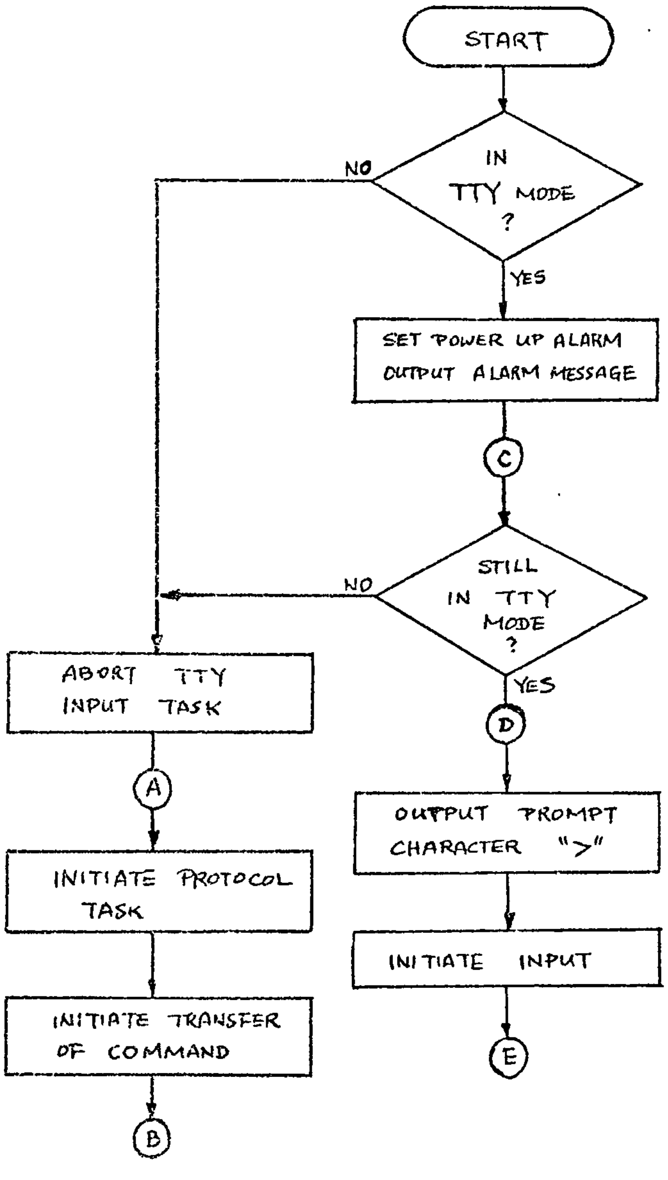


Figure 4-3 Flow chart of interpreter task

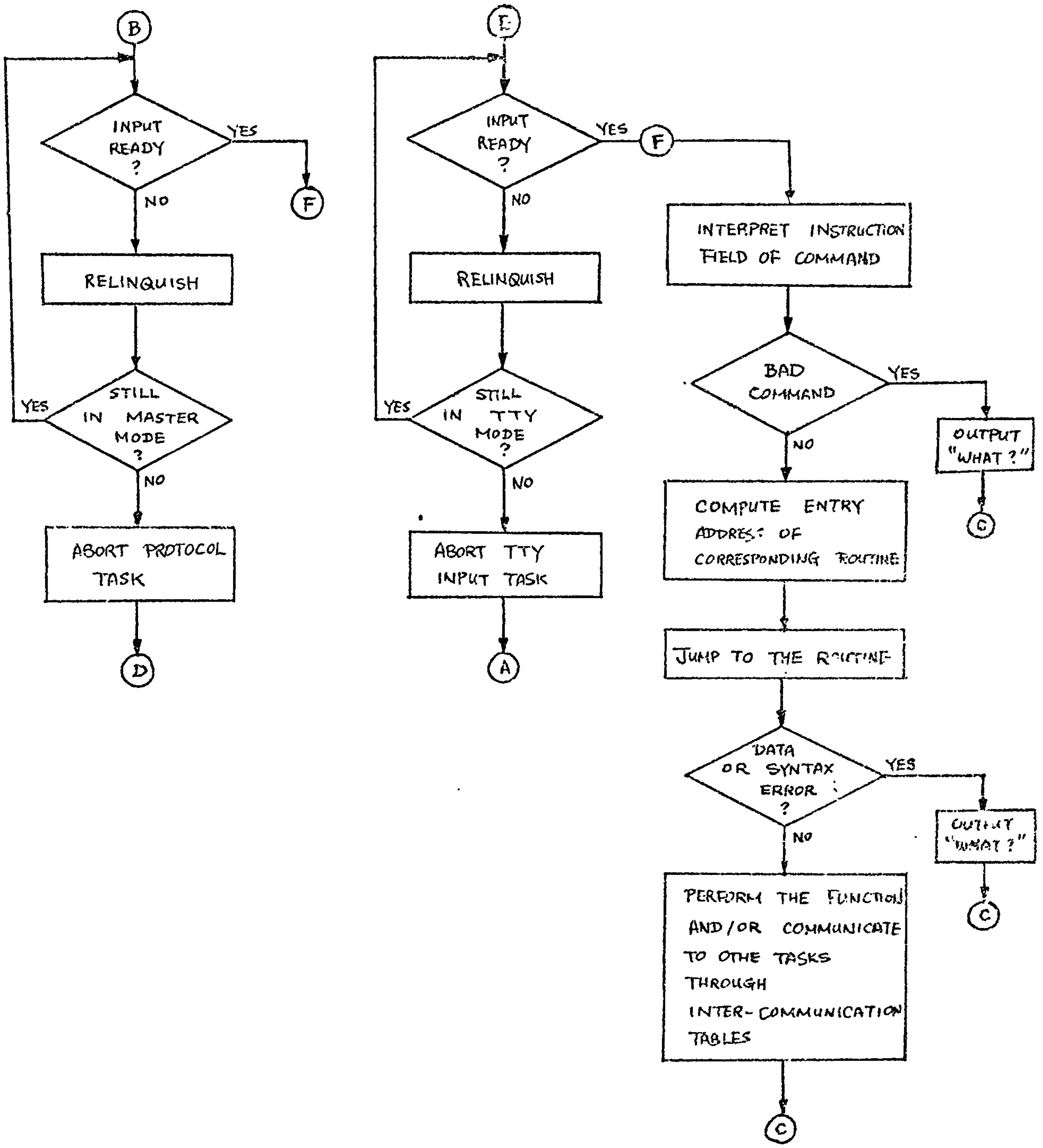


Figure 4-3(cont.) Flow chart of interpreter task the CHNVAL global common block.

- Calculates and accumulates the heating energy in units of steam condensate in the CNDSAT global common block.
- Updates high and low channel alarm flags and limit check flag in the CHNALM global common block.
- Updates the alarm status in the DAINTC global common block.

The flow chart of the data acquisition task is shown in Figure 4-4, and the block diagram showing the task interfaces is presented in Figure 4-5.

4.1.3 Temperature Averaging Task

This task also contributes to the implementation of the data acquisition function. It reads and averages the temperatures of the chilled water supply and return to the building. The averaged temperatures are used to compute and accumulate the cooling energy in units of ton-hour every five seconds. It gives accuracy to the accumulated cooling energy consumption.

The flow chart of the temperature averaging task is shown in Figure 4-6, and the block diagram showing the task interfaces is presented in Figure 4-7.

4.1.4 Set Point Task

The set point task contributes to the implementation

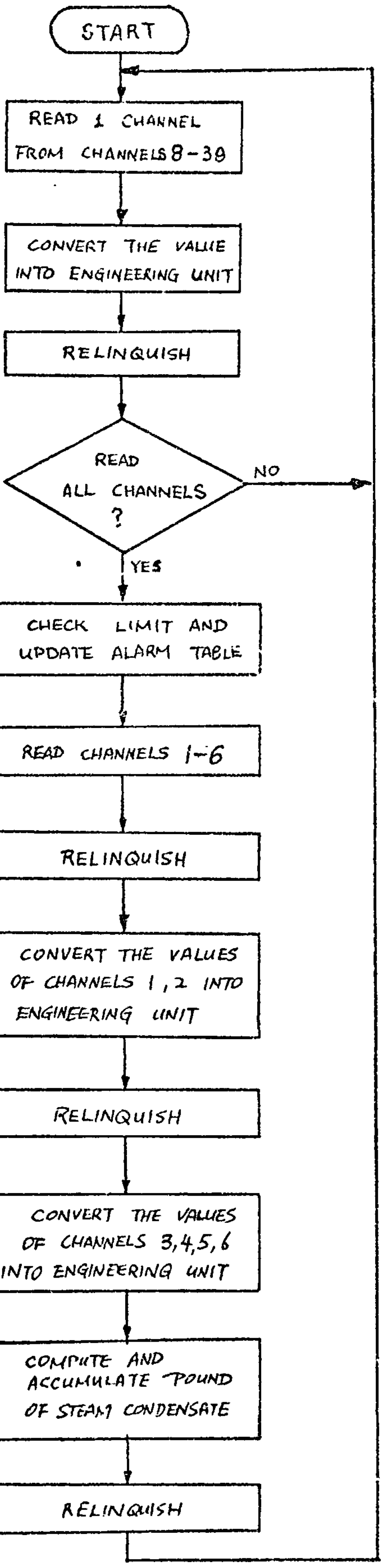

START
READ 1 CHANNEL FROM CHANNELS 8-30
CONVERT THE VALUE INTO ENGINEERING UNIT
RELINQUISH
READ ALL CHANNELS ?
NO
YES
CHECK LIMIT AND UPDATE ALARM TABLE
READ CHANNELS 1-6
RELINQUISH
CONVERT THE VALUES OF CHANNELS 1, 2 INTO ENGINEERING UNIT
RELINQUISH
CONVERT THE VALUES OF CHANNELS 3,4,5,6 INTO ENGINEERING UNIT
COMPUTE AND ACCUMULATE POUND OF STEAM CONDENSATE
RELINQUISH

Figure 4-4 Flow chart of data acquisition task

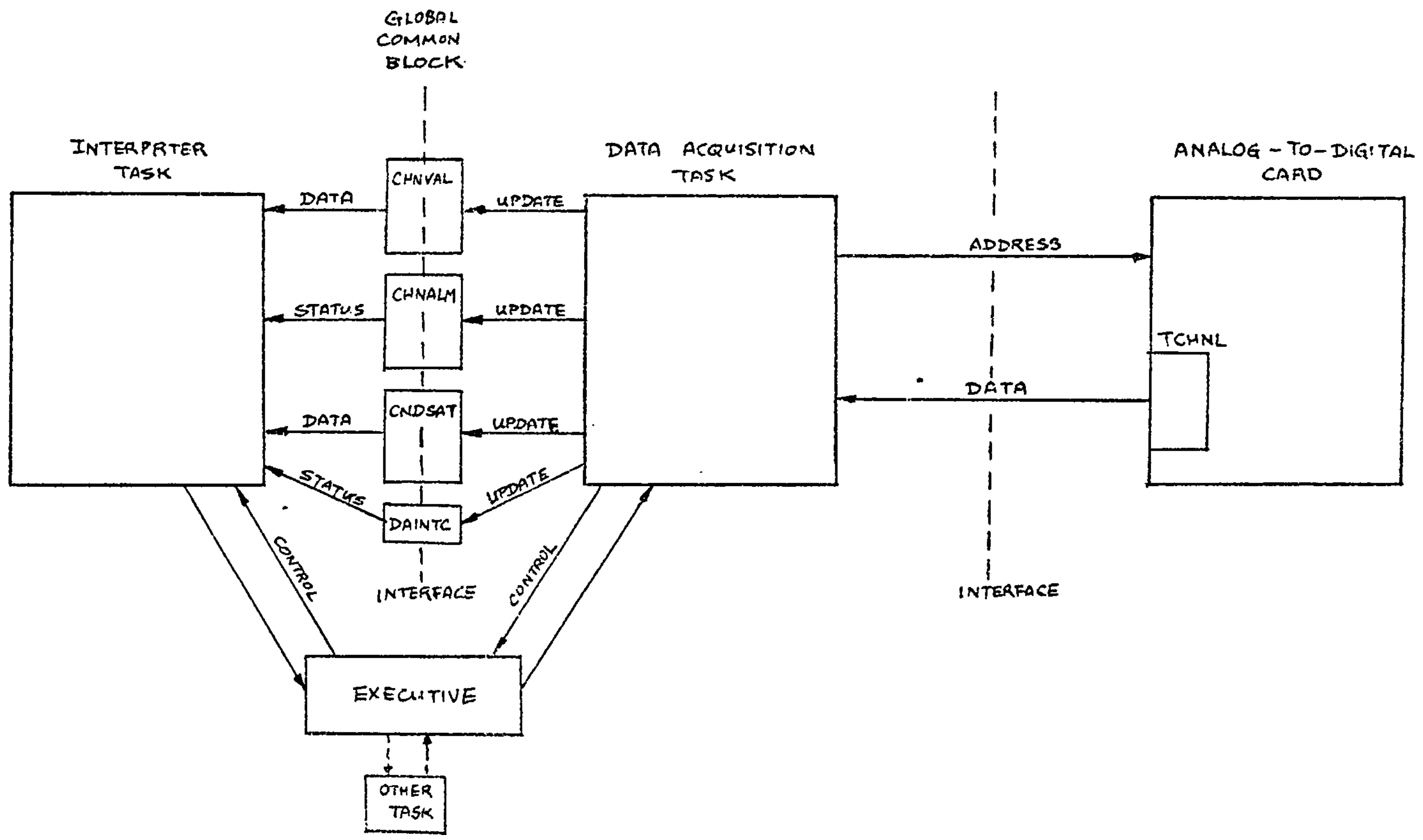


Figure 4-5 Interface block diagram of data acquisition task

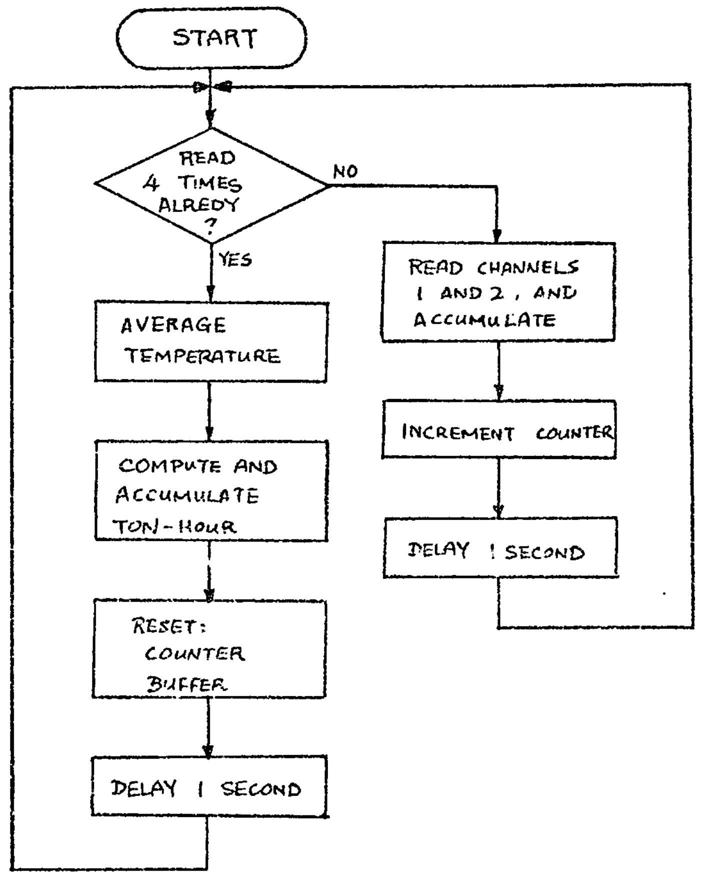


Figure 4-6 Flow chart of temperature averaging task

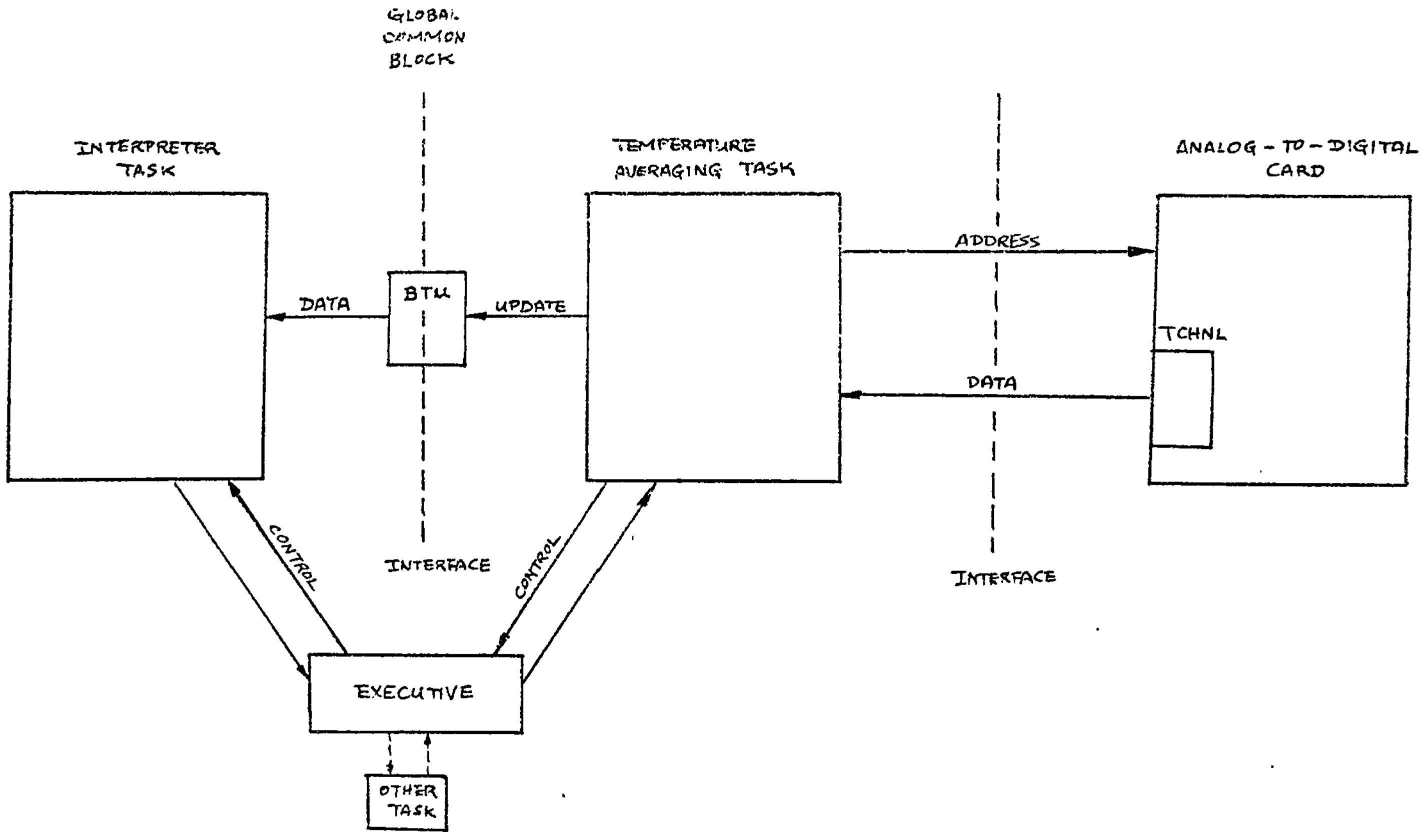


Figure 4-7 Interface block diagram of temperature averaging task

of the set point control function.

