NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us a poor photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30. Please read the authorization forms which accompany this thesis.

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de mauvaise qualité.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30. Veuillez prendre connaissance des formules d'autorisation qui accompagnent cette thèse.

THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED

LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE

DIRECT DIGITAL CONTROL OF A DISTILLATION COLUMN

by

Marc Castonguay

A thesis submitted to the School of Graduate Studies
in partial fulfilment of the requirement for
the degree of Master of Applied Science

in the

Department of Chemical Engineering
University of Ottawa
October 1979

ABSTRACT

A ten tray binary distillation column of pilot plant scale was interfaced to a Nova digital minicomputer for direct digital control.

The aim of the present study was three-fold; to develop a flexible software package to allow on-line computer control, to develop a system needing a minimum of hardware interfacing, and finally to use these systems to control a distillation column.

System software interfaces were developed for all control equipment and for the Hewlett-Packard Interface Bus (HP-IB). RTOS, a Datageneral Corp. operating system, was modified and debugged to complete the software package.

A complete hardware system, consisting of a Datageneral Nova minicomputer, a Hewlett-Packard digital voltmeter model 3455A and scanner model 3495A, and a Datageneral digital to analog converter, was assembled to complete the hardware control system. Various pneumatic/electric and electric/pneumatic converters were used to interface the electronic equipment with the pneumatic instruments.

ACKNOWLEDGEMENTS

The author wishes to express his sincere thanks to his supervisor, Dr. F.D.F. Talbot for his encouragement and technical guidance.

He is grateful to Mr. G. Gasperetti and the rest of the technical staff of the Chemical Engineering Department for their help in constructing and maintaining the equipment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

DDC:	Direct digital control
M(t):	Output signal
Mr:	Base output signal
Kc:	Proportional gain
$\varepsilon(t)$:	error
τ_I:	Integral constant
DVM:	Digital voltmeter
HP-IB:	Hewlett Packard Interface Bus
CR:	Carriage return (TELETYPE)
LF:	Line feed (TELETYPE)
D/A:	Digital to analog
P-I:	Proportional integral
LC:	Level controller
FC:	Flow controller
t:	Discrete time between control outputs

INTRODUCTION

Distillation is probably the most widely used separation technique in the chemical and petroleum industries. Batch distillation of essentially binary mixtures of ethanol and water, has been practised for centuries. However, only in the present century has distillation been understood and developed on a large scale.

In the early stages of process control, a distillation column with its complex mass and energy transfers, was controlled for steady-state operation by using an individual analog controller on each process variable[1]. However, with increasingly complex processes, emphasis shifted from individual operations to whole process units. The use of optimization techniques and overall control schemes permitted smoother and more economical plant operations[2].

The arrival of the digital computer, with its speed and flexibility, was a milestone in the development of process control technology.

Presently the digital computer may be implemented in a process computer control system in one of the following four levels:

1. Off-line: The digital computer in an off-line system is used for data-logging and process optimization. Data-logging usually results in a gathering of redundant data taken near normal operating conditions. Therefore, in general,

data-logging in itself is not regarded as a justifiable reason for the purchase of a digital computer.

2. Direct-Digital Control (DDC). Direct-digital control is essentially a direct replacement of an analog controller with a digital computer. Although one medium size computer may replace hundreds of analog controllers, there usually exists no break-even point at which the cost of the analog controllers equals the cost of the digital computer. There are three reasons for this: one being that programming costs are usually higher than expected and difficult to anticipate; another being that the use of the digital computer almost doubles the hardware requirements since analog back-ups are needed in the event of computer failure; and finally, the analog controllers are very reliable and replacement inexpensive in contrast with a direct-digital control computer.

3. Supervisory Control: There must exist an overall plant optimization strategy if financial returns are to be maximized. This strategy, the result of consideration of the combined effect of many different options, is best determined by a computer. Implementation, again is best suited to a computer as time is of the essence, however, due to the shortcomings of the DDC system, a compromise had to be made. The basis of the supervisory control system is that although the computer determines the optimum operating strategies, the analog control system implements them. Interfacing is achieved by computer manipulation of the analog controller's setpoint.

4. Hierarchy Concept: The proceeding sections presented distinct approaches to the applications of computer control to process units. The hierarchy concept combines these approaches by incorporating the best of both. Most applications are supervisory in nature but the more attractive DDC loops are retained. The hierarchy concept is achieved by having a grouping of DDC computers, each responsible for the control of a plant, supervised by a supervisory computer. This computer is grouped with other supervisory computers and as a group is coordinated by a scheduling computer. DDC becomes more attractive in this case as few additional analog inputs are needed. The analog back-up problem is solved by the use of the supervisory computer for control of critical loops in case of DDC computer failure.

This hierarchy concept also permits corporate decisions to be executed in a much shorter time thus increasing efficiency.

The aim of the present study was three-fold; to develop a flexible software package to allow on-line computer control, to develop a system needing a minimum of hardware interfacing and finally to use these systems to control a distillation column. The final part, in reality a verification of the first two, was implemented using a simple control scheme to control a pilot plant scale, ten-tray distillation column.

Many terms used throughout this study, applying to

computers and computer applications are explained in a glossary in Appendix I. This glossary, by no means complete, is written in relation to the Nova minicomputer system purchased from Datageneral Ltd.

Control Schemes for Distillation Columns

I Distillation Control

a) Multiloop Control Scheme

The dynamics and control schemes of simple systems are well known[3], however in practice the systems to be controlled are usually quite complex. In addition quantitative dynamics in terms of transfer functions for these complex systems are rarely available.

The control of a distillation column is a good example of such a system; there are many control loops with a great number of variables. However, it is essential that proper control of a distillation column be achieved if product specifications are to be met. Figure 1 shows a diagram representing a typical distillation column. The control of such a column can be defined as the column operation at or near specified conditions.

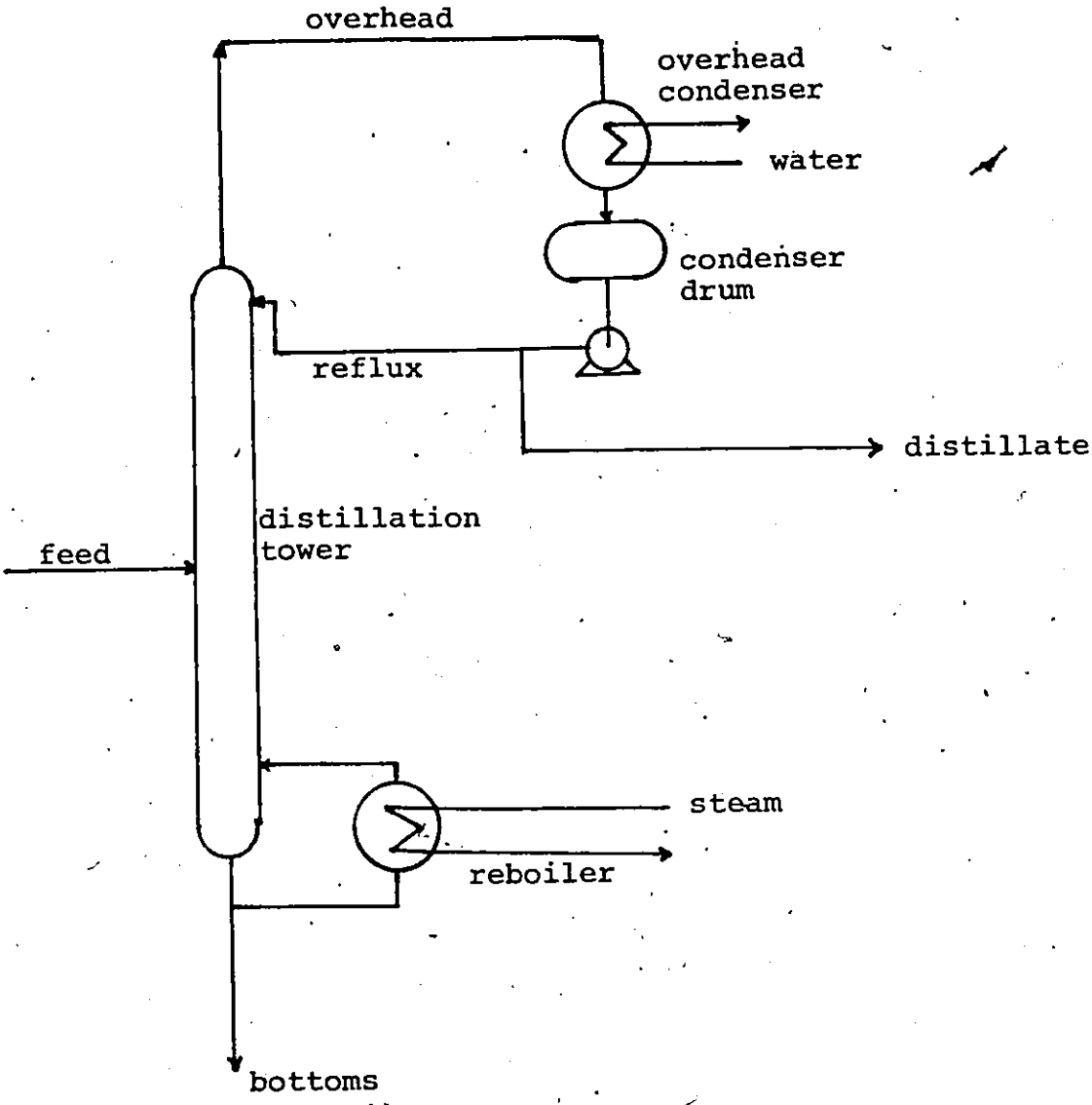
In the case of binary distillation with

1) a specified feedrate and composition with constant feed temperature,
2) a specified steam rate to the reboiler with a constant saturation temperature,
3) a specified cooling water rate with constant water temperature and,
4) a specified reflux ratio.

FIGURE 1

TYPICAL DISTILLATION TOWER

Typical Distillation Column



the steady state conditions are set. Specifying these variables fixes the distillate composition and rate, bottoms composition and rate, and column pressure. The independent variables could be seen as input variables and the second group as dependent variables. Any change of an independent variable will redefine the system and will result in a change in all of the output variables.

b) Distillation Control Philosophies

The most common control system applied to distillation columns is referred to as the energy or heat balance control scheme and is represented in Figure 2. Another newer control system, the material balance control scheme, appears in Figure 3. In a material balance control scheme, a product flow is always manipulated to control composition, while for the heat balance control scheme heat input is the manipulated variable.

With the heat balance control scheme a constant reflux is maintained on the distillation tower, both products are under level control; their flows therefore are subject to influence from disturbances that affect the column. Product composition is controlled by adjusting boil-up relative to a fixed reflux. Increasing boil-up or vapour flow results in two changes; separation is increased and the material balance is shifted. Separation is increased since more low boiling material is removed from the bottoms, this tends to make the bottoms purer. The shift in material balance

FIGURE 2

HEAT BALANCE DISTILLATION CONTROL SCHEME

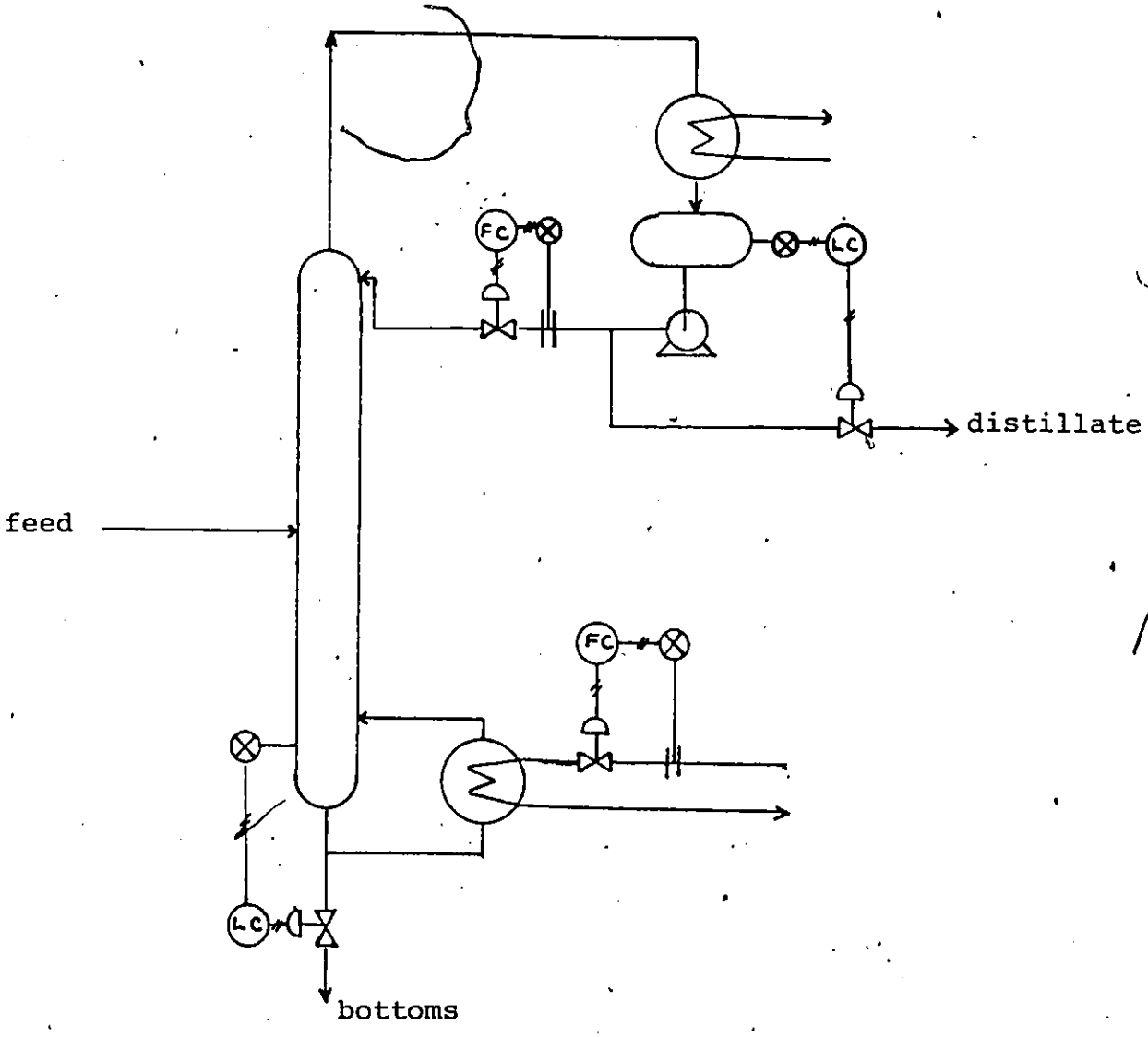
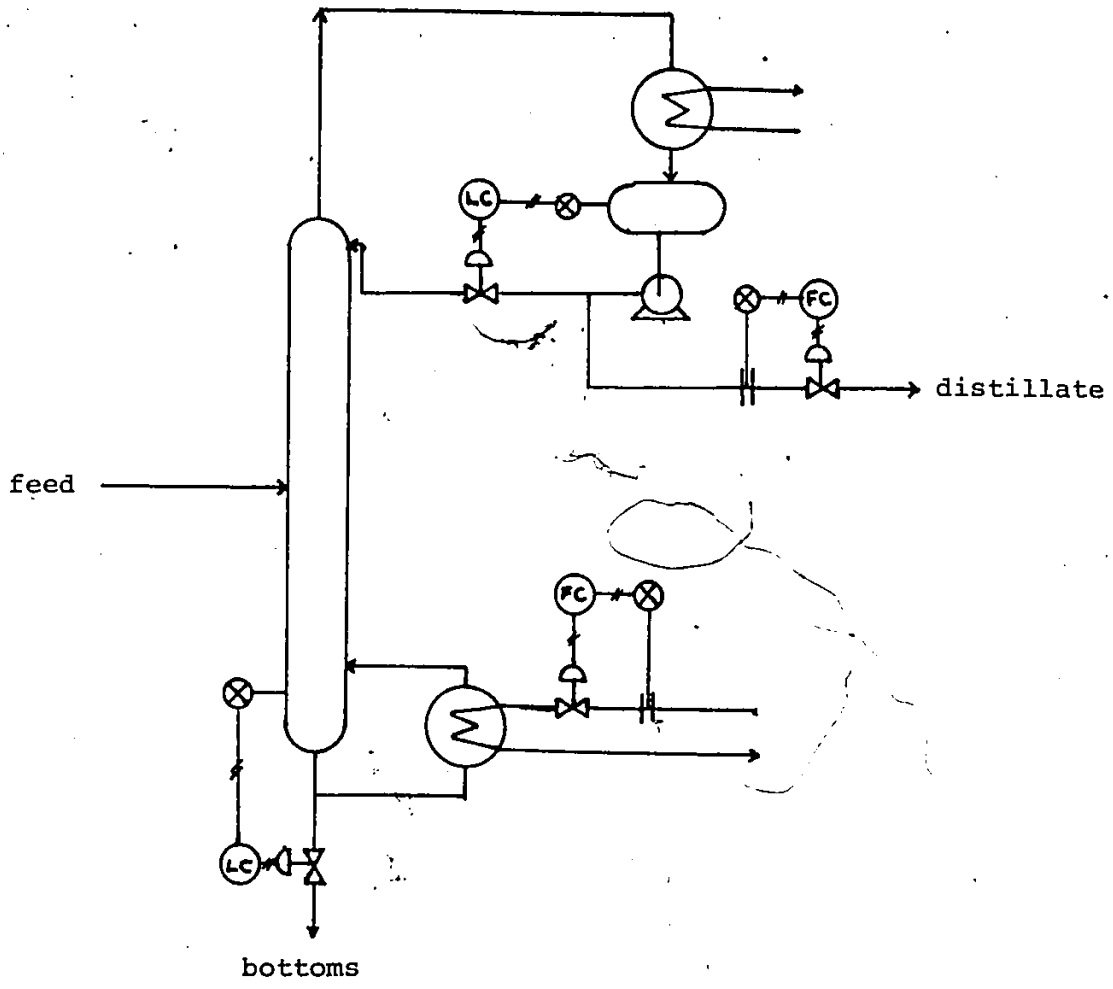