It is designed to have the following capabilities:

- Every control line is set to manual mode when the power is turned on.
- Resets a control line to manual control by outputing zero to the corresponding DAC location when requested by the interpreter task. The mode table and the set point value table are then updated.
- Adjusts the set point of a specific control line if requested by the interpreter task. It accepts value from 0 to 100 and outputs 0 to 200 to the corresponding DAC location for normal control (hot deck), and 200 to 0 for reverse control (cold deck). The mode table and the set point value table are then updated.
- Checks each control line's integrity and updates the set point alarm table periodically. When an alarm condition arises, the alarm is passed to the interpreter task through the SPINCM inter-communication table.
- Checks set point control current loop power supplies periodically. In case of a power failure, the alarm condition is also passed to the interpreter task through the SPINCM inter-communication table.

- The task runs every five seconds.

The set point task interfaces with the hardware set point module by writing into or reading from local blocks on the set point module. It adjusts the value of a set point by loading one of the 32 consecutive DAC locations with values between 0 and 200. It establishes the integrity status of the control lines in the form of status bit map from the 4 LOOP locations which are eight bytes apart from each other. It also reads the current loop power supplies status in form of status bit from the 4 CARD locations which are also eight bytes apart from each other.

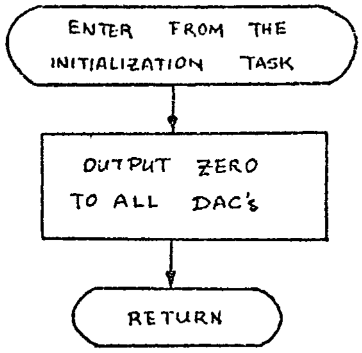
The flow chart of the set point task is shown in Figure 4-8, and the block diagram showing the task interfaces is presented in Figure 4-9.

4.1.5 Keyboard Task

The keyboard task contributes to the implementation of the operator communication function.

The task is designed with the following capabilities:

- Polls the MSB of the PIAA+1 periodically for command entry detection.
- Reads the input command from the 4 MSB of PIAA and sends the command code to the display task through REQJOB.
- Reads the channel number from the 4 MSB of PIAA and sends it to the display task in binary format through CHNNO and in BCD format through



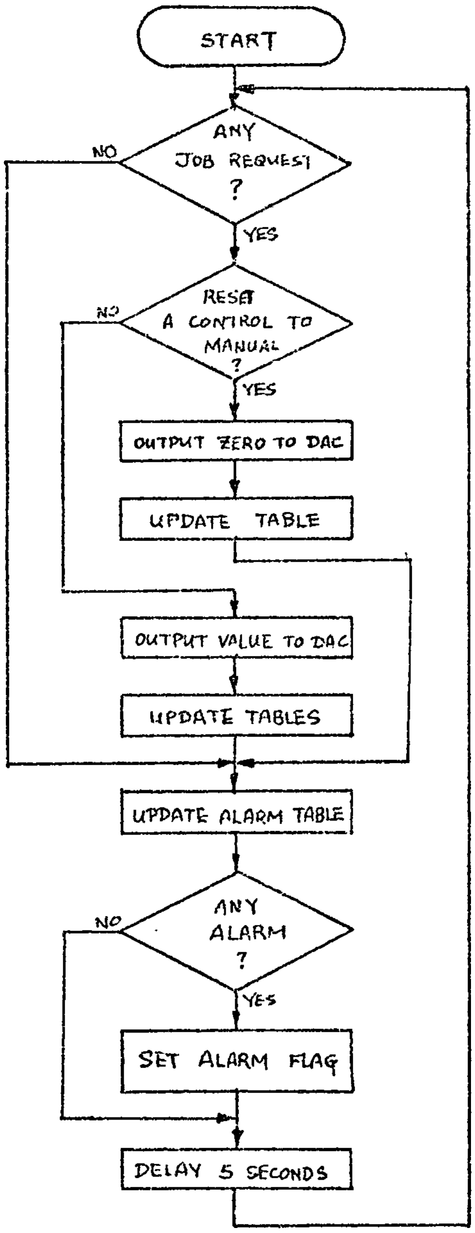


Figure 4-8 Flow chart of set point task

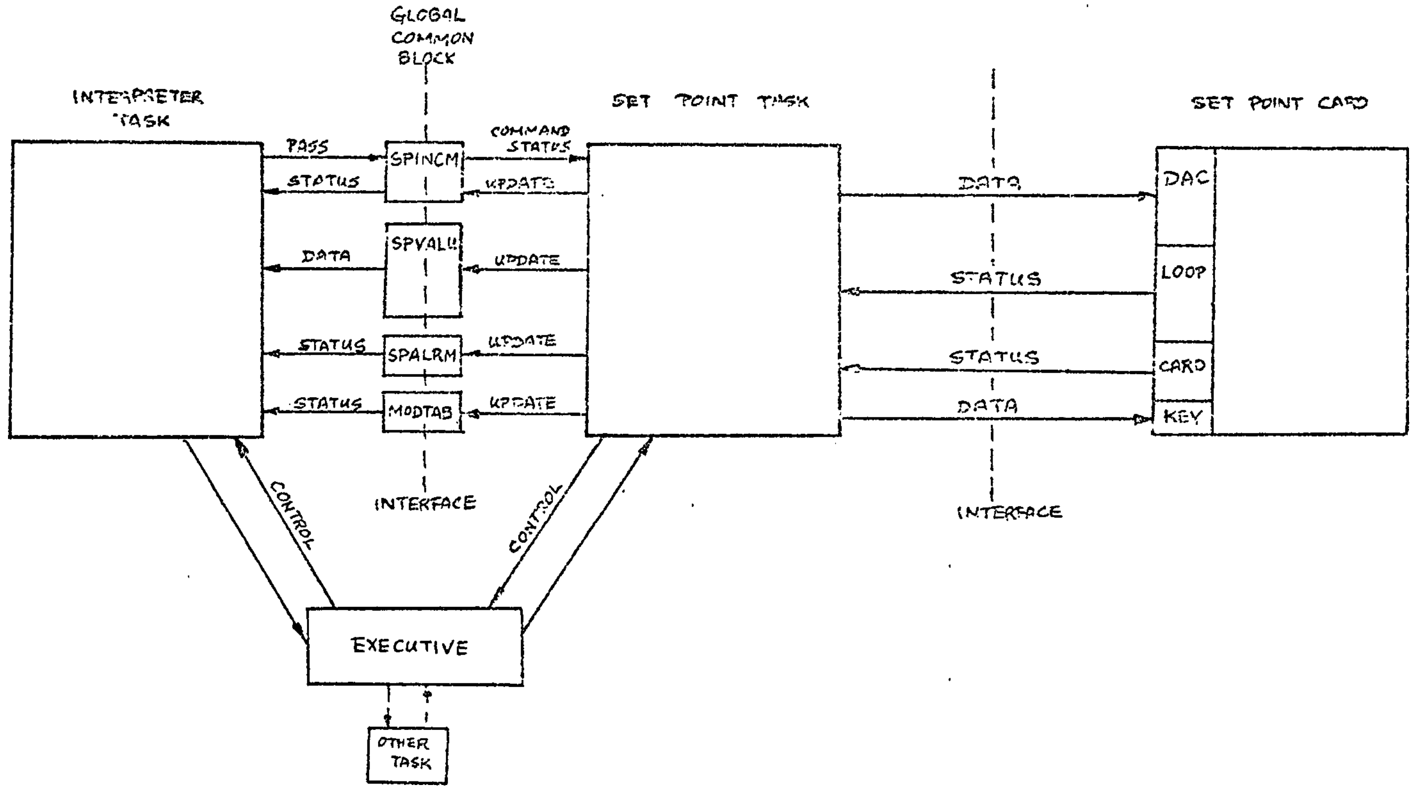


Figure 4-9 Interface block diagram of set point task

CHNNO1.

The keyboard task only interacts with the display task. It is always ready to accept a command entry.

The flow chart of the keyboard task is shown in Figure 4-10, and the block diagram showing the task interfaces is presented in Figure 4-11.

4.1.6 Display Task

This task also contributes to the implementation of the operator communication function.

It is designed to have the following capabilities:

- Self-initializes, presets, and displays the system time.
- Outputs channel reports, cooling energy report, heating energy report, and time of the day according to the code passed through REQJOB by the keyboard task.
- Receives channel number, channel value, time of the day, ton-hours of chilled water, pounds of steam condensate through CHNNO, CHNVAL, TIME, BTU, and CNDSAT respectively and outputs to the 4 LSB of PIAB.

The display task does not interact with any other user task nor the routines of the executive except the keyboard task.

The flow chart of the display task is shown in Figure 4-12, and the block diagram showing the task interfaces is presented in Figure 4-13.

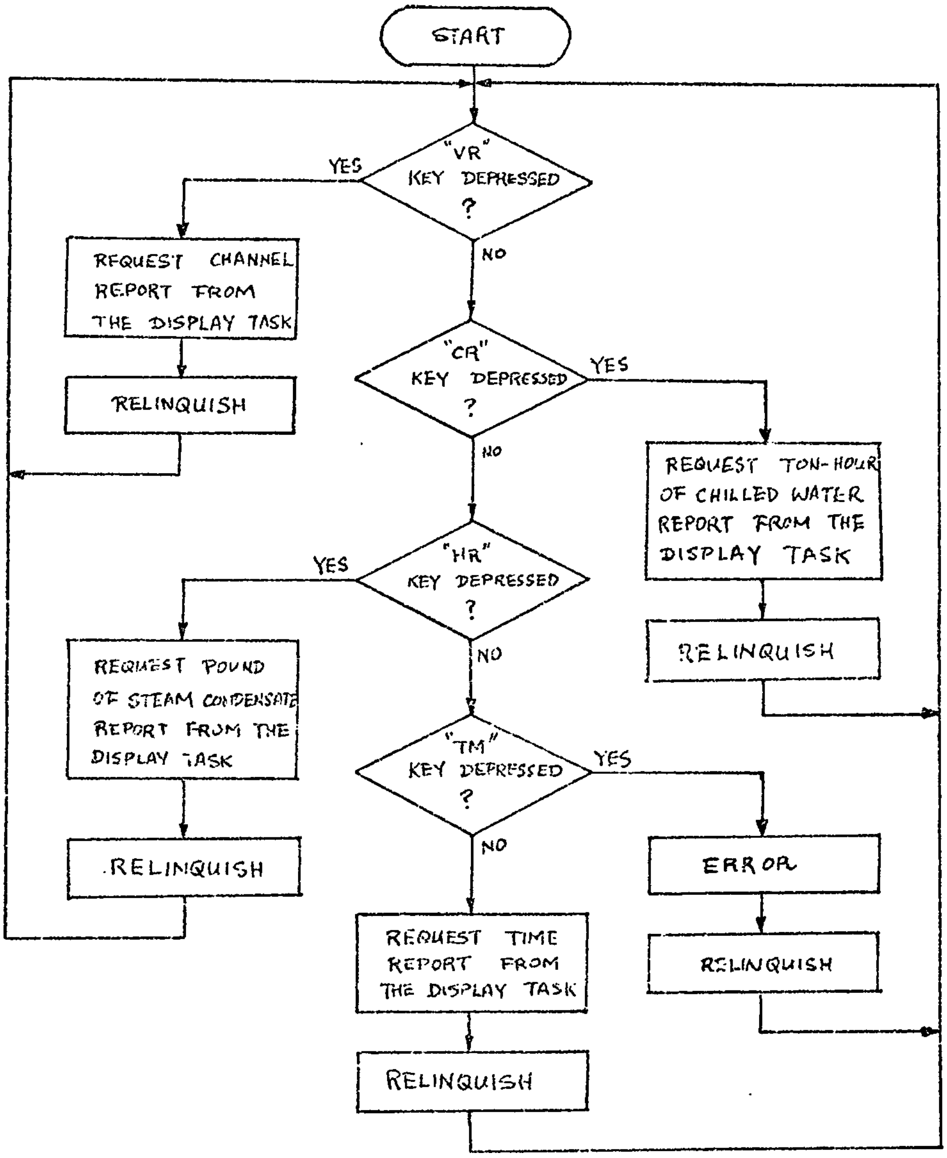

START
"VR" KEY DEPRESSED ?
YES
NO
REQUEST CHANNEL REPORT FROM THE DISPLAY TASK
RELINQUISH
"CR" KEY DEPRESSED ?
YES
NO
REQUEST TON-HOUR OF CHILLED WATER REPORT FROM THE DISPLAY TASK
RELINQUISH
"HR" KEY DEPRESSED ?
YES
NO
REQUEST POUND OF STEAM CONDENSATE REPORT FROM THE DISPLAY TASK
RELINQUISH
"TM" KEY DEPRESSED ?
YES
NO
ERROR
RELINQUISH
REQUEST TIME REPORT FROM THE DISPLAY TASK
RELINQUISH

Figure 4-10 Flow chart of keyboard task

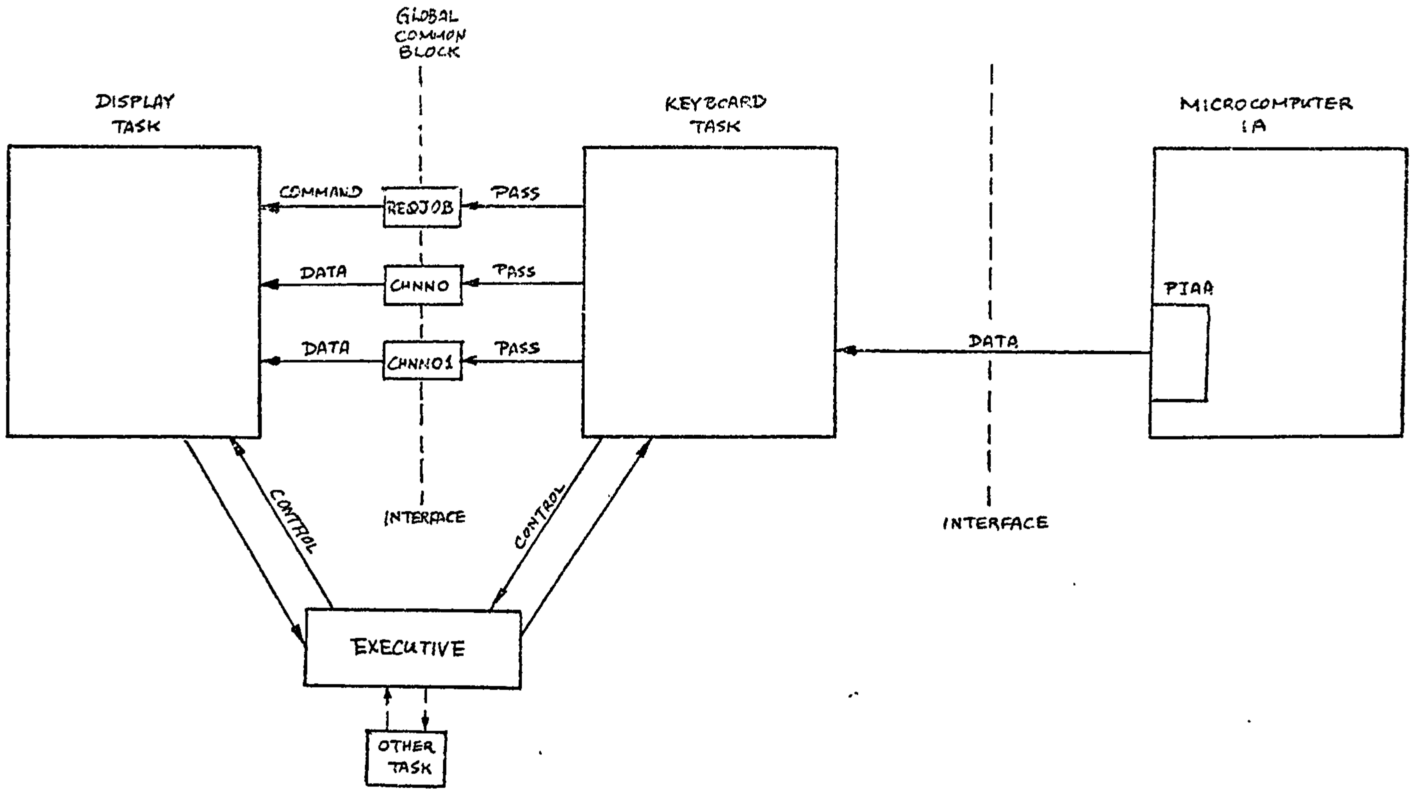
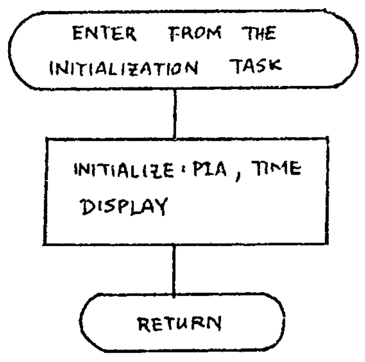