FIGURE 3

MATERIAL BALANCE DISTILLATION CONTROL SCHEME



results from increasing the amount of material transferred from the bottom of the column to the top. The direct effect is to improve the purity of the bottoms product by decreasing the flow and to reduce the purity of distillate by increasing its flow.

The material balance control scheme maintains a constant overhead product stream through a flow controller. The reflux rate is manipulated by level control of the accumulator drum. This results in making the tower reflux a dependent variable. This is a very controversial issue since distillation design procedures center around reflux and reflux ratio. The material balance control scheme came into being when the conventional heat balance control system was ineffective for control over close separations. Other advantages of the material balance control scheme lie in its insensitivity to variations in enthalpy. An increase in heat input to the tower will result in raising the product level in the accumulator drum and increasing the reflux. The net effect on composition will be much smaller than if the reflux were flow controlled and the increased boil-up were passed to the distillate product as with the energy balance scheme.

In particular to computer control, the material balance scheme is most advantageous since it is readily convertible to incorporate a feed forward control scheme. Although possible to incorporate into a heat balance control scheme, a feed forward control scheme would be much more difficult to operate efficiently.

c) Control Scheme Used

The energy or heat balance control scheme was used on the binary distillation in this work. There were several hardware limitations to the conversions to a material balance control scheme.

Two control loops, the reflux flow control and the steam flow control, were uncoupled and connected to a minicomputer. The remaining loops continued with their analog controllers. The minicomputer was used simply as the basic proportional/integral controller it replaced. This uncoupling was done to only two loops because of hardware limitations, notably the lack of additional pneumatic/electric interface transducers and additional digital/analog converter channels.

The basic concept for this uncoupling was not to develop some exotic control scheme but rather to verify the computer hardware and software systems developed. Once both systems are established any type of control scheme could be used with the minicomputer.

The control system developed for the minicomputer controlled two loops but monitored three flow measuring elements. The third monitored flowrate was that of the feed; this was to enable future work to include a feed forward control system.

II Design of Distillation Column

The distillation column was assembled from 9 inch glass pipe having an inside diameter of 8.34 inches and an outside diameter ~~of 9~~ 25 inches[4]. The column was composed of ten bubble cap trays with each tray having two bubble caps. Distributor plates were used for the feed and reflux liquid streams. The trays and distributor plates, made from brass, had a thickness of 3/8 of an inch. The bubble cap riser consisted of a 1-1/2 inch type "K" seamless copper water pipe while the bubble caps were made from 2 inch type "K" seamless copper water pipe caps. Each bubble cap contained 28 1/8 inch x 1/2 inch slots located 1/16 of a inch from the bottom of the cap. Two inch type "K" seamless copper water pipes were used as circular downcomers; extending 1-1/2 inches through the tray they provided the necessary weir height. Tray spacing including distributor plates was one foot. On each tray, located between the downcomer and bubble caps, a copper-constantan thermocouple and a liquid sample outlet were used for temperature and composition measurements. Liquid samples could be withdrawn from each tray by the use of a remotely controlled solenoid valve. A water cooler was used to cool liquid samples.

The feed system consisted of two stainless steel drums, a preheater and a distibution plate. The two feed drums were 24 inches in diameter and 68 inches high. The feed distribution plate was located above the fourth tray

from the bottom of the column. Using steam as the heating medium, the feed preheater consisted of a shell and tube heat exchanger. The shell side of the heat exchanger was made of 3 inch carbon steel pipe while the tube side consisted of ten three foot lengths of 3/8 inch type "K" seamless copper tubing. The heating steam was manually controlled with a pressure reducer.

The overhead system consisted of a total condenser, an overhead reflux drum and two product tanks. Two shell and tube heat exchangers operating in series made up the condensation system. Having a total length of four feet, the two heat exchangers were similar to the feed preheater heat exchanger; the only notable difference being that the shell side was constructed of 3 inch type "K" seamless copper tubing instead of 3 inch carbon steel. Cooling water was controlled by a hand valve and rotameter. The flowrate of the distillate product, withdrawn from the overhead, was controlled by liquid level control on the product drum. The two stainless steel product tanks measured 18 inches in diameter and 28 inches long. The reflux, pumped onto the reflux distributor plate, was under flow control.

The bottom's system consisted of a thermosyphon reboiler, a bottom product cooler and two storage tanks. The reboiler, a 8 inch diameter copper shell and tube heat exchanger, was mounted vertically on the bottom of the column. The shell's thickness was 0.125 inches while the copper tubes had a 5/8 inch outside diameter and a 0.045 inch

wall thickness. The heat exchanger tubes, numbering forty-eight and eighteen inches in length surrounded a central 2" pipe. Steam condensed on the shell side and liquid circulated on the tube side. The bottom product withdrawal rate was governed through a reboiler liquid level control system. A bottom's cooler, of the same design as the overhead condenser, served to cool the bottom's product before storage. The two horizontal stainless steel storage tanks measured 24 inches in diameter and 48 inches in length.

The piping system was designed to permit direct transfer between feed, bottom and product tanks. This allowed for product recycling and blending.

III Control Loop Algorithm

For Proportional-Integral Control (P-I)[2], the mode of control is described by the relationship

$$M(t) = Kc\varepsilon(t) + \frac{Kc}{\tau_I}\int_o^t \varepsilon\, dt + Mr \qquad (1)$$

There are two control action terms; the first term is proportional to the error and the second is proportional to the integral of the error. Using numerical integration, the output signal (M) can be calculated for the discrete time "t"; in this form the control scheme can be implemented with the use of a digital computer.[4] Using the trapezoidal rule for numerical integration, equation (1) becomes, at time N,

$$M_N = Kc\left[\varepsilon_N + \frac{t}{\tau_I}\sum_{k=o}^{N}\frac{\varepsilon_k + \varepsilon_{k-1}}{2}\right] + Mr \qquad (2)$$

This equation, known as the position form of the algorithm, enables the actual valve position (M_N) to be calculated from the error sequence. An alternative to this is known as the velocity form of the algorithm which calculates the change in valve position rather than its actual position. At time (N-1)

$$M_{N-1} = K_c \left[\varepsilon_{N-1} + \frac{t}{\tau_I} \sum_{K=o}^{N-1} \frac{\varepsilon_k + \varepsilon_{k-1}}{2}\right] + Mr \qquad (3)$$

Finding ΔM results in the velocity form of the algorithm:

$$\begin{aligned} \Delta M_N &= M_N - M_{N-1} \\ &= K_c \left[(\varepsilon_N - \varepsilon_{N-1}) + \frac{t}{2\tau_I} (\varepsilon_N + \varepsilon_{N-1})\right] \end{aligned} \qquad (4)$$

This form of the algorithm has the advantage that it needn't be initialized.

In this study a modified version of the velocity form of the proportional-integral control algorithm was used. The modifications being that upon termination of manual control, the (N-1) error was assumed zero and that the current process inputs were used as control setpoints. This last modification was incorporated into the control scheme because of the large range differences between the measuring element and the control valve. That is to say a control valve input of 8 PSIG may cause the valve to remain closed while an output of 8 PSIG from a d/p cell may indicate a substantial flow. This was done to assure a "bumpless"

transfer from manual to automatic control. It is analogous to the procedure for "bumpless" transfer on analog controllers; from a manual or output related setpoint to an automatic control or input related setpoint.

The computer controlled variables in this study were limited to the reflux and steam flowrates. Feed flowrate was monitored to permit a feed forward control scheme to be implemented in the future. The distillate and bottom flowrate were under analog level control while the feed was on flow control. In all cases, the control scheme involved the proportional-integral control algorithm.

Figure 4 shows the experimental column before control loop uncoupling while Figure 5 represents it after.

FIGURE 4

SCHEMATIC DIAGRAM OF EXPERIMENTAL COLUMN

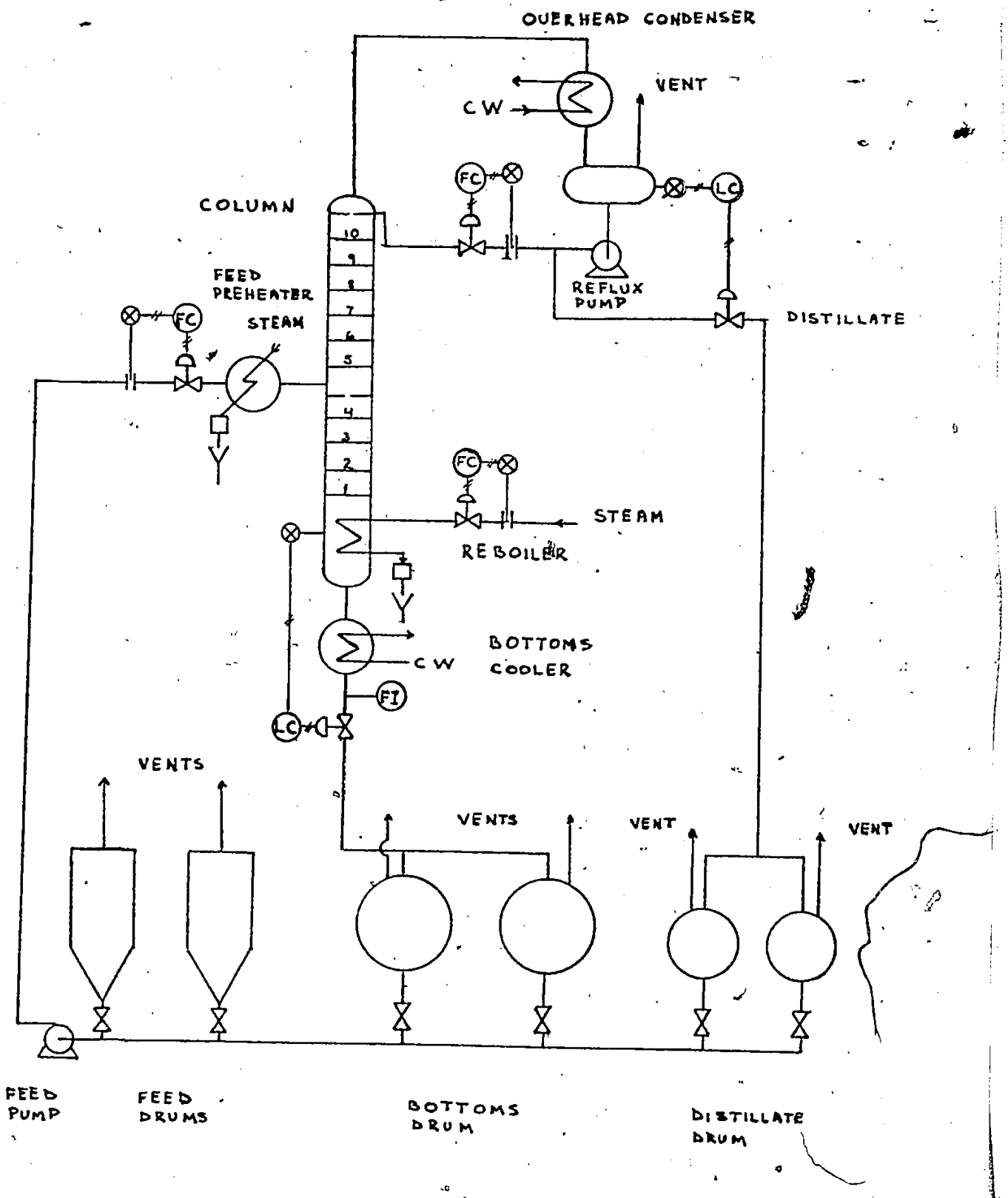
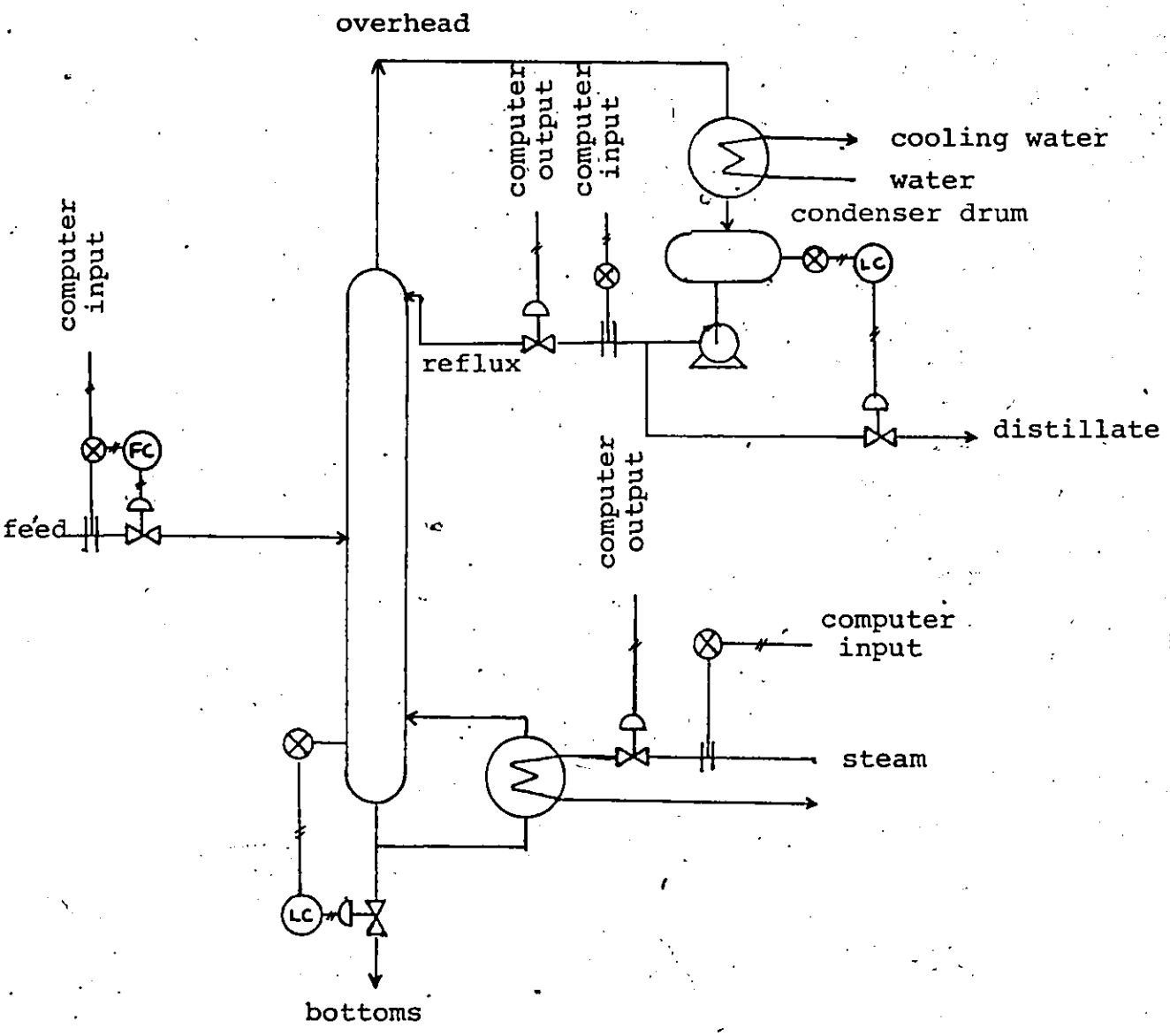


FIGURE 5

CONTROL LOOP UNCOUPLING OF COLUMN



Computer Operating System

Software, that is to say computer programs, can be divided into three major categories[5] (Table 1). Software which performs a specific task such as solving an equation or maintaining a filing system is called applications software. On the other hand, software can also aid programmers in writing applications software, this is often called developmental software. Examples of this are compilers, assemblers and debuggers. Lastly there is software which is used to control the operation of the computer system. This is a master program which controls all activities of the computer system. It is referred to as Operating System, Executive or Supervisor.