Figure 4-11 Interface block diagram of keyboard task


ENTER FROM THE INITIALIZATION TASK
INITIALIZE : PIA , TIME DISPLAY
RETURN

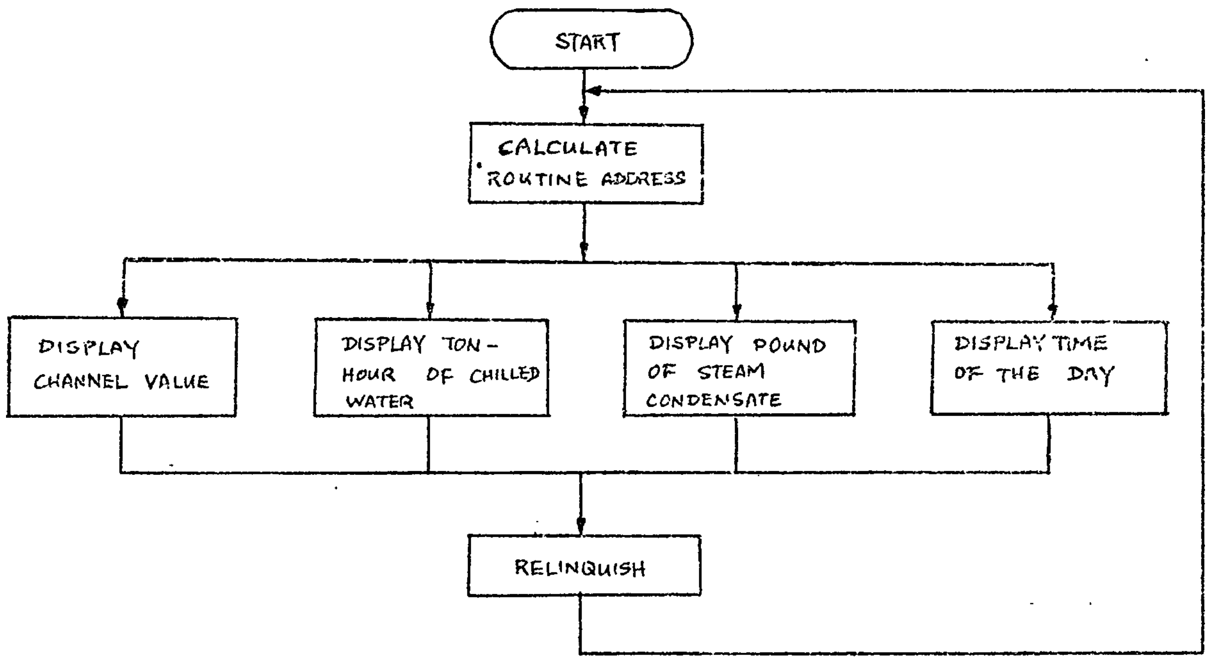

START
CALCULATE ROUTINE ADDRESS
DISPLAY CHANNEL VALUE
DISPLAY TON-HOUR OF CHILLED WATER
DISPLAY POUND OF STEAM CONDENSATE
DISPLAY TIME OF THE DAY
RELINQUISH

Figure 4-12 Flow chart of display task

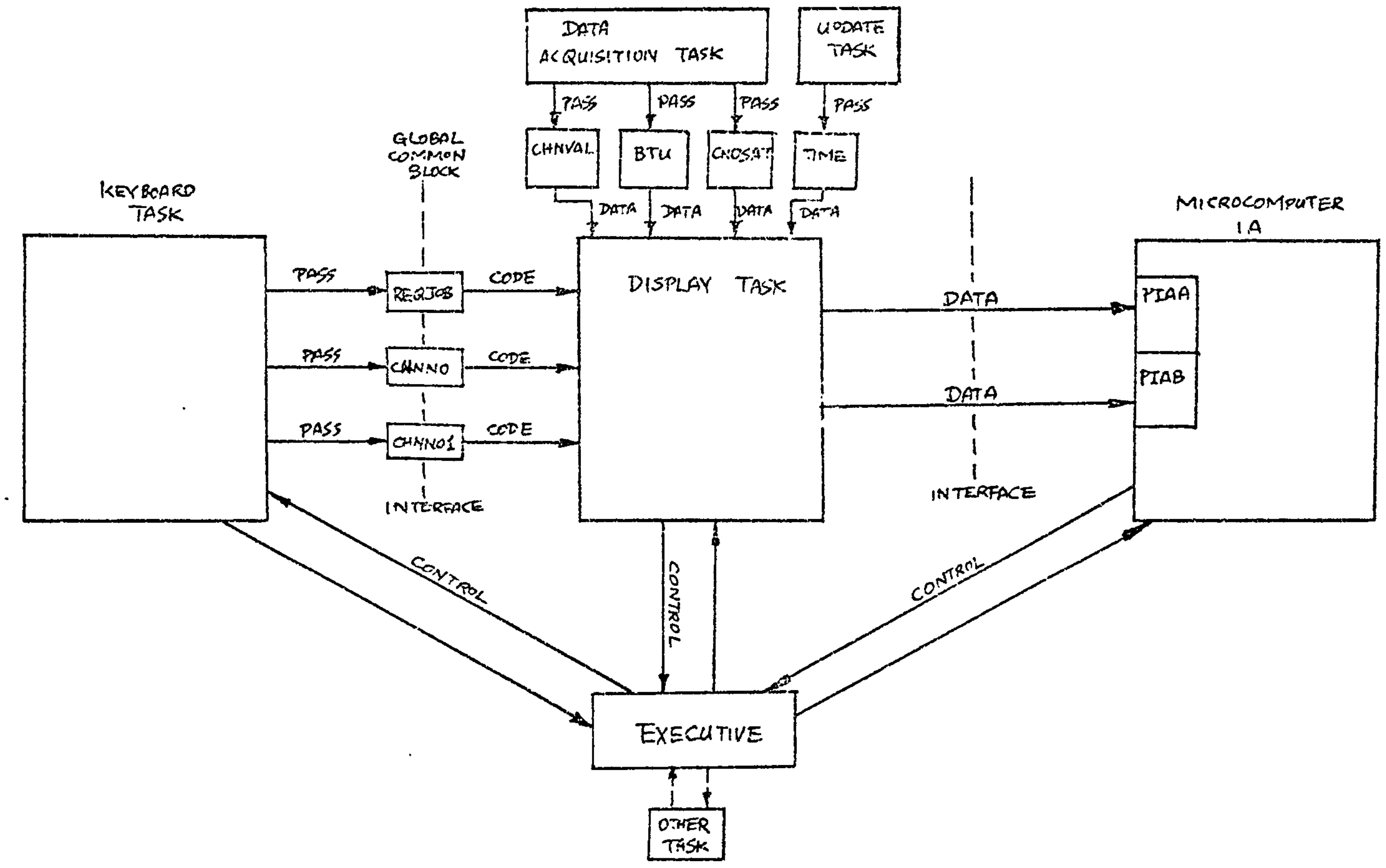


Figure 4-13 Interface block diagram of display task

4.1.7 Debug Task

This task is designed for software debugging and system maintenance. It allows the operator to examine and change the contents of RAM locations, to examine the contents of ROM locations and to interact with peripheral devices interfaced through memory addresses. It does not interact with any other user task.

The flow chart of the debug task is shown in Figure 4-14, and the block diagram showing the task interfaces is presented in Figure 4-15.

4.1.8 Paper Logging Task

The paper logging task contributes to the implementation of the periodic log function.

It is designed with the following capabilities:

- Two logging options are available: one which reports a selected set of channels (maximum of six channels), and the other which reports all configured channels. Both options provide a periodic report which includes time of day, pounds of steam condensate consumption and ton-hours of chilled water consumption.
- The default period for logging is set to 12 minutes, but it can be changed to any integer number of minutes.
- The logging function can be started or stopped at any time.

The paper logging task receives the channel numbers or

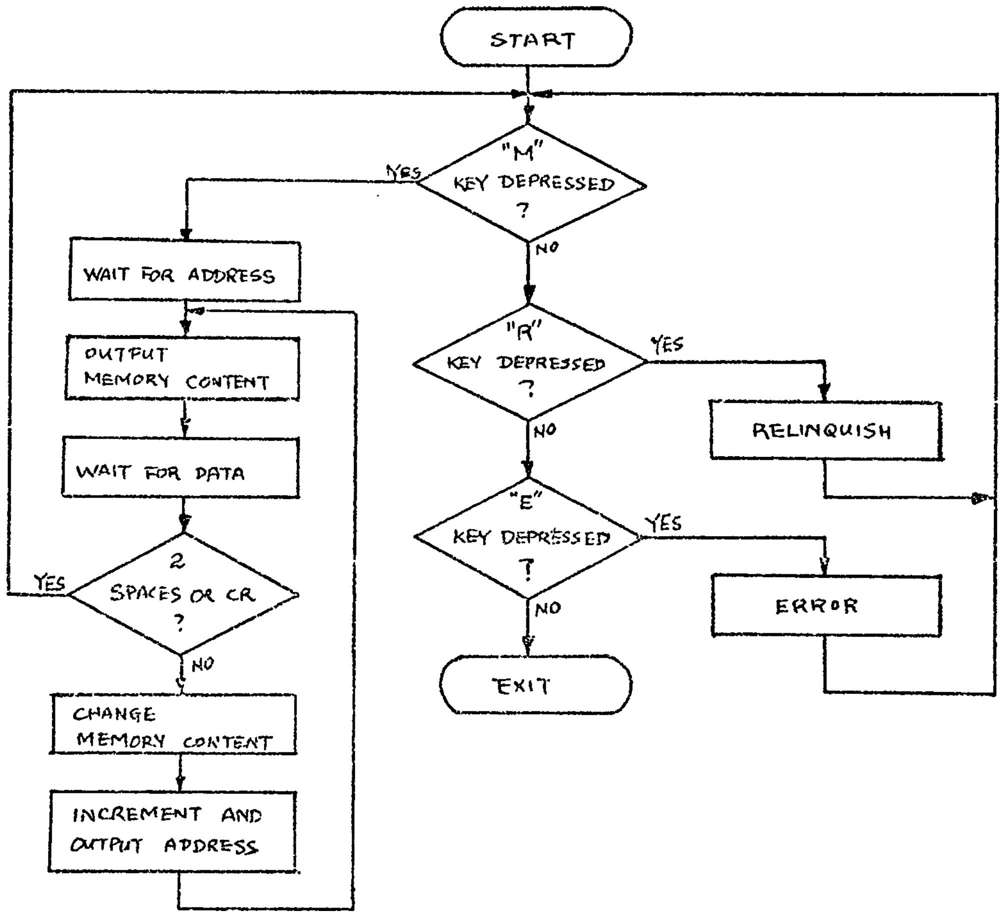


Figure 4-14 Flow chart of debug task

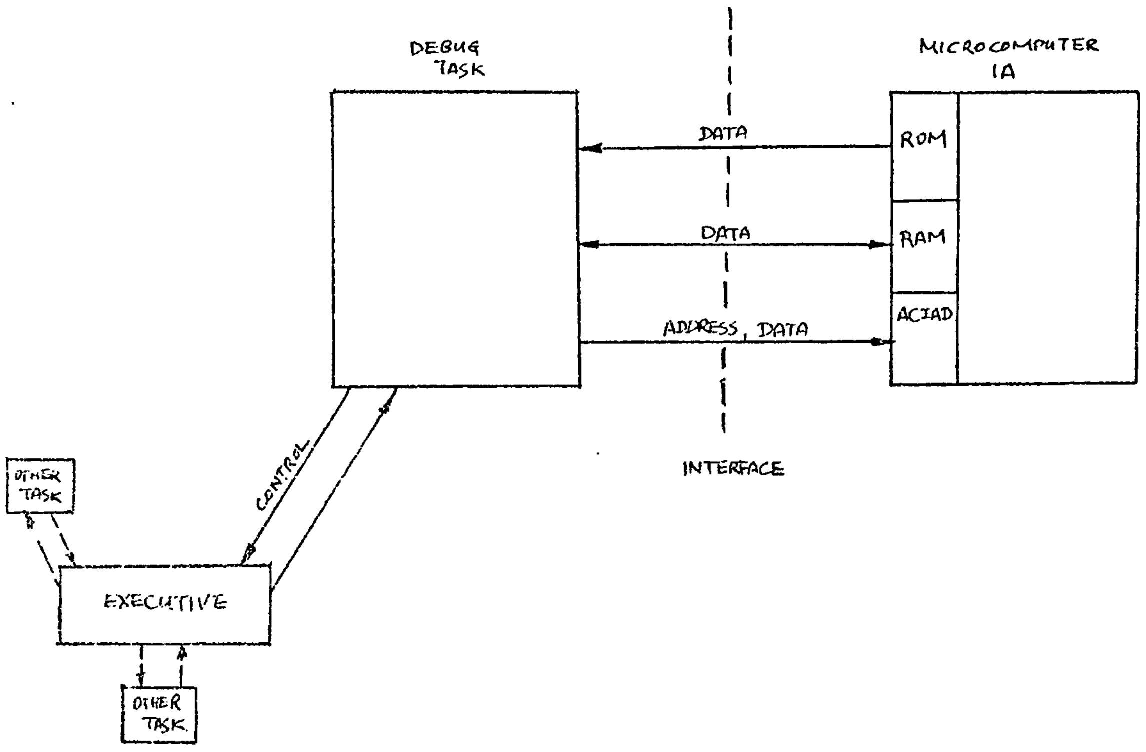


Figure 4-15 Interface block diagram of debug task

the all channel report option flag through the CHNUM common block, the period in seconds or in minutes through the DELAY common byte, as they are passed from the interpreter task. It fetches data from the global common blocks CHNVAL, BTU, CNDSAT, and TIME; and then passes to the TTY output task through the buffer BUFFER and requests output services through UFT. The paper logging task is the only other task that requests TTY output service besides the interpreter task. Notice that the two tasks use separate output buffers and user file tables.

The flow chart of the paper logging task is shown in Figure 4-16, and the block diagram showing the task interfaces is presented in Figure 4-17.

4.1.9 Protocol Task

The protocol task contributes to the implementation of the master processor communication function. It acts as an interface between the Central Control Unit (CCU) protocol task and the TCU interpreter task.

The protocol task is designed to have the following capabilities:

- Receives a block of data with commands coded in ASCII from the master processor through the ACIA on the master/TTY interface.
- Transfers and receives status, address and counts to and from the interpreter task through the inter-communication table CHNTBL.

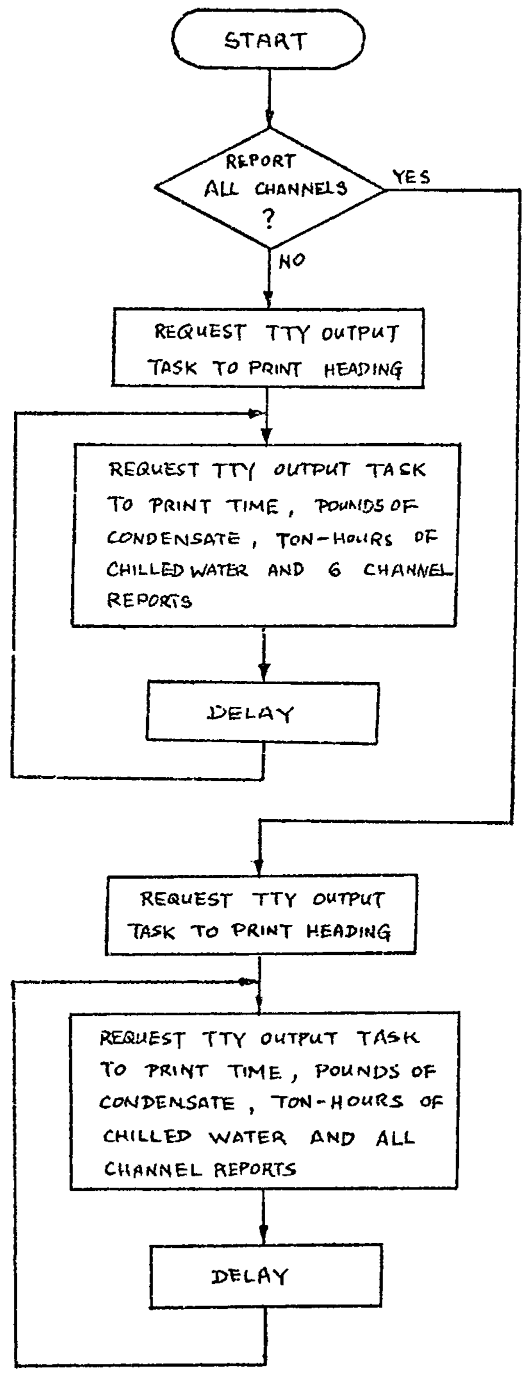

START
REPORT ALL CHANNELS ?
YES
NO
REQUEST TTY OUTPUT TASK TO PRINT HEADING
REQUEST TTY OUTPUT TASK TO PRINT TIME, POUNDS OF CONDENSATE, TON-HOURS OF CHILLED WATER AND 6 CHANNEL REPORTS
DELAY
REQUEST TTY OUTPUT TASK TO PRINT HEADING
REQUEST TTY OUTPUT TASK TO PRINT TIME, POUNDS OF CONDENSATE, TON-HOURS OF CHILLED WATER AND ALL CHANNEL REPORTS
DELAY

Figure 4-16 Flow chart of paper logging task

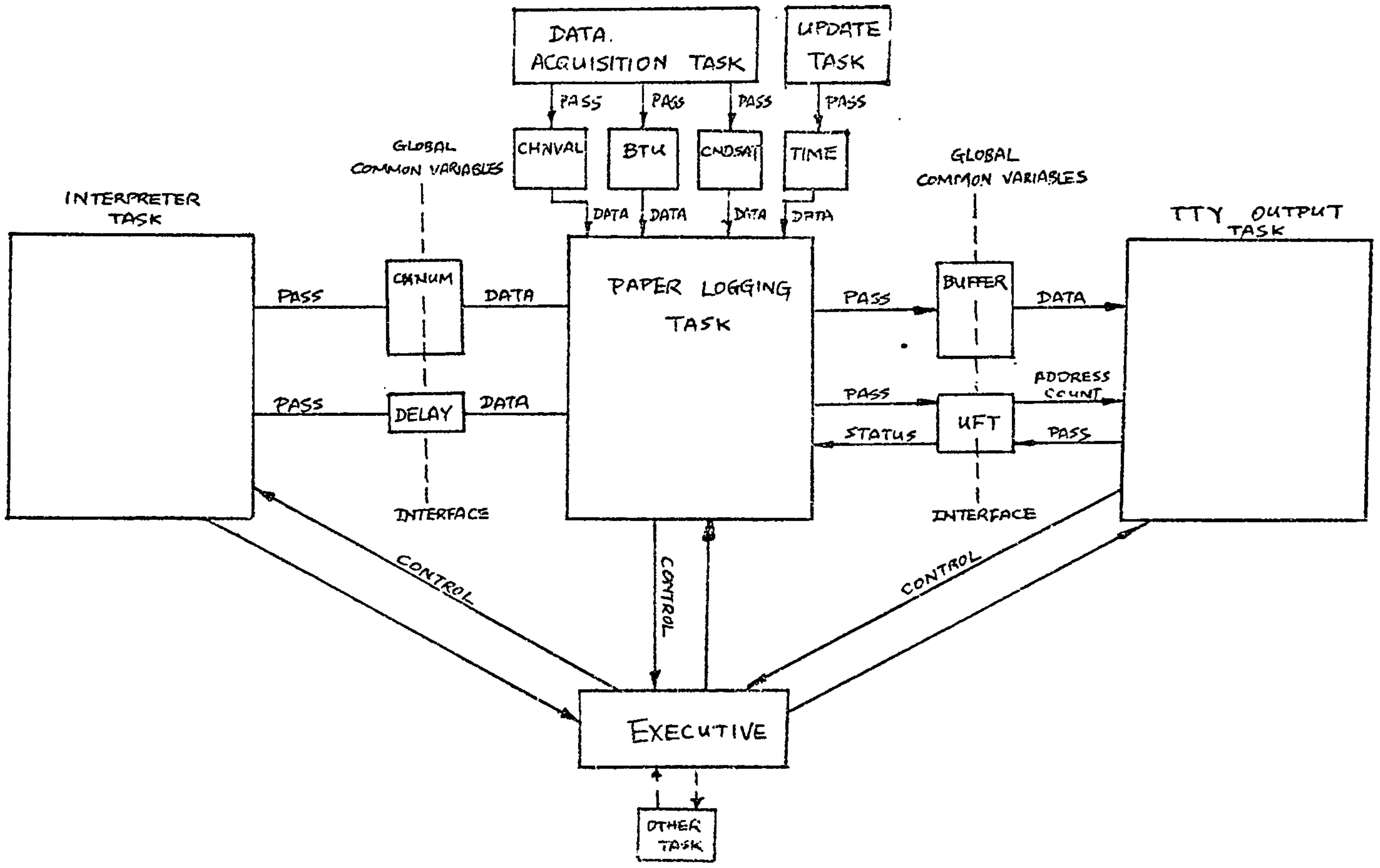


Figure 4-17 Interface block diagram of paper logging task

- Passes data received to the interpreter task through the input buffer INPBUF.
- Transmits to the master processor a status word indicating the mode of operation of the TCU, the existence of communication errors, and the verity of the received command.
- Fetches data from the buffers specified by the interpreter task and transmits them in appropriate format to the master processor.