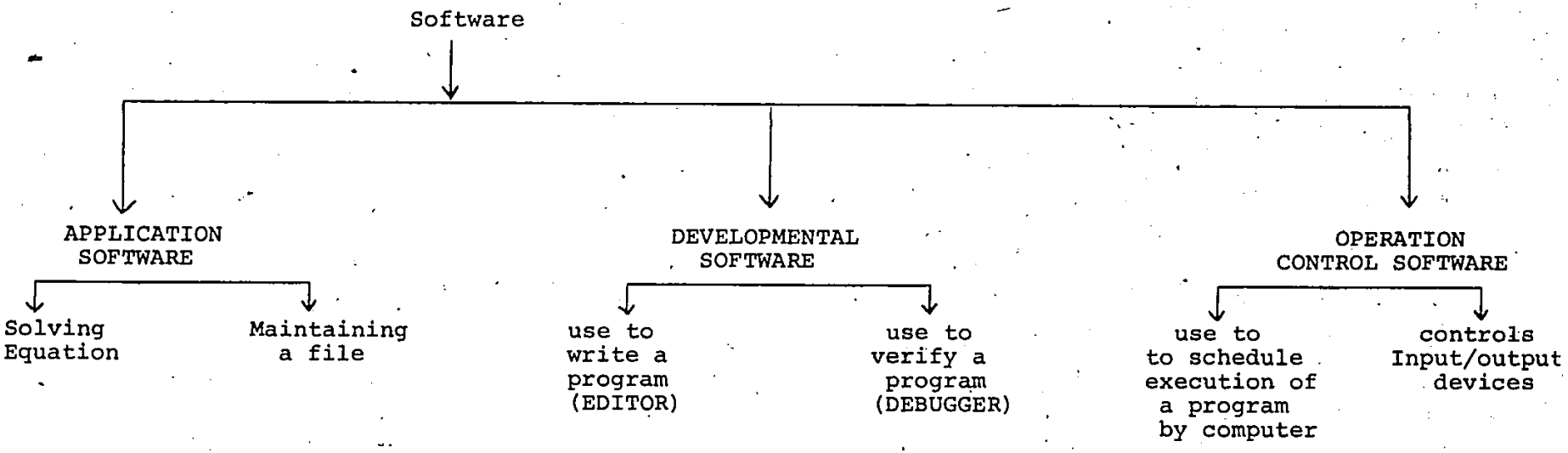
The role of the Operating System is simply to coordinate major activities of the computer system, with a minimum of user interaction, as efficiently as possible. Most minicomputer operating systems perform task scheduling, control of I/O operations as well as handle communications between computer system and operator. Because of the limited resources available, minicomputer operating systems tend to be much less powerful than larger computer systems, however their basic objectives are the same.

A specific type of operating system relating to a real external time environment, is called a real-time operating system[6]. Real time in this context, refers to the operating system's ability to execute tasks according to a predetermined

Breakdown of Software Categories

and examples of each

Table I - Software Categories



time schedule. This time scheduling is required for the operation of external equipment in the computer system. Scheduling is accomplished using a clock referred to as a real time clock. Program sequences are executed in response to interrupt requests from the real-time clock, sensors or control logic as well as in response to keyboard commands. Program sequences or tasks are assigned priorities permitting efficient coordination for their execution.

RTOS

The Datagenral real-time operating system for the Nova computer family[7] (RTOS) is a multi-purpose monitor capable of controlling a wide variety of real-time input/output devices. The system, entirely core resident, relieves the programmer of the job of task scheduling and priority handling necessary to handle a number of tasks in a real-time environment. A task is defined as any logically complete program segment. Only one task can have control of the central processing unit (CPU) at a time. RTOS provides supervisory control and determines which tasks are pending execution and in what order they are to be executed; this is called Task state transition. This is one of the four basic operations within RTOS, the others are task synchronization/communication, real-time clock servicing and input/output servicing.

A task may be in any of four states. An executing task has control of the CPU and will retain control until the

task scheduler reschedules a new executing task. A pending task is waiting for the use of the CPU and will be called upon to become the executing task by the task scheduler. A suspended task must wait for the occurrence or completion of some real-time operation to again become the executing task. A dormant task is a task which is neither suspended, pending or executing. This situation occurs if the task has not been created in the system or if the task has been completed. This logic appears in Figure 6.

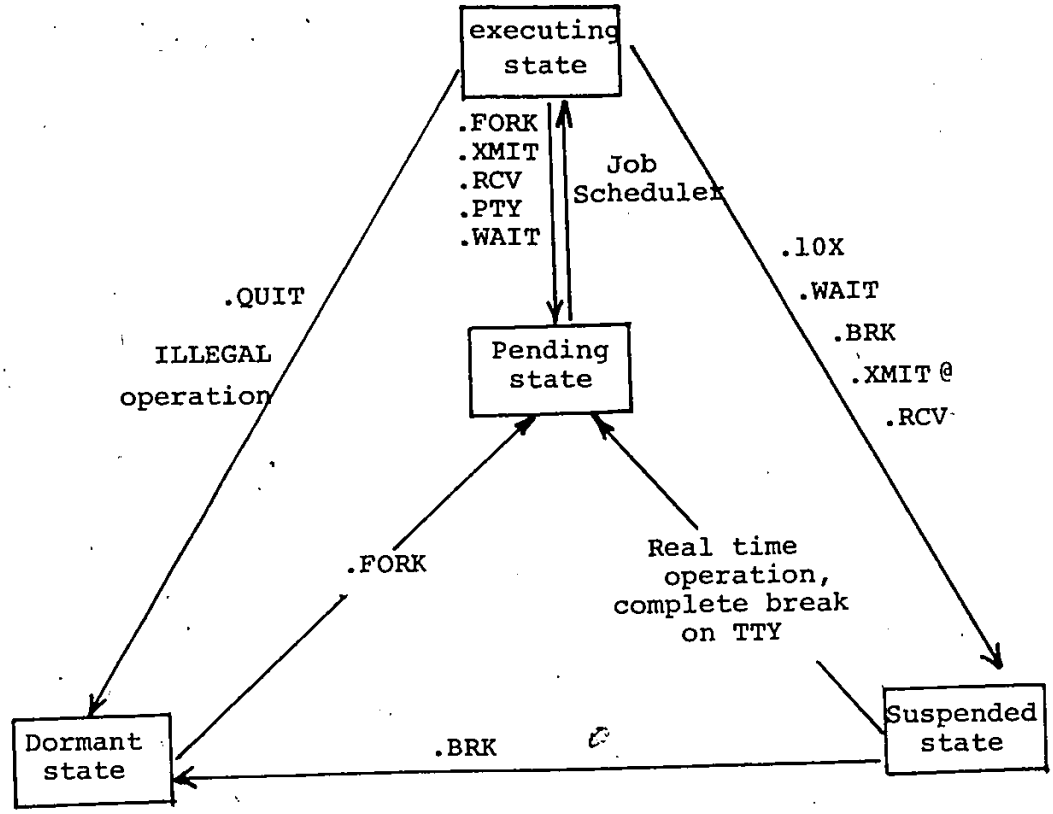
User programs communicate with RTOS through the use of a small set of meta-instructions consisting of machine language subroutine calls. In this study only six of the eight available meta-instructions were used; .10X, .FORK, .PTY, .WAIT, .BRK, .QUIT. Each of these meta-instructions will be described briefly.