The format of the block of characters transferred between processors by the protocol task is shown in Figure 4-18. The flow chart of the protocol task is shown in Figure 4-19, and the block diagram showing the task interfaces is presented in Figure 4-20.

4.2 Operating System

The operating system is the central co-ordinator of all program activities and supports the concurrent execution of system tasks. It consists of a real-time executive, an update task, an initialization task, a terminal input task, and a terminal output task.

4.2.1 Executive

The real-time executive is the core element of the operating system. It schedules the execution of all tasks in a round-robin fashion and provides the following services:

- Activate a task
- Relinquish a task

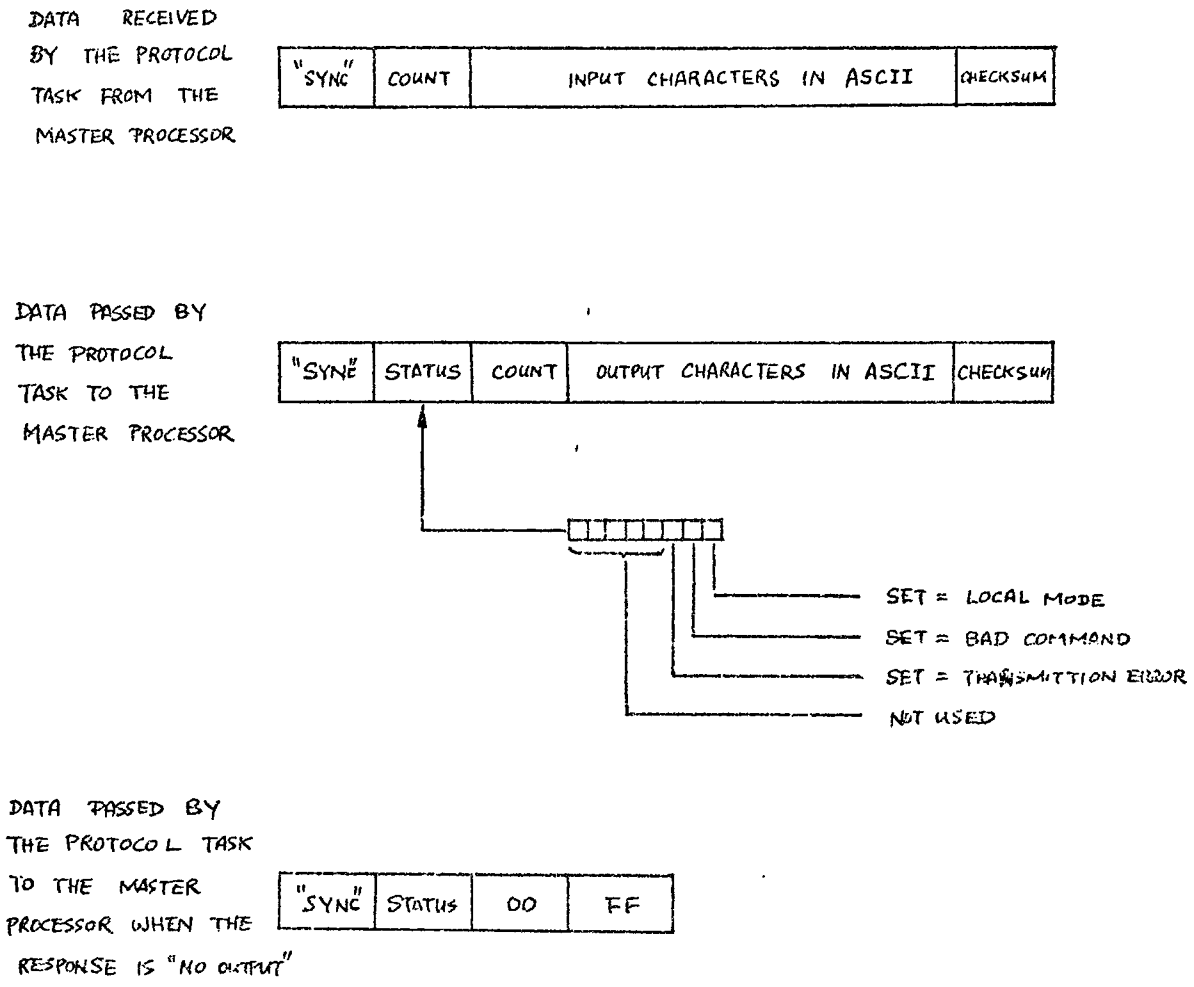

DATA RECEIVED BY THE PROTOCOL TASK FROM THE MASTER PROCESSOR
"SYNC"
COUNT
INPUT CHARACTERS IN ASCII
CHECKSUM
DATA PASSED BY THE PROTOCOL TASK TO THE MASTER PROCESSOR
"SYNC"
STATUS
COUNT
OUTPUT CHARACTERS IN ASCII
CHECKSUM
SET = LOCAL MODE
SET = BAD COMMAND
SET = TRANSMITTION ERROR
NOT USED
DATA PASSED BY THE PROTOCOL TASK TO THE MASTER PROCESSOR WHEN THE RESPONSE IS "NO OUTPUT"
"SYNC"
STATUS
00
FF

Figure 4-18 Receiving and sending sequence between the protocol task and the master processor

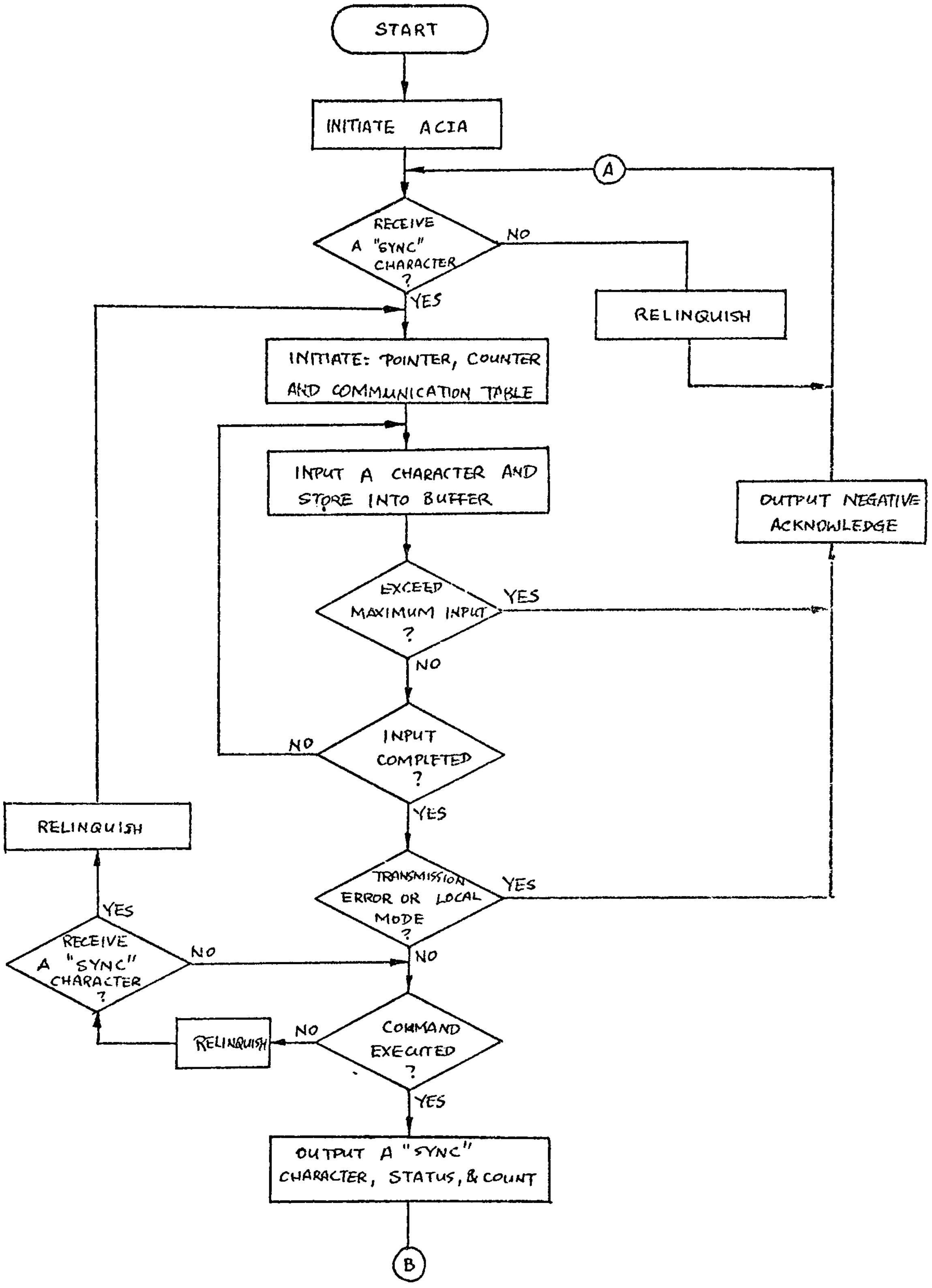

START
INITIATE ACIA
A
RECEIVE A "SYNC" CHARACTER ?
NO
RELINQUISH
YES
INITIATE: POINTER, COUNTER AND COMMUNICATION TABLE
INPUT A CHARACTER AND STORE INTO BUFFER
OUTPUT NEGATIVE ACKNOWLEDGE
EXCEED MAXIMUM INPUT ?
YES
NO
INPUT COMPLETED ?
NO
YES
RELINQUISH
TRANSMISSION ERROR OR LOCAL MODE ?
YES
NO
YES
RECEIVE A "SYNC" CHARACTER ?
NO
COMMAND EXECUTED ?
NO
RELINQUISH
YES
OUTPUT A "SYNC" CHARACTER, STATUS, & COUNT
B

Figure 4-19 Flow chart of protocol task

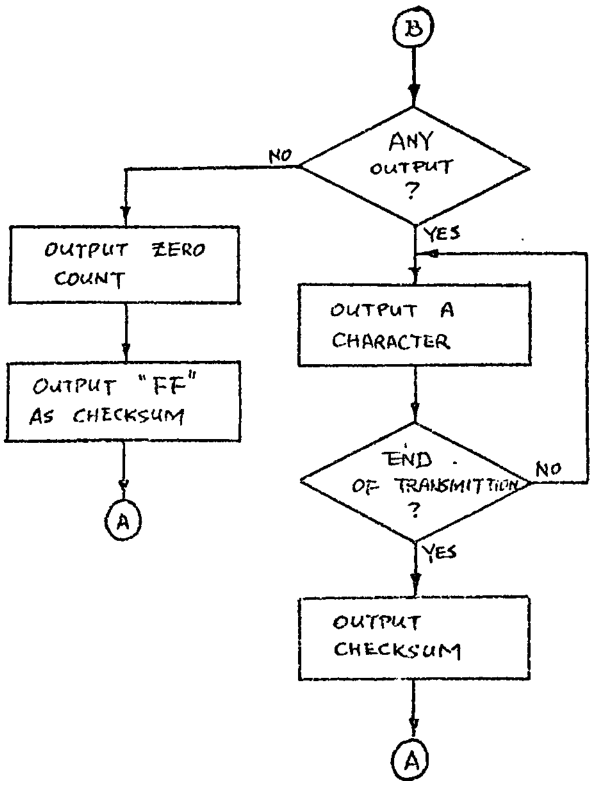


Figure 4-19(cont.) Flow chart of protocol task

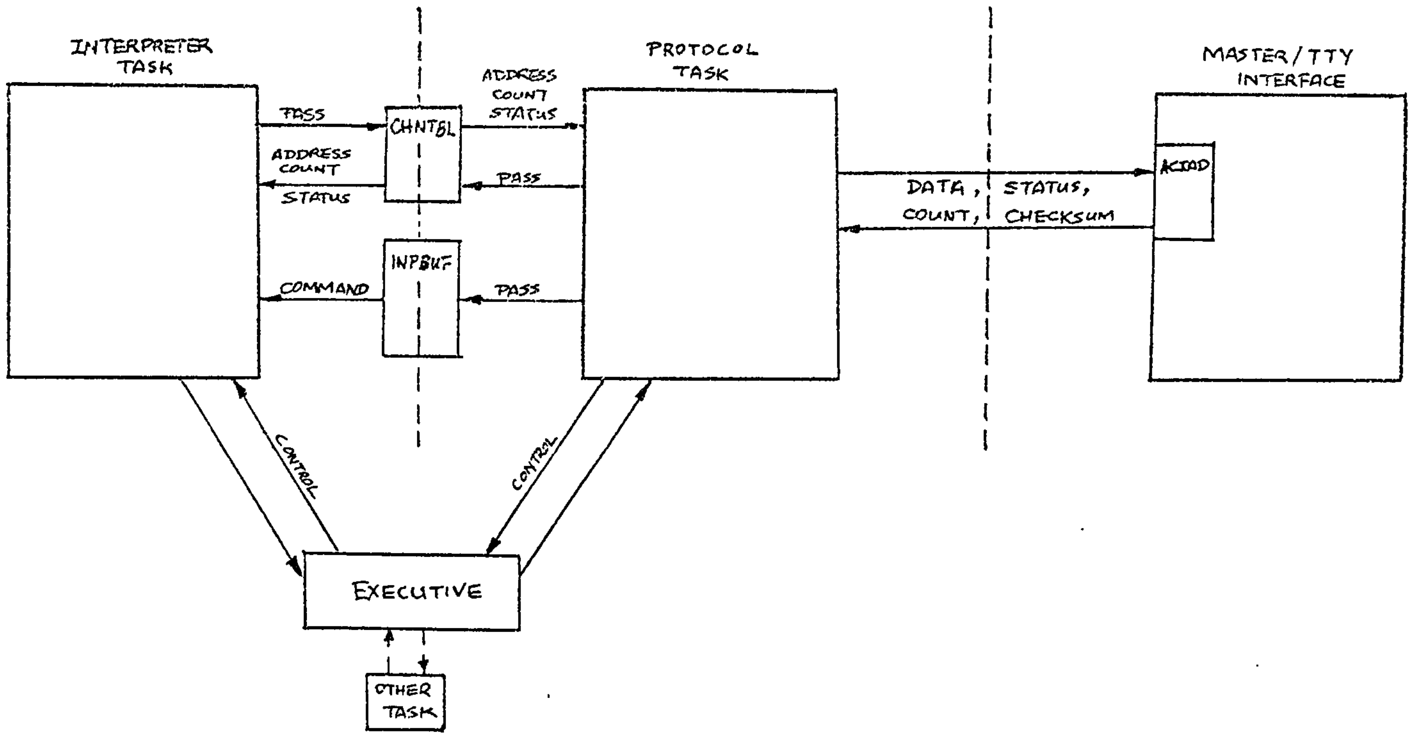


Figure 4-20 Interface block diagram of protocol task

- Exit a task
- Delay a task
- Abort a task
- Multiply two binary numbers
- Input a data buffer
- Output a data buffer

Communication between each user task and the executive is accomplished through the A,B, and X registers. Register A contains the executive service request code, registers B and X contain relevant data and buffer address respectively associated with the function requested.

Control is passed sequentially to each task in the order of its assigned task number among the tasks that are active. The assignment of task numbers in the TCU is as follows:

0	Interpreter Task
1	TTY Input Task
2	TTY Output Task
3	Update Task
4	Protocol Task
5	Set Point Task
6	Data Acquisition Task
7	Temperature Averaging Task
8	Display Task
9	Keyboard Task
10	Debug Task

11 Paper Logging Task

The flow chart of the executive is shown in Figure 4-21. Communication between the executive and the operating system supporting tasks is shown in Figure 4-22.

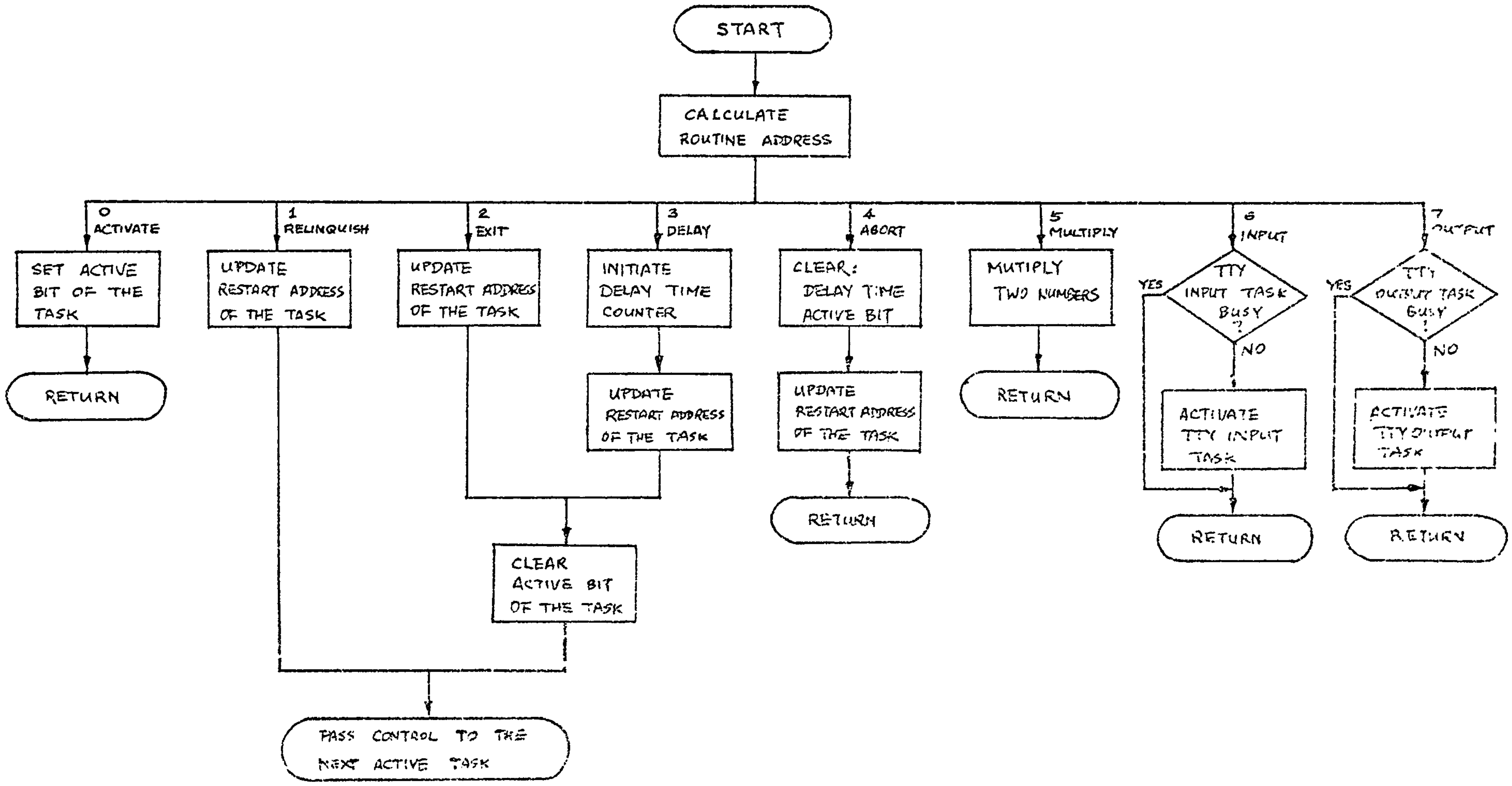
4.2.2 Update Task

The update task is the task most closely associated with the executive. It contributes to the implementation of the time update function.

It performs the following functions periodically:

- Updates the system operational mode and passes this information to the interpreter task through the common variable OPTMOD.
- Updates the time of the day in the TIME common block every second and outputs a change of state to an LED on the front panel through a bit of the PIACA.
- Updates the delay time stored in the DLYTIM block of suspended tasks.
- Updates the task status stored in the TSKSTS block of suspended tasks.
- Updates the date of the year in the DAY common block.

The flow chart of the update task is shown in Figure 4-23.



START
CALCULATE ROUTINE ADDRESS
0 ACTIVATE
1 RELINQUISH
2 EXIT
3 DELAY
4 ABORT
5 MULTIPLY
6 INPUT
7 OUTPUT
SET ACTIVE BIT OF THE TASK
RETURN
UPDATE RESTART ADDRESS OF THE TASK
UPDATE RESTART ADDRESS OF THE TASK
INITIATE DELAY TIME COUNTER
UPDATE RESTART ADDRESS OF THE TASK
CLEAR: DELAY TIME ACTIVE BIT
UPDATE RESTART ADDRESS OF THE TASK
RETURN
MUTIPLY TWO NUMBERS
RETURN
TTY INPUT TASK BUSY ?
YES
NO
ACTIVATE TTY INPUT TASK
RETURN
TTY OUTPUT TASK BUSY ?
YES
NO
ACTIVATE TTY OUTPUT TASK
RETURN
CLEAR ACTIVE BIT OF THE TASK
PASS CONTROL TO THE NEXT ACTIVE TASK

Figure 4-21 Flow chart of the executive

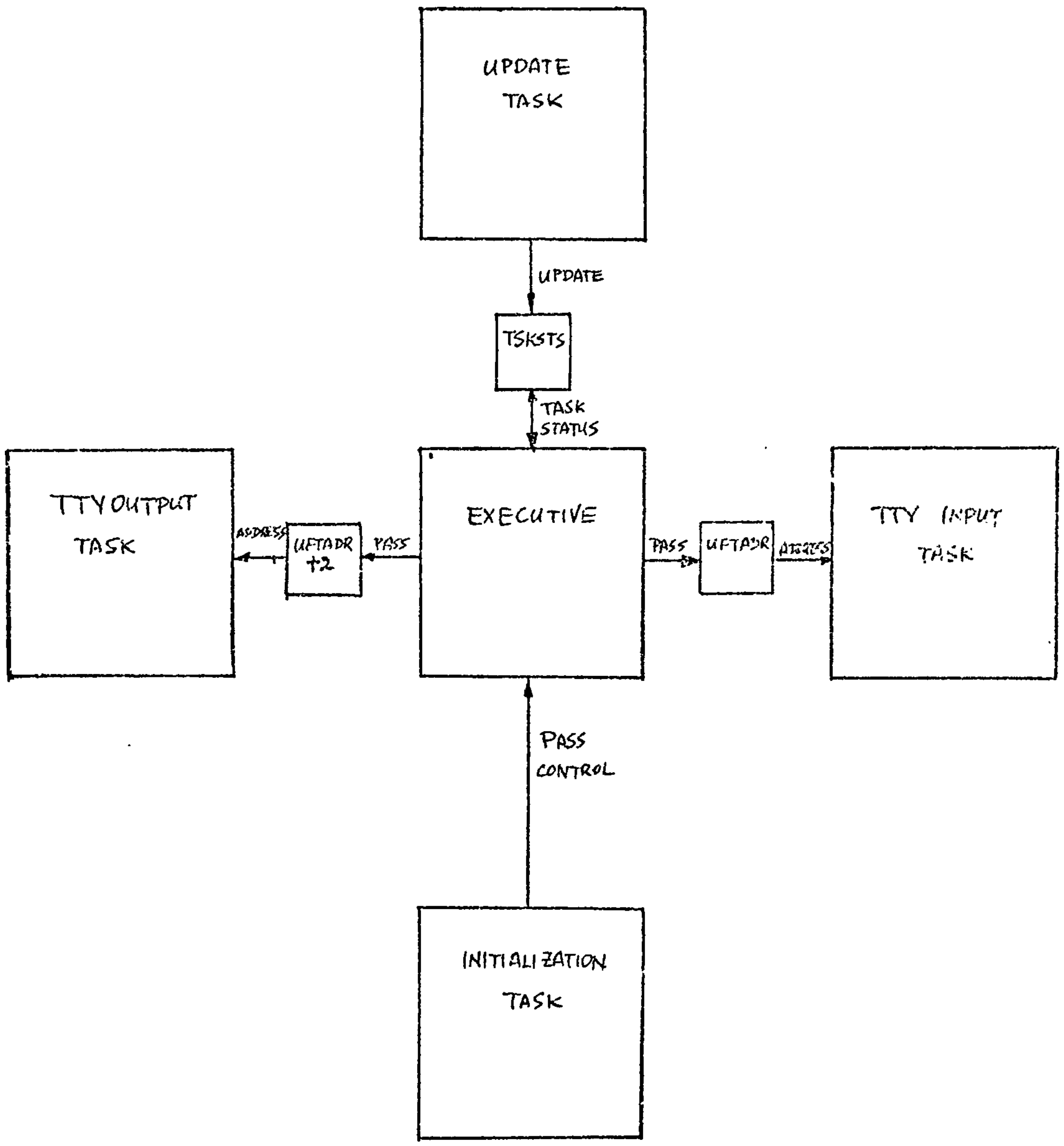


Figure 4-22 Communication between tasks of the operating system

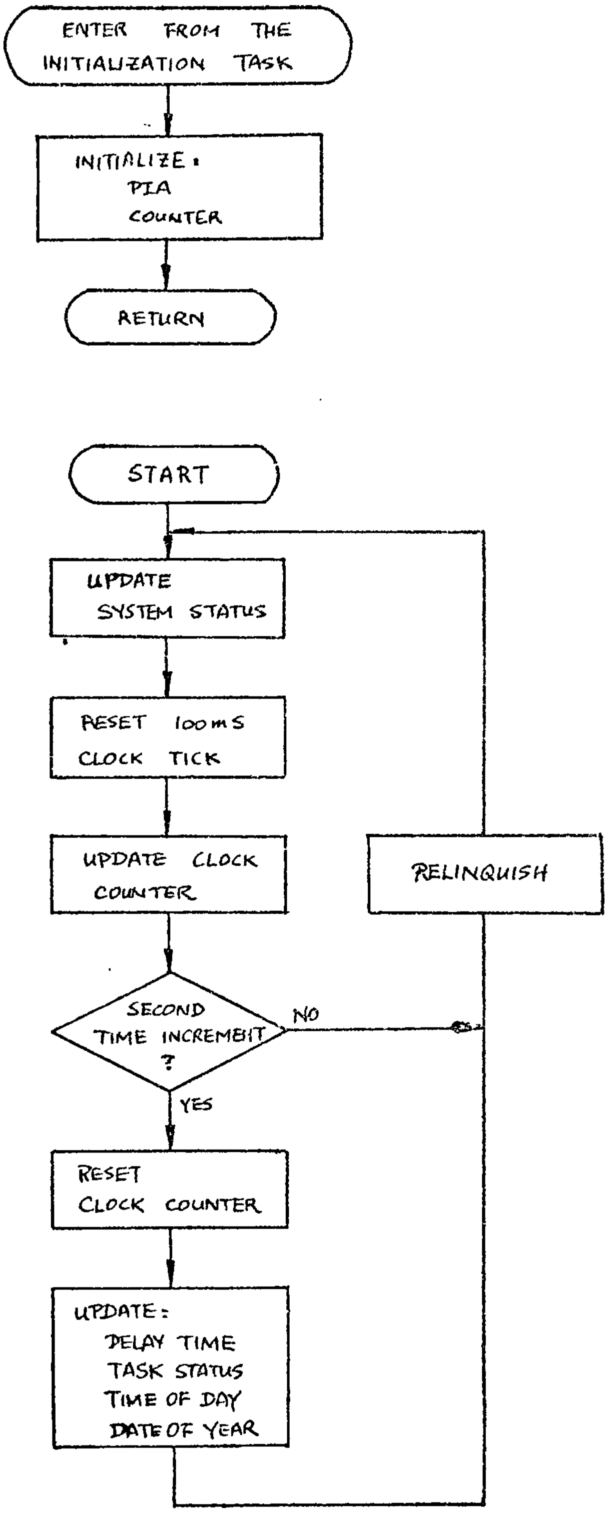


Figure 4-23 Flow chart of update task

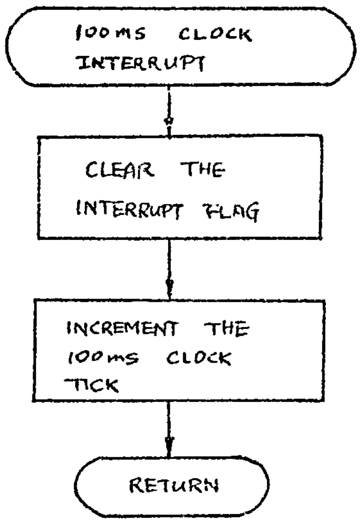


Figure 4-23(cont.) Flow chart of update task

4.2.3 Initialization Task

This task implements the hardware and software initialization of the system when power is turned on or the master reset button is depressed.

It initializes the stack pointer and the restart address of each task, and sets all scratch locations to zero. It also transfers control to the initialization routine of each individual task before passing control to the executive.

The flow chart of the initialization task is shown in Figure 4-24.

4.2.4 TTY Input Task

This task contributes to the implementation of the operator communication function. It fetches command from the ACIA data register and passes to the interpreter task through the input buffer INBUF. The TTY input task is activated only on demand by the interpreter task.

The flow chart of the TTY input task is shown in Figure 4-25.

4.2.5 TTY Output Task

This task also contributes to the implementation of the operator communication function. It passes data from the output buffer of a task that requests this executive service to the ACIA data register. The TTY output task is activated only on demand.

The flow chart of the TTY output task is shown in Figure 4-26.

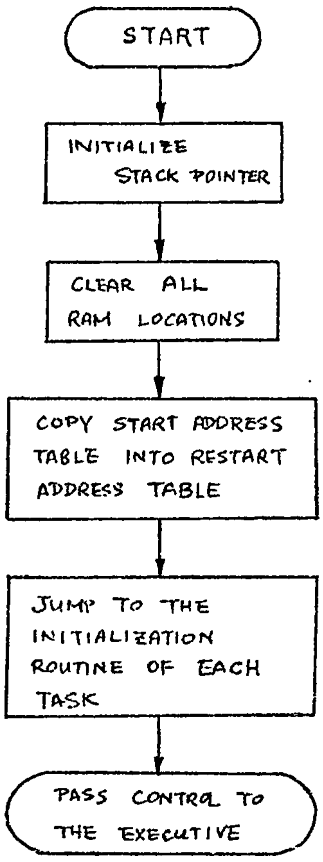

START
INITIALIZE STACK POINTER
CLEAR ALL RAM LOCATIONS
COPY START ADDRESS TABLE INTO RESTART ADDRESS TABLE
JUMP TO THE INITIALIZATION ROUTINE OF EACH TASK
PASS CONTROL TO THE EXECUTIVE

Figure 4-24 Flow chart of initialization task

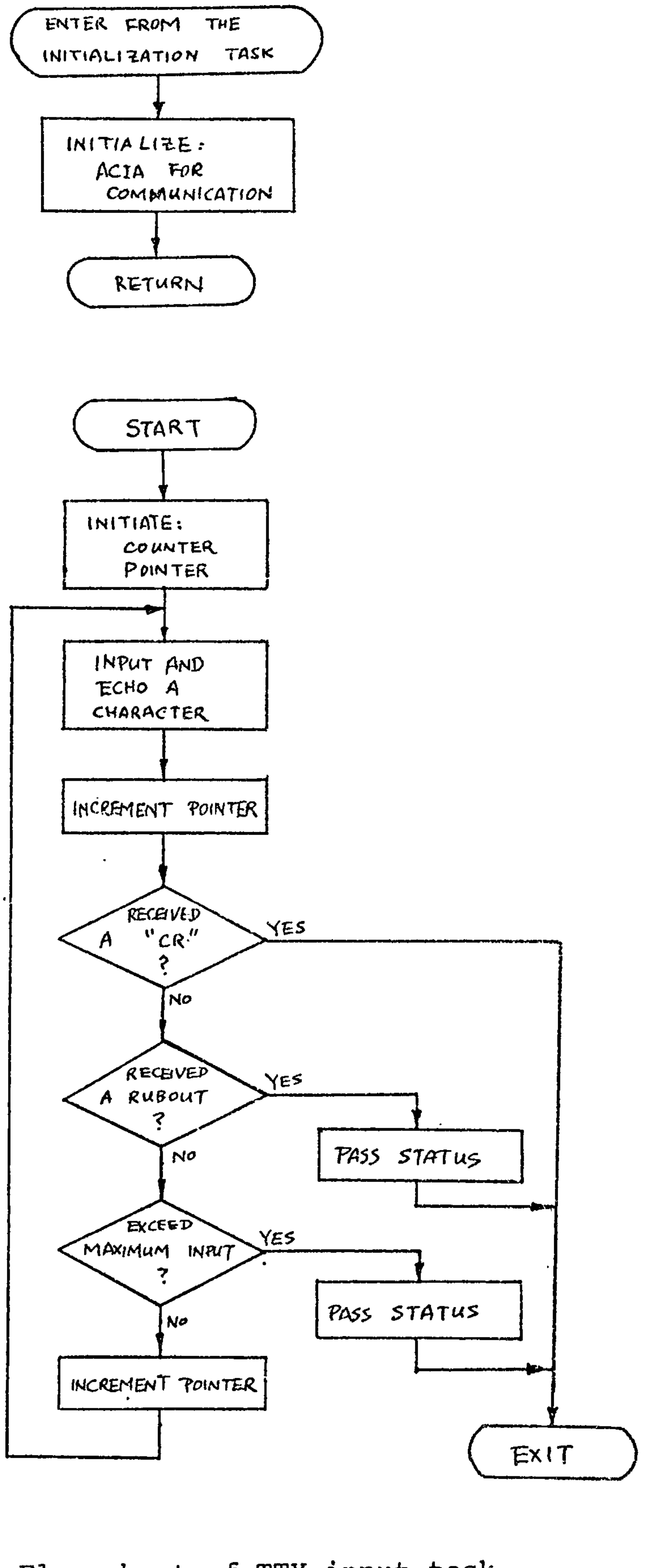


Figure 4-25 Flow chart of TTY input task

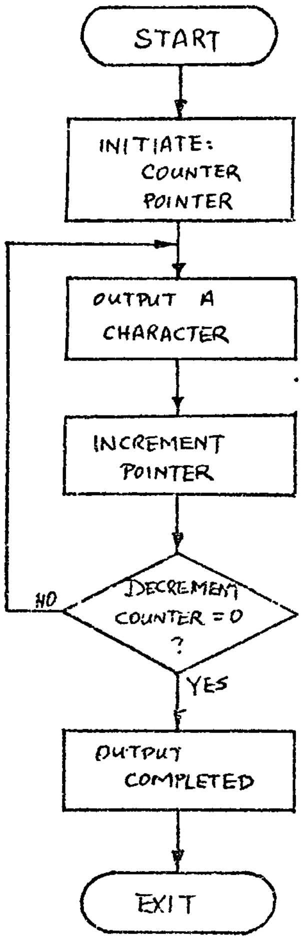


Figure 4-26 Flow chart of TTY output task

4.3 Communication Between Tasks

The transfer of commands, data, address and status between tasks is achieved through a well defined set of global common variables which includes all variables identified in the block diagrams presented in the previous section. This section describes these global common blocks as related to their size, types, and functions. The types of variables transferred are binary, BCD, ASCII, and logical.