The .10X meta-command is the standard I/O call in RTOS. This instruction causes initiation of the desired I/O then forces the current task to be placed in a suspended state until the desired I/O is complete, thereby allowing RTOS to execute an alternate task in the meantime. Since most I/O devices are slow compared to the computer this permits more efficient use of the computer's processing capabilities.

The .FORK meta-command is the basic mechanism in RTOS for the creation of parallel processes. The .FORK instruction execution creates a new "TASK" at a specific priority level. The tasks are totally independent of each

FIGURE 6

RTOS TASK STATUS



other and are placed in a pending state, the system scheduler then determines which task (the highest priority level) will be executed. In RTOS a high priority level is indicated by a low priority number.

The .PTY meta-command is used to initialize or dynamically alter the priority of the executing task. Therefore with the .FORK instruction, the programmer is capable of assigning high priority levels to tasks which are always important, or, low priority levels to tasks which are secondary. While with the .PTY instruction the user is capable of changing the priority level of a certain task after, for example some logic test has been completed within the task.

The .WAIT meta-command is used to delay the execution of the current task for a certain time interval. When the current executing task encounters a .WAIT instruction, it is placed in the suspended state allowing the task scheduler to start execution of another task of high priority level until the time interval is complete. This meta-instruction allows the programmer to use the operations system's method of measuring time (real-time clock).

The .BRK meta-command is not a general purpose meta-instruction. This special purpose command allows the operator to communicate with the executing program through the operator's teletype. The .BRK meta-instruction causes the executing task to stop and wait in the "background" until a specified ASCII character is entered through the teletype.

When this character is entered, execution of the "background" task is resumed after the .BRK statement. This can be used as a convenient way for the operator to initiate a task within the program.

The .QUIT meta-command is used to terminate the execution of the currently executing task. It is the program's signal to RTOS that a task is complete, it is implemented simply as an entry to the task scheduler.

The .XMIT and .RCV meta-commands are a complimentary set of meta-instructions permitting inter-task communication and task synchronization. These meta-commands were not used in this study.

RTOS, though only having eight meta-instructions, can be a very powerful yet simple operating system. The key to programming within RTOS is the understanding of these eight meta commands.

Handlers:

Input/output through RTOS is achieved by the .IOX meta-command which is a machine language subroutine call. This subroutine determines which I/O device needs servicing then jumps to a service program. These service programs are called drivers or handlers. With the purchase of the RTOS program, a small number of company-written handlers are supplied. If other devices are to be added to the system, the user must supply his own handler to interface the device to RTOS. Three handlers have been written for this study

for use with RTOS. Their operation and uses will be briefly discussed.

1. Power-Fail/Auto Restart Handler:

An option available with the NOVA family of computers is the power-fail/auto-restart monitor. With this option, when AC power fails there is a delay of 1 to 2 milliseconds before the processor shuts down. The power monitor option warns the Operating System when power is failing by setting the Power-Fail flag; this results in an interrupt demand. However, for an interrupt to result, the interrupt monitor must be activated or enabled (INTEN). This option is interfaced with RTOS through a power-fail/auto-restart handler. RTOS monitors the power-fail flag for every interrupt processed.

Upon power failure, RTOS executes a jump (JMP) to the handler where all pertinent information such as accumulator content and program counter is stored. Before the processor shuts down, the restart address is stored in location PFSET on page 0 of memory and the address of location PFSET plus the indirect (@) instruction is stored in location 0 on page 0.

The user program being executed within RTOS must contain on page 0 of memory a pointer labelled PFSET. This arrangement is necessary because of the relocatability of the various programs.

Upon power restoration, the processor executes a "jump to location 0", page 0, this contains the instruction "JMP @ PFSET"while location PFSET contains the address of the restart section of the power-fail/auto-restart handler. Control is thereby returned to normal program execution.

For the automatic restart option to function, the power switch of the processor must be in the "lock" position.

The power-fail/auto-restart option is a very valuable device which works well with one exception. This will be discussed with the HP-IB handler.

2. Digital to Analog Converter Handler

The digital to analog converter handler is a program which interfaces a digital to analog converter or D/A, with RTOS, the operating system.

All transactions carried out within a computer system are done in a digital manner. However, it is possible to communicate a digital result calculated within the CPU to an external source by an analog signal. This is achieved by using a digital to analog converter. This communication between the computer and an external device could permit, for example, a computer opening or closing a valve.

The D/A handler is called upon when an .10X meta-command referring to the D/A is encountered during program execution, the D/A handler outputs a voltage signal and then holds it. No interrupts are possible from the D/A as nothing is ever input from the D/A converter. A D/A converter may

have many channels or ports where it can output an analog signal; only two channels were present in the hardware system used.

3. HP-IB Handler:

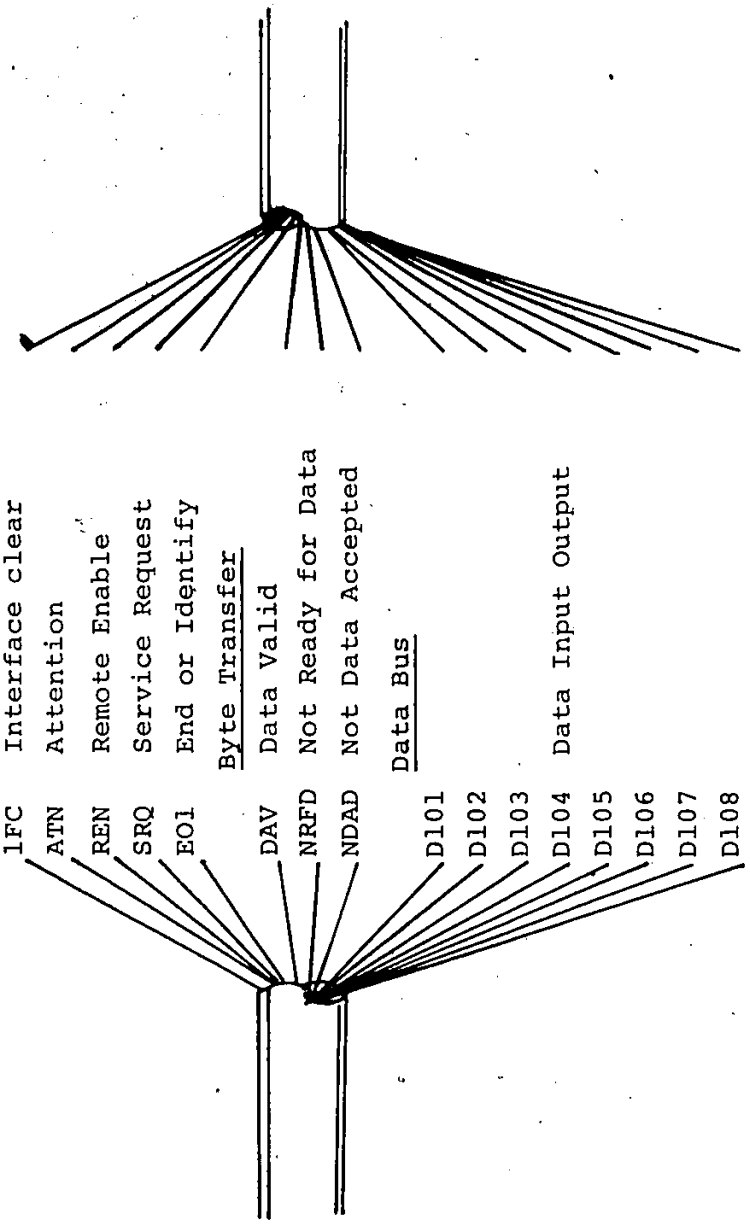
The goal of any instrumentation system is to monitor some process or perform measurements on some device. The essential communication's link between components of the system is provided by an interface system. Interfacing is a major problem in assembling a system since each device requires a separate hardware interface as well as a separate software handler. Hardware and software can be eliminated by creating a standard interface system. Such a system has been developed and is called the IEEE 488-1975 standard[8], it defines the mechanical, electrical and functional specifications in device and system independent terms; in this way the interface definition applies to a broad range of devices. The IEEE 488-1975 standard is often referred to as the HP-IB or the Hewlett-Packard Interface Bus since the Hewlett-Packard Co. developed the HP-IB system from which originated the standard.