A description of the global common variables used for inter-task communication is presented in terms of the following attributes: (1) names of the tasks, (2) name of the global common variable, (3) size of the global common variable, (4) type of the global common variable, and (5) description of the global common variable. The variables and their attributes are presented in the following format:

(1) Name of the tasks

(2) Name of the variable	(3) Size of the variable	(4) Type of the variable

(5) Description of the variable

Interpreter Task/Data Acquisition Task

CNDSAT	4 bytes	BCD

- 4 bytes used for pound of steam condensate accumulation.
- The interpreter task initiates and reads from these locations. The data acquisition task updates the values stored in these locations.

- CNDSAT contains the two most significant digits.
- CNDSAT+3 contains the two least significant digits.

<u>CHNVAL</u>	80 bytes	BCD

- 2 bytes per channel used for storing channel values.
- The interpreter task reads from these locations. The data acquisition task writes into these locations.

<u>CHNALM</u>	40 bytes	Bit map

- 1 byte per channel, contains the channel alarm status.
- Bit 7, 1, and 0 are set by the data acquisition task to indicate no limit check, a high limit alarm, and a low limit alarm respectively.
- The interpreter task reads from these locations.

<u>LMTABL</u>	160 bytes	BCD

- 4 consecutive bytes per channel, contains the channel limits. The first two bytes contain the low limit and the next two higher bytes contain the high limit.
- The interpreter task reads from and writes into these locations. The data acquisition task reads from these locations.

<u>DAINTC</u>	5 bytes	Flag bits

- Second byte used for status storage, 1 indicates the existence of an alarm in at least one byte of CHNALM; 0 indicates that no alarm condition exists.

- Other bytes are not used.

<u>Interpreter Task/Temperature Averaging Task</u>

<u>BTU</u>	4 bytes	BCD

- 4 bytes used to accumulate the value of ton-hours of chilled water.
- The interpreter task sets the initial values and reads these locations.
- The data acquisition task updates the values stored in these locations.
- BTU contains the two most significant digits.
- BTU+3 contains the two least significant digits.

<u>Interpreter Task/Set Point Task</u>

<u>SPVALU</u>	32 bytes	Binary

- 1 byte per set point used for storage of the corresponding set point value.
- Value ranges from 0 to 100.
- The interpreter task reads from these locations. The set point task writes into these locations.

<u>MODTAB</u>	4 bytes	Bit map

- 1 bit per set point used to store its control mode.
- 1 denotes automatic control mode. 0 denotes

manual control mode.

- The first channel's status is stored in bit 7 of MODTAB. The 32nd channel's status is stored in bit 0 of MODTAB+3.
- The interpreter reads from these locations. The set point task writes into these locations.

<u>SPALRM</u> 4 bytes Bit map

- 1 bit per set point used to store its loop integrity alarm status.
- 1 denotes existence of an alarm condition. 0 denotes that the control loop is operational.
- Other description are the same as MODTAB's.

<u>SPINCM</u> 5 bytes Packed format

- 5 bytes for inter-communication between the two tasks.
- For SPINCM, bits 7, 1, and 0 are set by the interpreter task to signal for a job request, the adjustment of a set point and the return of a set point to manual mode respectively. This byte is cleared by the set point task after it processes the request.
- For SPINCM+1, bits 7, 4, 3, 2, 1, and 0 are set by the set point task for alarm, power failure for set points 25-32, power failure for set points 17-24, power failure for set set points 9-16, power failure for set points

1-8, and loop integrity alarm respectively. The interpreter task reads these locations. The set point task writes into these locations.

- For SPINCM+2, set point control loop number is specified in this location in binary form. The interpreter task writes into and the set point task reads from this location. The channel number ranges from 0 to 31.
- For SPINCM+3, set point control value is passed from the interpreter task to the set point task in this location as a binary number. The channel value ranges from 0 to 100.
- For SPINCM+4, it is not used.

<u>Interpreter Task/Paper Logging Task</u>

<u>CHNUM</u> 6 bytes Packed format

- The interpreter task passes six channel numbers in binary to the paper logging task if bit 7 of CHNUM is reset.
- If bit 7 of CHNUM is set, the interpreter task requests the paper logging task to print all channel reports periodically, other bytes are then unused.

<u>DELAY</u> 1 byte Packed format

- If bit 7 of CHNUM is set, the interpreter task writes into DELAY which is used to stored the delay time for the paper logging task;

otherwise, the interpreter task does not write into this location.

- If bit 7 of DELAY is set, the delay time is in minutes, and it is stored in the rest of the bits; if bit 7 is reset, the delay time is in seconds.

Interpreter Task/Protocol Task

CHNTBL 8 bytes Packed format

- Eight consecutive bytes. The first four bytes are used to output, and the last four bytes are used to input when the TCU communicates with the CCU.
- CHNTBL is used as an output flag. Bit 7 is set by the interpreter task if it requests output. It is reset if there is no output request.
- CHNTBL+1, CHNTBL+2 are used by the interpreter task to pass an output buffer address to the protocol task.
- CHNTBL+3 is used by the interpreter to pass the number of output bytes to the protocol task.
- CHNTBL+4 is used as an input flag. Bit 7 is set by the protocol task when input from the CCU is completed. It is reset by the interpreter task if the command received is executed.

- CHNTBL+5, CHNTBL+6 are used by the protocol task to pass the input buffer address to the interpreter task.
- CHNTBL+7 is used by the protocol task to pass the number of input bytes to the interpreter task.

Interpreter Task/Update Task

TIME	3 bytes	BCD

- The 3 bytes are used for storing time of day in hours, minutes and seconds respectively.
- The interpreter task writes into and reads from these locations. The update task only writes into these locations.

DAY	2 bytes	BCD

- They are used to store the julian date of the year.
- The interpreter task writes into and reads from these locations. The update task only writes into these locations.

Interpreter Task/TTY Input Task

INBUF	22 bytes	ASCII

- The TTY input task passes the user command to the interpreter task through the buffer INBUF.

INUFT	4 bytes	Packed format

- 4 bytes used for communication between the two tasks.

- INUFT is reset by the TTY input task if a request for input by the interpreter task is granted. Bit 7 is set by the TTY input task if input is completed. Bits 1 or 0 is set if a rubout command is received.
- INUFT+1, INUFT+2 are used by the interpreter task to pass the starting location of INBUF in binary to the TTY input task.
- INUFT+3 is used by the interpreter task to pass the maximum number of input character in binary to the TTY input task.

Interpreter Task/TTY Output Task

OUTBUF	50 bytes	ASCII

- The interpreter passes the message it wants to output to the TTY output task through the buffer OUTBUF.

OUTUFT	4 bytes	Packed format

- 4 bytes used for communication between the two tasks.
- OUTUFT is reset by the TTY output task when an output request of the interpreter task is granted; otherwise bit 6 is set. Byte 7 is set by the TTY output task when the output request is completed. The interpreter task reads from this location.
- OUTUFT+1, OUTUFT+2 are used by the interpreter

task to pass the starting location of the OUTBUF in binary to the TTY output task.

- OUTUFT+3 is used by the interpreter task to pass the number of output character in binary to the TTY output task.

Keyboard Task/Display Task

REQJOB	1 byte	Binary

- 1 byte used by the keyboard task to pass a command code to the display task.
- 1 means channel report.
- 2 means ton-hours of chilled water report.
- 3 means pound of steam condensate report.
- 4 means time report.

CHNNO	1 byte	Binary

- 1 byte used by the keyboard task to pass a channel number to the display task.

CHNNO1	1 byte	BCD

- 1 byte used by the keyboard task to pass a channel number to the display task.

Display Task/Data Acquisition Task

CNDSAT	4 bytes	BCD

- See Interpreter Task/Data Acquisition Task.
- The display task fetches data from these location.

CHNVAL	80 bytes	BCD

- See Interpreter Task/Data Acquisition Task.

- The display task fetches data from these locations.

<u>Display Task/Temperature Averaging Task</u>

<u>BTU</u>	4 bytes	BCD

- See Interpreter Task/Temperature Averaging Task.
- The display task fetches data from these locations.

<u>Display Task/Update Task</u>

<u>TIME</u>	3 bytes	BCD

- See Interpreter Task/Update Task.
- The display task fetches data from these locations.

<u>Paper Logging Task/Data Acquisition Task</u>

Same as Display Task/Data Acquisition Task.

<u>Paper Logging Task/Temperature Averaging Task</u>

Same as Display Task/Temperature Averaging Task.

<u>Paper Logging Task/Update Task</u>

Same as Display Task/Update Task.

<u>Update Task/Executive</u>

<u>TSKSTS</u>	12 bytes	Flag bit

- 12 consecutive locations used to store the task status of each user task.
- MSB set means the corresponding task is active. MSB reset means the corresponding task is inactive.

- The update task updates a task's status when its delay time is up.
- The executive updates a task's status when requested by a user task. It also checks the tasks' status and passes control to the active tasks in round-robin fashion.

<u>TTY Input Task/Executive</u>

<u>UFTADR</u>	2 bytes	Binary

- The executive passes the starting address of the user file table of the user task which requests input service to the TTY input task.

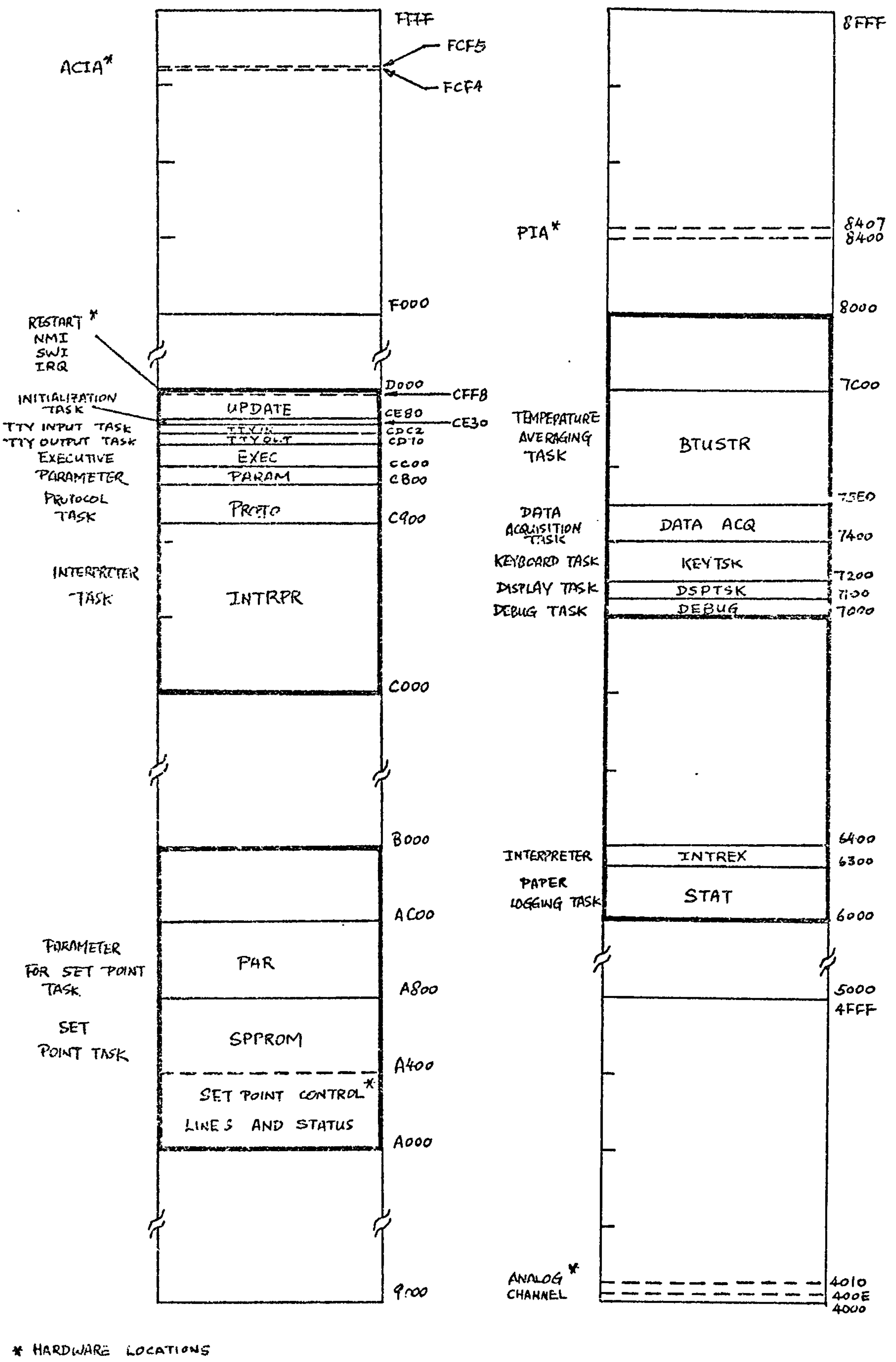
<u>TTY Output Task/Executive</u>

<u>UFTADR+2</u>	2 bytes	Binary

- The executive passes the starting address of the user file table of the user task which requests output service to the TTY output task.

4.4 <u>Memory Map</u>

The utilization of memory for program storage and peripheral devices is presented in Figure 4-27. The types of memory used, i.e. ROM or memory mapped I/O are also indicated in the diagram. The global common variables and the local scratch variables of the task programs use RAM locations from 0 to 3FF(hexadecimal).



* HARDWARE LOCATIONS

Figure 4-27 Memory map of tasks and hardware locations

CHAPTER 5

HARDWARE ARCHITECTURE

The specification and design of hardware components play an important role in achieving the system design objectives. Hardware components are required to support program execution and to translate digital data into signals compatible with peripheral equipment. Discussion of the hardware components will focus on two major interfaces: the field equipment interface (electrical signals) and the software interface (memory image of the hardware as viewed by a task).

5.1 Hardware Module Description

The system hardware configuration is illustrated in Figure 5-1 and includes the following modules: (1) the basic digital unit, including the processor unit, clock, memories and power supply, (2) the set point control interface, (3) the analog input interface, (4) the keyboard and display interface, and (5) the Master/TTY interface.

5.1.1 Basic Digital System Hardware

This module is the core of the TCU system hardware. It supports the operation of all other hardware interface modules. A monoboard microcomputer 1A, a 16K ROM board and a chassis(M68MMLC1) made by Motorola, Inc. are used to implement the basic system.

The monoboard microcomputer 1A includes: a MC6800

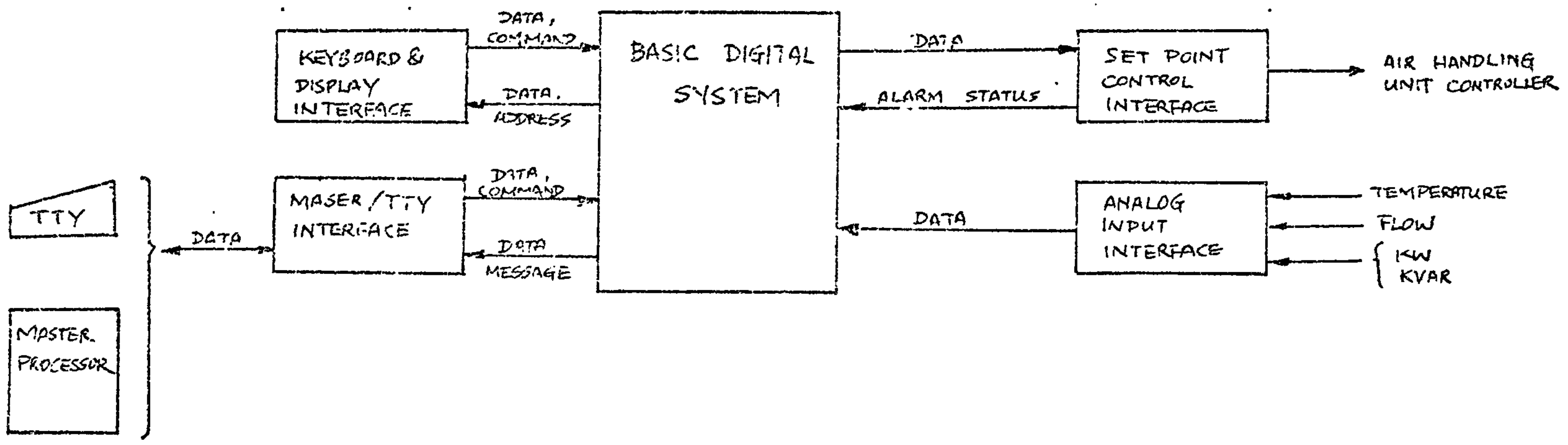


Figure 5-1 Block diagram of the hardware architecture

microprocessing unit with associated clock oscillator, power on reset timer, memory decoding logic, 1K bytes of RAM, 4K bytes of ROM sockets, one RS-232 compatible interface, two programmable peripheral interface adapters, and bus drivers to support the interfaces with other modules.

The 16K ROM board is used to store firmware programs and system parameters.

The chassis provides the necessary supply voltages +5V, +12V, and -12V, and the connectors for intermodule communication.

5.1.2 Set Point Control Interface

The set point control interface is implemented with analog output cards, analog output termination chassis, current-to-air pressure transducers (Fairchild T-5100) and pressure selectors. The elements of this interface are illustrated in Figure 5-2.

The analog output card accepts binary data ranging from 0 to 200 and converts it into a 4-20 mA dc current that drives a current-to-pressure transducer to produce an output pressure of 3-15 psi. This pressure is used to adjust the set point temperature of the air handling unit controller. A high-pressure selector provides the mechanism for operation of the controller in either manual or automatic mode.