The HP-IB has a party line bus structure with sixteen signal lines, half assigned to the (bus) data function and the other eight assigned to specific control and status functions. Figure 7 represents this structure.

Data transfer is achieved in a byte-serial bit-parallel manner on the bus and is accomplished through a three-wire handshake. The three I/O lines forming the handshake

FIGURE 7

HP-IB HARDWARE STRUCTURE



are "Ready for Data" (RFD), "Data Valid" (DAV) and "Data accepted" (DAC).

With the HP-IB it is possible to interconnect up to 15 devices with up to 20 meters total transmission length. The minicomputer connected to the HP-IB can serve not only as listener and talker on the system but also as system (HP-IB) controller. As system controller, the minicomputer would direct communications between talker and listener devices on the HP-IB.

A software driver or handler, interconnecting in a programming capacity the HP-IB with RTOS, must be written in a general manner to permit I/O data transfers for a number of devices.

In the present system, two instruments were interconnected with the minicomputer by the HP-IB; a Hewlett-Packard digital voltmeter (DVM) model 3455A and a Hewlett-Packard Scanner model 3495A. The digital voltmeter, able to receive instructions and send results, is both a talker and a listener on the HP-IB while the scanner, only able to receive instruction, is strictly a listener. Since only one device (DVM) connected to the HP-IB is capable of issuing an interrupt, no serial polling or parallel polling capabilities were included in the HP-IB handler.

The major distinction of this handler over most of the others, is that the HP-IB handler is designed for an interface system and not an individual device.

Referring to the shortcomings of the Power-Fail/ Auto-restart handler with regards to the HP-IB, upon power failure one of two things may happen: each device may reset itself as on initial power turn-on, or, each device may ignore the power failure and maintains its programmed settings. What happens is dependent on each device so a combination of both happening in a system is possible. The duration of the power failure is the key to the power failure results. If the power failure is the result of a pulse or sudden drain of voltage and is restored within 0.5 to 1 second, the effects of the power failure would be limited to possibly one false reading from the DVM. The power-fail/auto-restart handler was written for this case therefore in the event of a true power failure, power restoration would not result in system restoration. Writing a handler to cover the other cases would be very difficult and would be too user program dependent for reinitialization of instruments.

Direct Digital Control

Hardware

1. NOVA:

A Datageneral corporation (DGC) NOVA minicomputer with 8192 (8K) 16 bit words, core memory was used[7]. The NOVA is a general purpose computer with four accummulators; two of which may be used as index registers. An ASR-33 Teletype-writer served as operator's consol and to print hard copies of programs. Other peripherals included a Datageneral programmable real-time clock nominally set at 100HZ, and a power-fail/auto-restart option providing protection against AC power loss. A Datageneral high speed paper tape reader and punch served for software development. Figure 8 shows the hardware system used.

2. Output Signal System:

Outward communications was achieved by a Datageneral model 4037B two channel digital-to-analog converter. The digital-to-analog converters -10 to +10 volts output signal was strengthened by a power booster amplifier system designed to output up to 100 milliamperes at 10 volts. This enabled the digital-to-analog converter to drive two current-to-air pressure transducers. The D/A converters' output signal had a resolution of 0.02 volts and a settling time of 15 μ seconds. Figure 9 shows the output signal system.

3. Input Signal System:

The input signals from the three process sensors were relayed to a Hewlett-Packard 3495A Scanner[10] which served as a programmable multiplexer. The model 3495A Scanner

FIGURE 8

HARDWARE SYSTEM

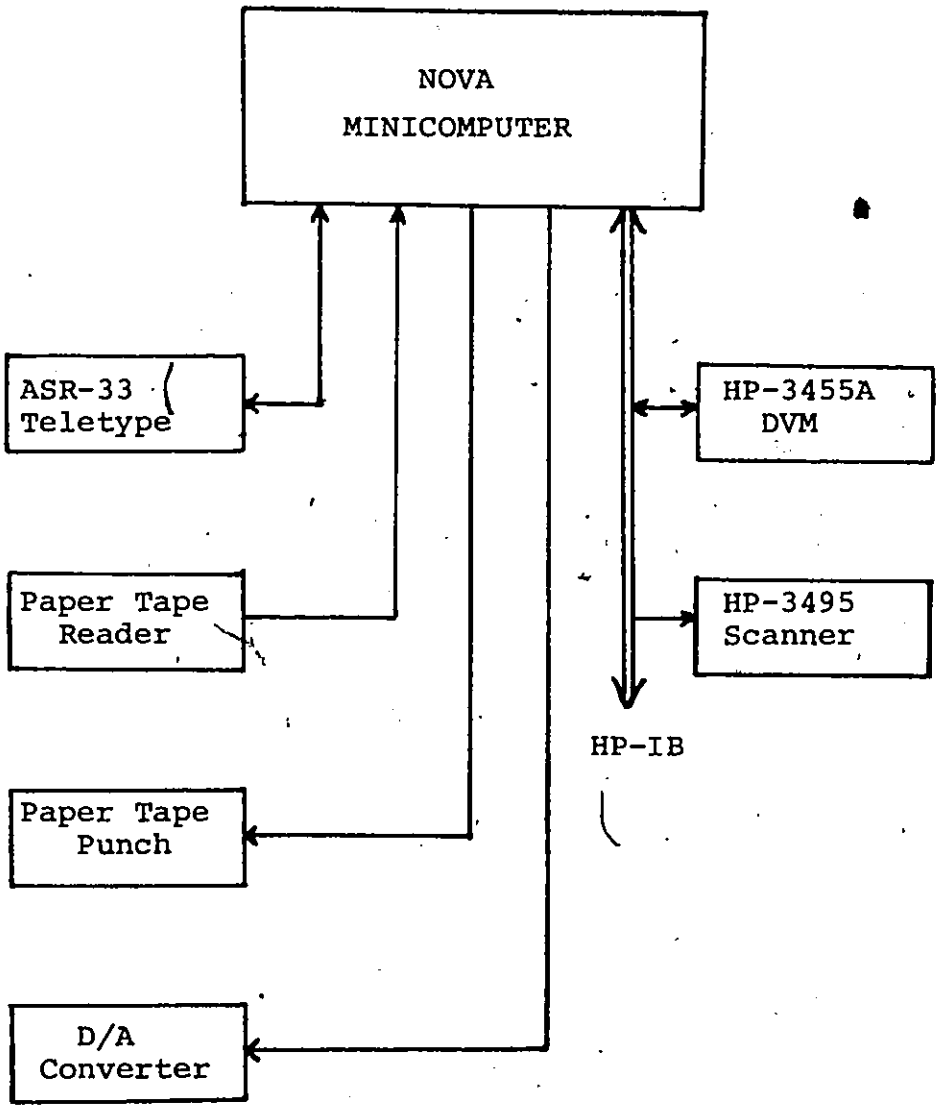
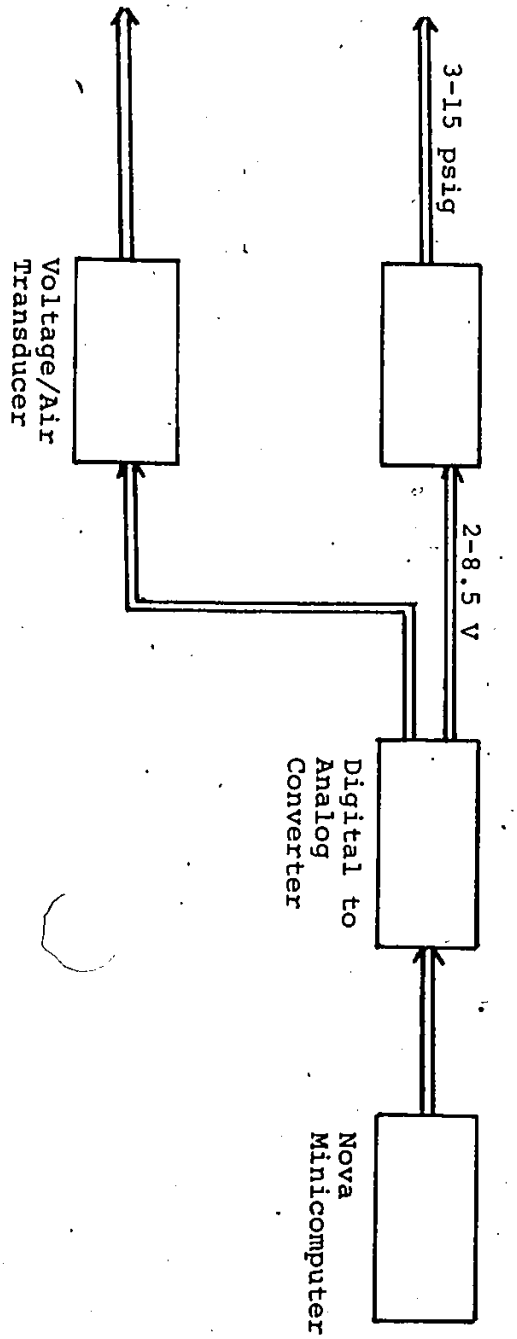