Each analog output card provides eight digitally controlled current loops, and up to four cards may be configured in a system. Therefore a total of thirty-two set

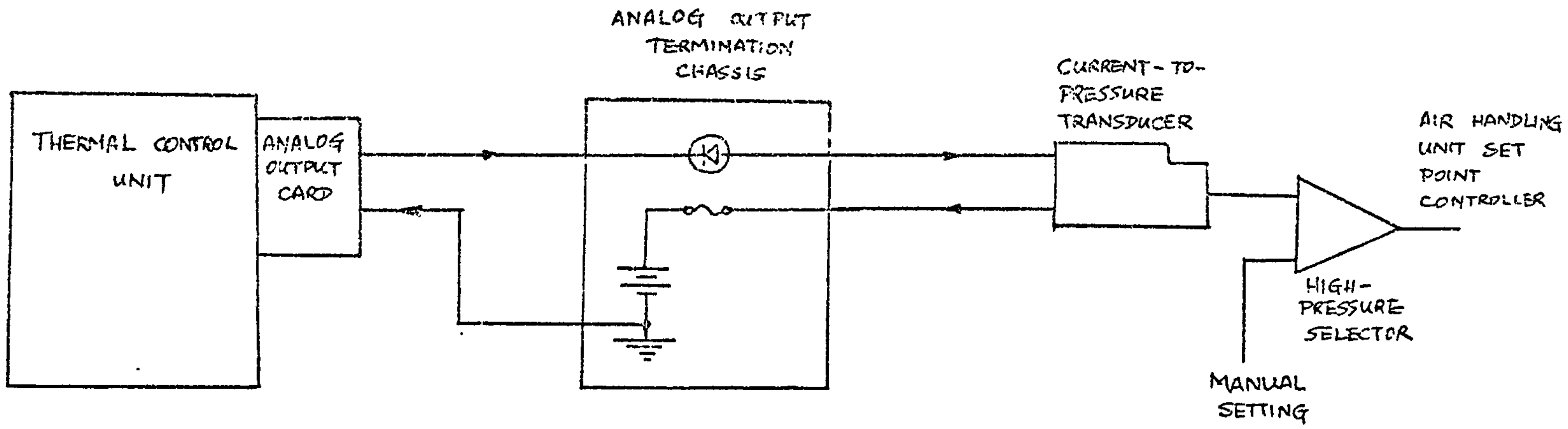

ANALOG OUTPUT TERMINATION CHASSIS
CURRENT-TO-PRESSURE TRANSDUCER
AIR HANDLING UNIT SET POINT CONTROLLER
THERMAL CONTROL UNIT
ANALOG OUTPUT CARD
HIGH-PRESSURE SELECTOR
MANUAL SETTING

Figure 5-2 Set point control interface

point controls may be configured in a system. The current of each control loop is monitored by the card to detect an open loop condition which will result in loss of control. The manual setting is set to a default value when the HVAC system is under automatic control.

Each analog output termination chassis has independent power supply to provide the control current. Each control line is fuse protected and has an LED to indicate loop continuity, its degree of illumination also gives an indication of the approximate amount of current flowing in the control line. A total of sixteen control lines can be accomodated in each chassis. The operational status of the termination chassis power supply is monitored by the set point task.

5.1.3 Analog Input Interface

The analog input interface is implemented with analog-to-digital card (Burr Brown MP7208), multiplexing module, analog input voltage conversion chassis, temperature-to-current transducer, flow-to-current transducer (Foto Flow BPN 800), and power-to-current transducer (Scientific Columbus DL342K5A2-6070). The elements of this interface are illustrated in Figure 5-3.

The temperature-to-current transducer converts temperature from 0 to 564.5°F to a current to 0.2244 mA. It has a time constant of 1 minute, and an accuracy of $\pm 0.5^{\circ}$F.

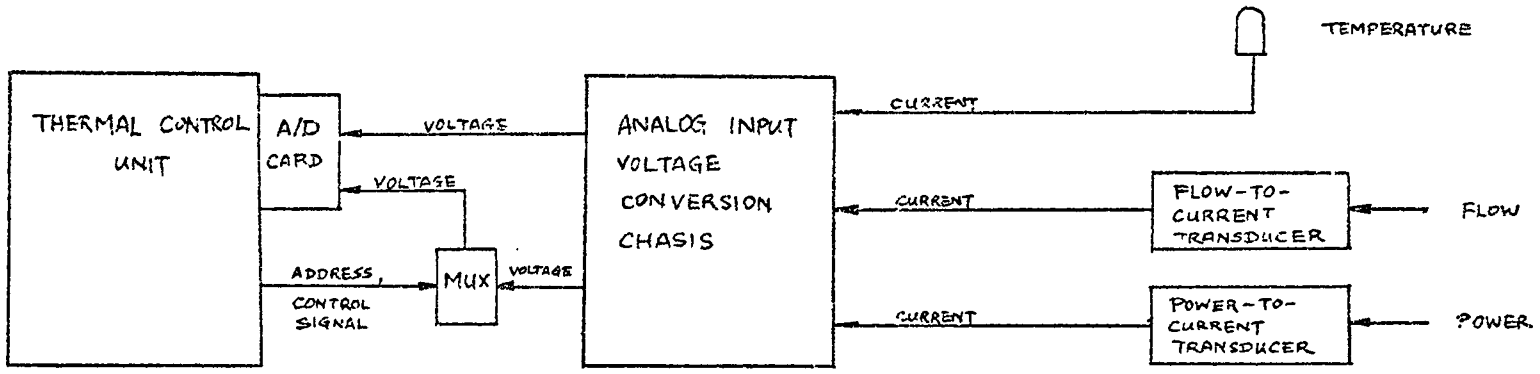


Figure 5-3 Analog input interface

The flow-to-current transducer converts water flow rate from 0 to 50 GPM to a current of 4 to 20 mA with an accuracy of ±0.5%. The power-to-current transducer converts power from 0 to 1500 kilowatts to a current of 0 to 1 mA with an accuracy of ±0.1%.

All measuring transducers are located remotely from the TCU. The currents from the transducers are converted into equivalent voltages in the range 0-5V at the analog input conversion chassis before being converted at the analog-to-digital card.

The analog-to-digital card has eight differential input channels. The first six channels are used for direct input from the transducers, so they are considered as the primary group. The last two channels are used for channel expansion. A multiplexing module which accepts channel address selection from the B side of PIA2 expand channel 8 into 32 channels for temperature data acquisition. This group of channels is considered as the secondary group. Channel 7 is reserved for further system expansion.

The analog-to-digital card has twelve bit resolution s for each channel. Data are read from the TCHNL locations by the data acquisition and temperature averaging tasks.

5.1.4 Keyboard and Display Interface

The keyboard and display interface comprises the keyboard pad, the keyboard and display card, assorted switches, LED indicators, and the two PIA's on the microcomputer 1A. Figure 5-4 illustrates the structure of the keyboard and display interface.

This interface interacts mainly with the keyboard task and the display task. It also supports the update task and the data acquisition task.

The following capabilities are provided by this card:

- Accepts key strokes from the keyboard pad, encodes the signals into binary data ranging from 0 to F(hexadecimal) and passes to the keyboard task for interpretation through the location PIAA.
- Signals the keyboard task for data entry through the location PIAA+1.
- Contains eight seven-segment LED displays each capable of displaying 0 to F(hexadecimal) as well as a decimal point.
- Accepts data, data position and control from the display task through the PIAA and PIAB locations, and displays the data on the LED's.
- Monitors the system operation mode, the TTY power supplies, the analog input voltage conversion chassis power supplies; and reports their status to the update task through location PIADA.
- Accepts multiplexer address selections from

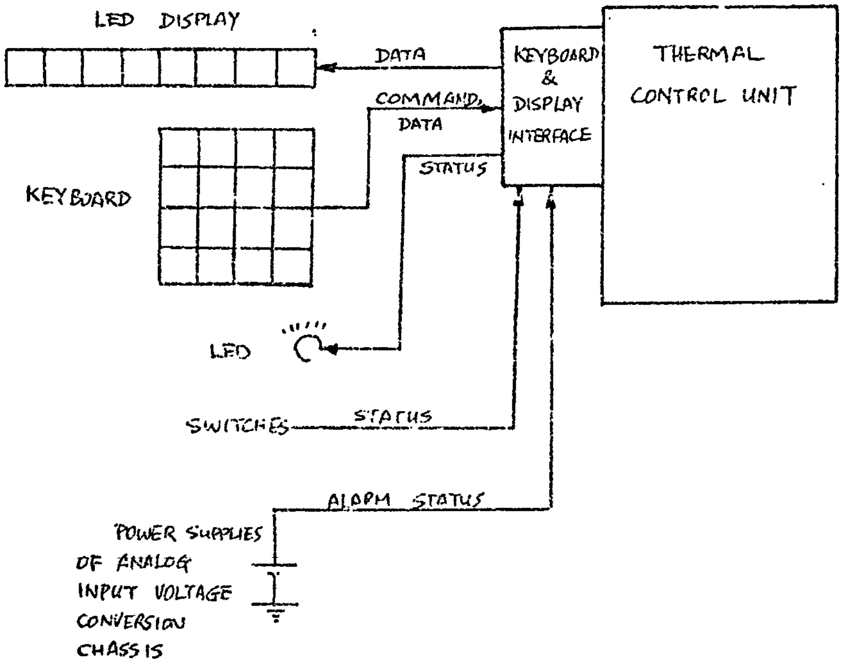


Figure 5-4 Keyboard and display interface

the data acquisition task and drives the LED conversion indicator on the front panel through location PIADB.

- Drives an LED to indicate whether local or remote operation has been selected. Drives the alarm and the time LED indicators on the front panel with signals from CA2 of PIA2 and CB2 of PIA2 as specified by the interpreter task and the update task respectively.

5.1.5 Master/TTY Interface

The master/TTY interface consists of the ACIA and the bit rate generator both on the microcomputer 1A, and a RS-232 to 20 mA current loop adapter and a 20 mA current loop to RS-232 adapter to interface with the master processor of TTY terminal remotely. Figure 5-5 illustrates the master/TTY interface.

The bit rate generator is set to 4800 bps and is divided by 16 at the ACIA to give a transmission rate of 300 baud.

5.2 Software Interface of Hardware Modules

The communication between hardware and software is accomplished through memory locations assigned to each hardware peripheral device. A task interacts with a peripheral device by writing into or reading from the assigned locations. This section describes these locations as related

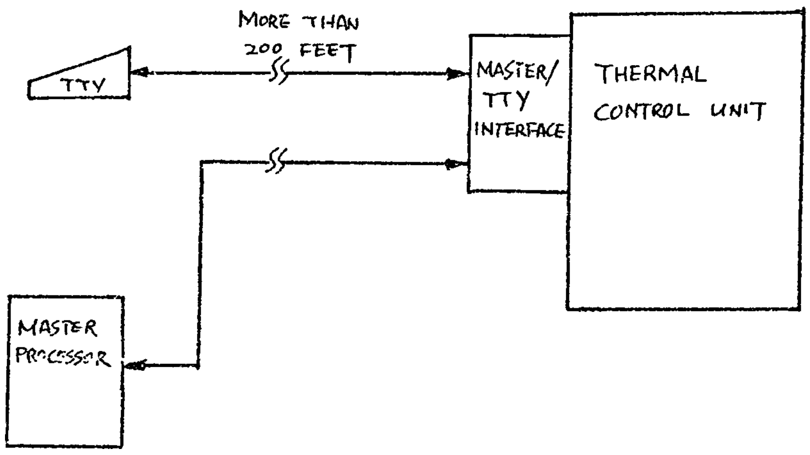


Figure 5-5 Master/TTY interface

to their size, types, and functions. The types of data transferred are either binary or flag bit.

A description of the hardware locations used for software interface is presented in terms of the following attributes: (1) name of the task/name of the hardware interface module, (2) name of the hardware locations, (3) size of the hardware locations, (4) type of data associated with the hardware locations, and (5) description of the hardware locations. The hardware locations and their attributes are presented in the following format:

(1) Name of the task/Name of the hardware interface module

(2) Name of the locations	(3) Size of the locations	(4) Type of data

(5) Description of the locations

Set Point Task/Set Point Card

DAC	32 bytes	Binary

- 32 consecutive locations. One byte associated with each control line. Data are written into these locations by the set point task. The range of data written into a location is from 0 to 200.

LOOP	4 bytes	Bit map

- Each byte provides the loop integrity status of eight loops.
- Bytes corresponding to distinct groups of eight channels are spaced eight addresses

apart.

- The first control line status is read from the MSB of LOOP, and the 32nd control line status is read from the LSB of LOOP+24.
- The set point task reads from these locations.

<u>KEY</u>	4 bytes	Binary

- 8 bytes apart for two consecutive words.
- KEY locations are used to protect against unauthorized writing into the DAC locations. Each KEY location protects 8 consecutive DAC locations.

<u>CARD</u>	4 bytes	Flag bit

- 8 bytes apart for two consecutive words.
- CARD locations are used to detect power supply failures at analog voltage conversion chassis. Each CARD location monitors one power supply. Only the MSB of the word is used.

<u>Data Acquisition Task, Temperature Averaging Task/Analog-to-digital Card</u>

<u>TCHNL</u>	16 bytes	Binary

- 2 bytes per each channel.
- The analog-to-digital card has 12 bits resolution, the 4 MSB are stored in the even address, and the 8 LSB are stored in the next higher location.

- The data acquisition task and the temperature averaging task read from these locations.

<u>MXCHLS</u>	5 bits	Binary

- The 5 MSB of side B of PIA2 are used for multiplexer address selection.
- The data acquisition task writes into these locations.

<u>Keyboard Task/ Keyboard and Display Interface</u>

<u>PIAA</u>	4 bits	Binary

- The keyboard and display interface passes commands and data to the keyboard task through the 4 MSB of side A of PIA1.
- The E key is used to request a channel report.
- The D key is used to request the ton-hour of chilled water report.
- The C key is used to request the pound of steam condensate report.
- The F key is used to request the time of day report.

<u>PIAA+1</u>	1 bit	Flag bit

- The MSB is set whenever a key is depressed.
- This bit is monitored by the keyboard task to detect a new data entry.

<u>Display Task/Keyboard and Display Task</u>

<u>PIAA</u>	3 bits	Binary

- The display task selects a digit element for display through the 3 LSB of side A of PIA1.

PIAB   1 byte   Packed format

- The display task outputs data and control to the keyboard and display interface through this memory location.
- Bit 7 is used for global blanking control.
- Bit 6 is used for decimal point control.
- Bit 5 is used for element blanking control.
- Bit 4 is not used.
- Bit 0 to bit 3 are used to transfer display data in binary.

Update Task/Keyboard and Display Interface

PIADA   1 byte   Packed format

- The update task updates the operating mode and the system alarm of the system through the PIADA.
- Bit 7 is used for local/remote mode indication.
- Bit 6 is used for master/TTY mode indication.
- Bit 5 and bit 4 are not used.
- Bit 3 is used to monitor -12V TTY power supply.
- Bit 2 is used to monitor 12V TTY power supply.
- Bit 1 is used to monitor analog input voltage conversion chassis #2 power supply.

- Bit 0 is used to monitor analog input voltage conversion chassis #1 power supply.

<u>PIACB</u>	1 bit	Flag bit

- The update task sets bit 3 to indicate a one second increment of the system clock.

<u>Data Acquisition Task/Keyboard and Display Interface</u>

<u>MXCHLS</u>	1 bit	Flag bit

- The data acquisition task uses the LSB of the B side of PIA2 to pass a conversion status to the keyboard and display interface card.

<u>Interpreter Task/Keyboard and Display Interface</u>

<u>PIACA</u>	1 bit	Flag bit

- The interpreter task sets bit 3 to indicate the existence of alarms within the system.

<u>TTY Input Task, TTY Output Task, Protocol Task/Master/TTY Interface</u>

<u>ACIAD</u>	1 byte	Packed format

- The ACIA is used for communication between the operator's terminal and the TTY input task as well as the TTY output task. The TTY input task reads from this location, and the TTY output task writes into this location.
- The ACIA is also used for communication between the master processor and the

protocol task when the system is in master mode. They both read from and write into this location.

To further clarify the operation of the two PIA's, their assignments are shown in Figure 5-6.

The allocation of system memory among hardware modules is described by the memory map shown in Figure 4-27.

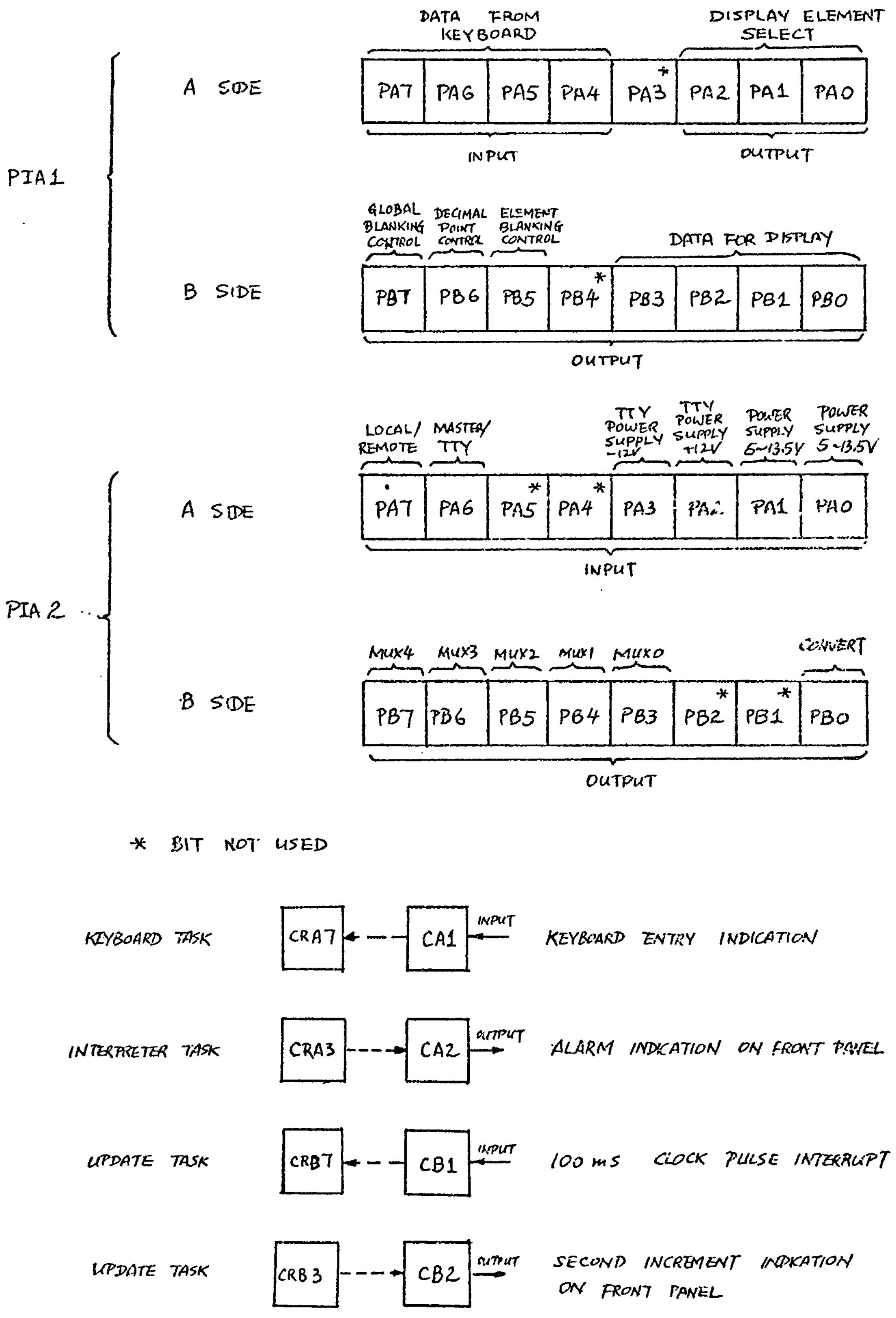


Figure 5-6 Assignments of the two PIA's

CHAPTER 6

SYSTEM EVALUATION, RECOMMENDATIONS AND CONCLUSION

In this chapter an evaluation of the system performance is made, design improvement recommendations are suggested, and conclusion is given.

6.1 System Evaluation

Tests of the TCU have demonstrated that it satisfies all the system requirements specified in chapter 2, and that it materializes the functional design described in chapter 3. The system was installed at Agnes Arnold Hall and is successfully operated by power plant personnel with very little digital system experience. The system has been operating continuously for more than three months performing energy monitoring and control functions. Its operation has been completely satisfactory.

6.2 Recommendations for System Improvement

Use of the system has revealed that improvement is desirable in the four areas listed below:

- Implementation of all commands at the front panel keyboard and provide the display with complete reporting capability.
- Provode means for task modification to allow the insertion of new control programs.
- Power failure recovery.
- Integration of the MCU and TCU as one unit.

6.2.1 Expansion of Keyboard and Display Capability

The system front panel only accepts commands associated with monitoring functions. No control capability has been implemented at the keyboard. By implementing the keyboard entry of all commands and providing a complete LED display of all reports the system will present the following advantages: (1) simplification of system testing and maintenance after installation since a TTY terminal is usually located at a remote site, and (2) lower cost of operation by allowing operation without TTY terminal. A disadvantage of the proposed implementation is the loss of capability to provide simultaneous and independent reports on the TTY and the LED display.

6.2.2 Automatic Control Program Insertion

An automatic set point control task is desirable for achieving complete digital control of the system. A task that performs this function may be added in memory in any of the available locations shown in Figure 4-27. This task should interact with the global common variables TIME, DAY, and CHNVAL to get the time of the day, date of the year and air temperatures as the parameters for set point temperature control. It should use the inter-communication table SPINCM to request the set point task to perform set point adjustments. Addition of the new task will require modification of the interpreter and the executive. Three new commands should be implemented: one for starting automatic set point control, the other for stopping it and the last for reporting its operation.

6.2.3 Power Failure Recovery

A number of power interruptions have occurred at the University of Houston since the TCU system was installed. As a result, the time of day, the date, the accumulated values of ton-hour of chilled water and the pound of steam condensate were lost. The paper logging function was terminated until reestablished by operator command. It is recommended that a battery operated clock and a small size core memory be included in the system. This modification will allow the system to recover from a power failure with exactly the same operation it had prior to the event.

6.2.4 Integration of The MCU and TCU as a Unit

Since one microprocessor is capable of handling both the Motor Control Unit(MCU) and TCU functions and the total program length is less than 65K, the two systems could be integrated as one unit. The advantages of this approach are: (1) only one processor unit, (2) reduction of communication lines to the remote Central Control Unit (CCU), and (3) simpler interaction between MCU and TCU programs.

6.3 Conclusion

The use of an organized design approach to the implementation of the Thermal Control Unit proved to be very effective. The disign objectives and operational objectives were successfully implemented. A complete description of all major hardware and software modules was presented. The interactions between modules were described. An evaluation of the system was made and recommendations for improvement of its performance have been suggested.

APPENDIX A

OPERATIONAL THEORY OF AIR HANDLING UNIT

Two types of air handling units are described in this appendix: the single duct unit, and the double duct unit.

Figure A-1 shows the schematic diagram of a single duct air handling unit which is used to supply air to a single zone. Return air from the zone and outside air are mixed in a fixed proportion and blown into the distribution duct. Hot water and chilled water are circulated through heat exchanger coils to heat up or cool down the air. The flow of hot water and chilled water is modulated by two valves which are controlled by the room thermostat.

Figure A-2 shows the schematic diagram of a double duct air handling unit which is used to supply hot air and cold air to a number of rooms. The return air from the rooms and the outside air are mixed in a fixed proportion and blown into the two ducts of the unit. Hot water and chilled water are circulated through heat exchanger coils of the hot deck and the cold deck to heat up and cool down the air in the two ducts. The flow of hot water and chilled water are restricted by the opennings of two valves which are controlled to maintain the air temperature equal to the set point value.

The temperature of each room is regulated by a local thermostat which adjusts the position of a damper to control the mixture of hot air and cold air discharged into the room.

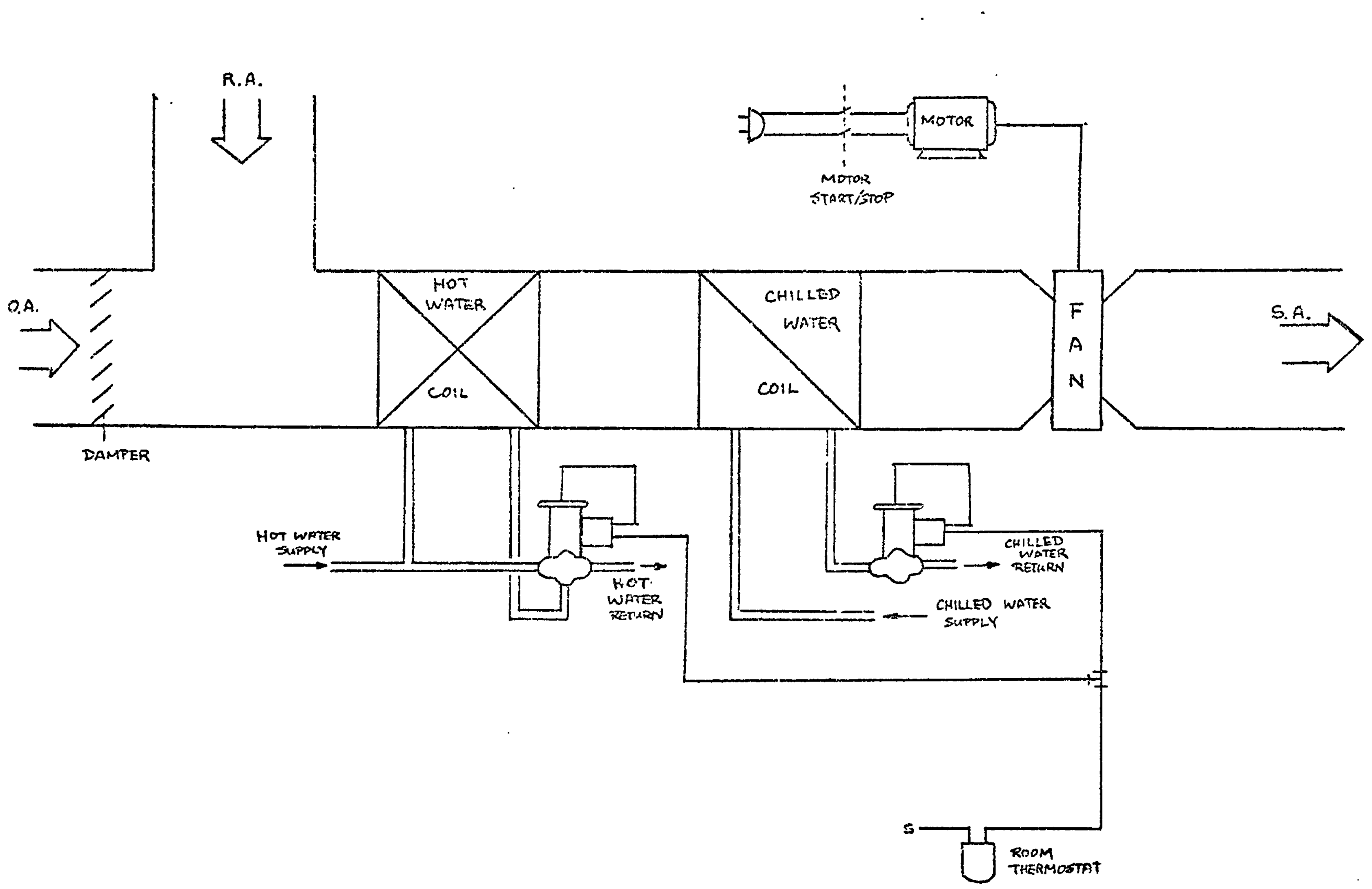


Figure A-1 Single duct air handling unit

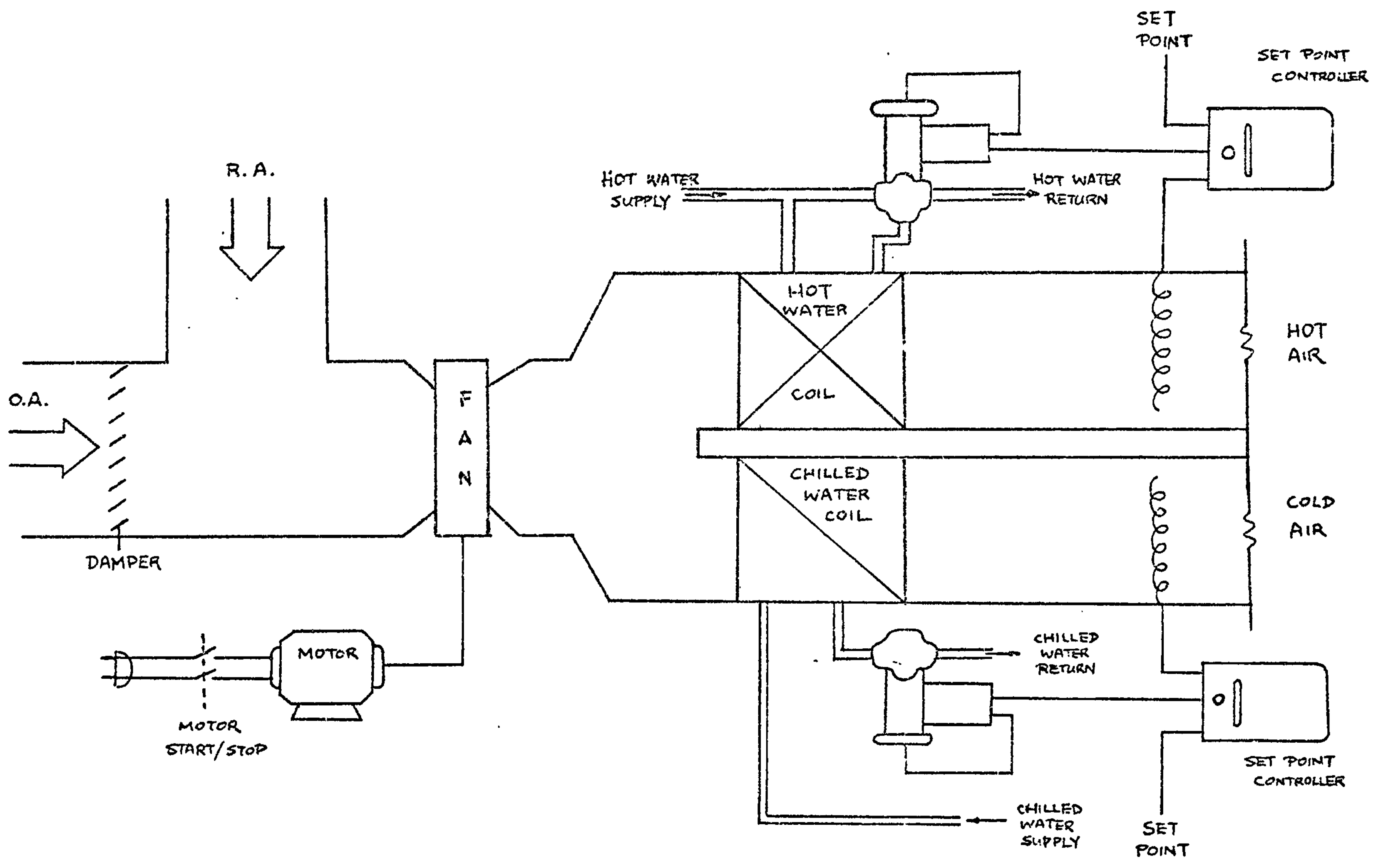


Figure A-2 Double duct air handling unit

APPENDIX B

SURVEY OF ENERGY SAVING CONTROL METHODS OF EXISTING HVAC EQUIPMENT

Existing HVAC equipment can be modified for control of its energy consumption with only slight modification. The required controls consist of: (1) on/off control of motor and position control of inlet air damper, and (2) set point temperature control. By combining these two types of control it is possible to achieve great savings on both electrical and thermal energy.

The motors that should be controlled are the air handling unit fan motors and the hot and chilled water pumps. An air handling unit should operate only when it is regulating an occupied zone. The hot and chilled water pumps that supply hot and chilled water to air handling units should operate only when the air handling units are on. These motors should be scheduled to start and stop in accordance with the time of the day and the day of the week. When the air handling units are started, no outside air but only return air should be circulated inside the building until the building has reached its desired temperature. In this manner no energy is wasted to heat up or cool down the outside air which is only necessary when the building is occupied. The hot and chilled water pumps should be turned on at different times according to the building temperature. The building only needs either heating or cooling but not both initially. During the day when the building is occupied a minimum of 20% of outside air is required. The openning of the inlet air damper should be adjusted according to enthalpy calculations based on the humidity of the air, outside air temperature, return air temperature, and air handling unit set point temperature, so that the least amount of energy will be used to condition the air mixture that passes through the air handling unit. At night the decrease in occupancy of the building allows for a reduction in the outside air required to maintain proper air quality. Therefore it is possible to adjust the outside air dampers to a minimum position. The hot and chilled water pumps should be turned off prior to the shut off of the air handling units since the thermal mass of the building can maintain the temperature of the air for a reasonable period of time. During cool summer nights, the outside air can be used as a free source of energy to regulate the building's temperature.

The temperatures which require control are the set point temperatures of the hot and cold decks of a double duct air handling unit and the set point temperature of the hot water converter and the chilled water. The set point temperatures of the decks of a double duct unit are usually set wide apart to provide a wide range of room temperature selection. The mixing of hot and cold air represents a great waste of energy, therefore the difference between the hot and cold deck set point temperatures should be minimized. This can be done by using a load analyzer which resets the hot deck set point temperature according to the demand of the room that requires most heating, and resets the cold deck set point temperature according to the demand of the room that requires most cooling. Instead of using a load analyzer which requires a sensor connected to each room serviced by the air handling unit, it is possible to instrument "worst case" rooms with temperature sensors and let the demand of these rooms reset the hot deck and cold deck set point temperatures. A dead zone should be used in the control of temperature so that energy is not consumed both in heating and cooling by the same air handling unit. The set point temperatures of the hot water converter and chilled water should be reset for minimal energy consumption in accordance with the set point temperatures of the air handling units. Set point control improves the efficiency of the HVAC system, specially in equipment for which sequencing of control is not possible.

BIBLIOGRAPHY

(1) Simmons, H.: The Economics of America's Energy Future. U.S. Energy Research and Development Administration; Washington, D.C.; 1975.

(2) Snively, Hugh V.: Energy Saving Applications for Computerized Automation. ASHRAE Transaction. Vol. 80, Part 1, pp. 35-41; February, 1974.

(3) Spethmann, Donald H.: The Importance of Control in Energy Conservation. ASHRAE Journal. Vol. 17, pp. 35-41; February, 1975.

(4) Shavit, Gideon: Energy Savings Through Computer Control of Fan Systems with Floating Space Temperature. ASHRAE Transaction. Vol. 83, Part 2, pp. 374-383; 1974.

(5) Snively, Hugh V.: Energy Saving Applications for Computerized Automation. ASHRAE Transaction. Vol. 80, Part 1, pp. 35-41; February, 1974.

(6) Spethmann, Donald H.: The Importance of Control in Energy Conservation. ASHRAE Journal. Vol. 17, pp. 35-41; February, 1975.

(7) Cole, H.: System/7 in a Hierarchical Laboratory Automation System. IBM Systems Journal. Vol. 13, pp. 307-324; 1974.

(8) Weissberger, Alan J.: Microprocessors Expand Industry Applications of Data Acquisition. Electronics. pp. 107-110; September, 1974.

(9) Reed, Mark and H.W. Mergler: A Microprocessor-based Control System. IEEE Transactions on Industrial Electronics and Control Instrumentation. Vol. 24, No. 3, pp. 253-257; August, 1977.

(10) Shih, James Y.: Automation System with Addressable Sensing Devices. ASHRAE Transaction. Vol. 83, Part 1, pp. 393-397; 1977.

(11) Pang, Kam K.: Reference Manual for The Analog Output Card. Digital Control and Automation Systems Lab., University of Houston. Report No. 77-6; November, 1977.

(12) Pang, Kam K.: Reference Manual for The Set Point Task. Digital Control and Automation Systems Lab., University of Houston. Report No. 78-2; August, 1978.

(13) Wang, C.: Reference Manual for The Thermal Control Unit (TCU) Interpreter. Digital Control and Automation Systems Lab., University of Houston. Report No. 78-4; August, 1978.

(14) Wang, C.: Reference Manual for The Thermal Control Unit (TCU) Executive. Digital Control and Automation Systems Lab., University of Houston. Report No. 78-3; August, 1978.

(15) Pereira, M.: Reference Manual for The Thermal Control Unit (TCU) Data Acquisition Software. Digital Control and Automation Systems Lab, University of Houston. Report No. 78-1; February, 1978.