FIGURE 9

OUTPUT SIGNAL SCHEME TO MINICOMPUTER



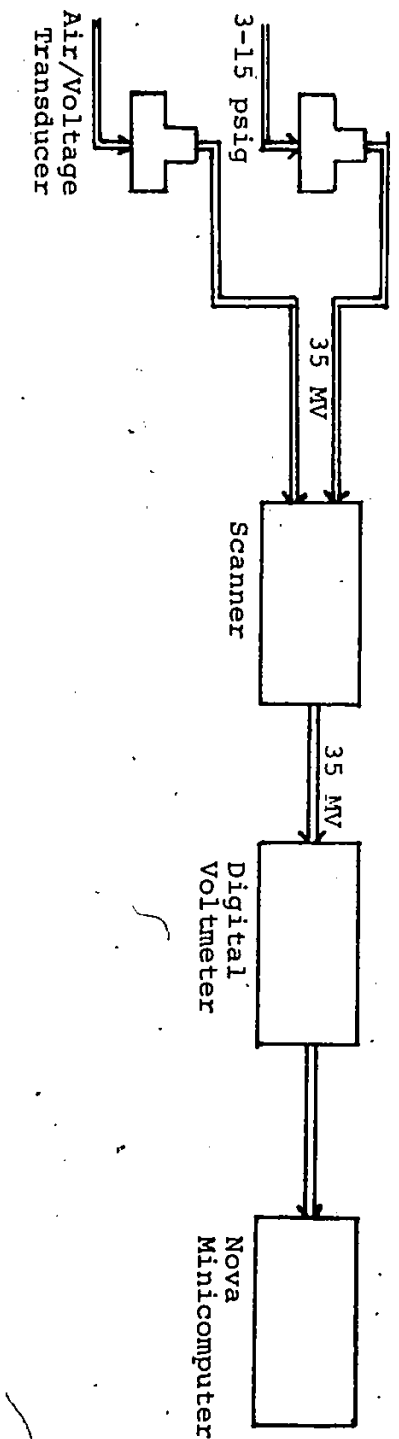
was loaded with a ten channel "low thermal" decade. The system could be expanded to any combination of four "low thermal" or "activator" channel decades. The "low thermal" decades are a ten-to-one analog multiplexer allowing ten input signals to be monitored by one measurement instrument in a break-before-make sequence. The activator decade relay have two sets of normally open contacts with four terminals per channel. Any number of channels in an actuator relay may be closed or open simultaneously. A Hewlett-Packard 3455A programmable digital voltmeter (DVM) measured the signal and inputed a digital number through the HP-IB data lines. Both the Scanner and the digital-voltmeter were connected to the NOVA minicomputer through the HP-IB interface. The model 3455A digital voltmeter makes AC voltage measurements with 5 digit resolution and DC voltage and resistance measurements with 5 or 6 digits resolution. The 3455A employs an automatic calibration feature which corrects for possible gain and offset errors in the analog circuitry to provide maximum accuracy. The 3455A digital voltmeter is fully HP-IB programmable for system applications. Figure 10 shows the input signal system.

4. Interfacing, Transducers

The minicomputer's analog output signal was converted to a pneumatic signal through the use of a Foxboro model 69TA-1 current-to-air transducer. The input signal ranged from 10 to 50 milliamps for an output of 3 to 15 psig.

FIGURE 10

INPUT SIGNAL SCHEME TO MINICOMPUTER



Air supply to the transducer was set at 20 psig. Although driven by current, the transducer was controlled by varying the D/A converter's output voltage signal; hence the need for a power booster amplifier. The power booster amplifier design is represented in Figure 11.

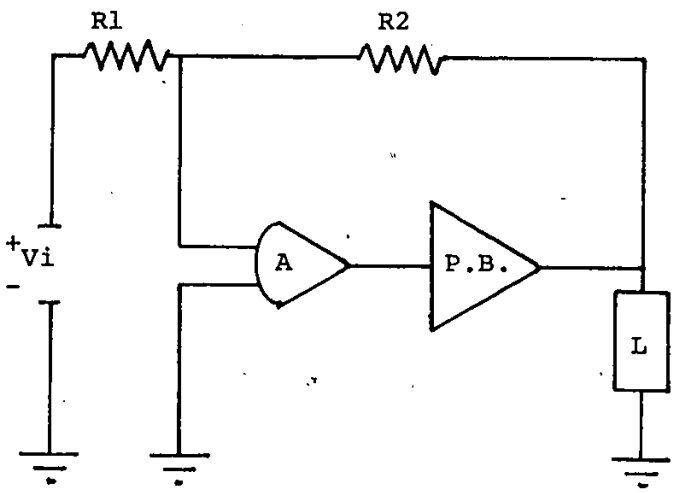
The sensors located on the distillation column output a 3 to 15 psig air signal. Therefore an air-to-voltage transducer was needed to enable interfacing with the scanner and digital voltmeter. The pressure transducers employed, utilized a bonded foil strain gage sensing element. Viatram tranducers, model 103, were employed, each having a range of 0 to 15 psig input and an output of 0 millivolts at 3 psig input and 2.5 millivolt times the power supply voltage at 15 psig. The power supply use to drive the pressure transducers provided 14 volts DC. Therefpre, full scale (15 psig) pressure to transducer outputs 0.035 volts.

5. HP-IB

The Hewlett-Packard interface bus system was employed as a result of a need for a new input signal monitoring system. The existing system, an analog-to-digital converter was limited by its maximum number of inputs (two) and more seriously by a persistent "noise" problem. This noise problem was the result of electrical interference on the various input signals. Ching[11] studied the problem and tried to eliminate the distortions by means of a hardware and software signal filter. The final conclusion was to

FIGURE 11

POWER BOOSTER AMPLIFIER DESIGN



R1 = 100K Ω Resistance (±5%)
R2 = 100K Ω Resistance (±5%)
Vi = Input Voltage
A = 3077/12C Burr Brown Operational Amplifier
PB = 3329/03 Burr Brown Power Booster Amplifier
L = Load, Current-to-Air Transducer

buy a system able to reject such distortions. A system most suitable to our needs, was the Hewlett-Packard digital voltmeter model 3455A. To complete the system, a compatible multiplexer Hewlett Packard model 3495A was also purchased. The hardware/software link between this equipment was achieved through the HP-IB.

Other advantages favouring the HP-IB system are its versatility and adaptability. It can readily accept a considerable amount of equipment with a minimum of software and hardware changes needed to the existing system. No special interface board is necessary for additional equipment to be added to the HP-IB.

Direct Digital Control

Software

Software, as opposed to hardware, is all of the material used in the development and implementation of programs used within the computer system[9]. The basic requirements of any software package varies in relation to its end use. For a direct-digital-control computer system, the software requirements are described as follows:[5]

1. Development Programs: Various developmental aids are supplied by the computer firm. These programs are usually in the form of editor, assembler and debugging programs. Editor programs are used in the actual writing of the program and in making corrections to it. Assembler programs are used in translating computer instructions written in mnemonic form into a machine compatible format. Debugging programs permit temporary stops in an executing program at various chosen locations, enabling the user to verify his program. These are but a few of the aids used in program development, others include program loaders and compilers.
2. Operating System: For any type of direct-digital-control software package, there must be some sort of program which is used to coordinate the overall system. Whether this coordinating program is directly incorporated within the basic control program or in a separate form, it must be present. In a separate form, it is called an operating system and

is usually written in a general non-dedicated manner to accommodate many different end users. The operating system schedules execution of programs according to priority, timing or special request. All instruments connected to the computer must be linked to the operating system through a small program called a handler.

3. Computation Aids: These aids are written in the form of general programs or subroutines and are usually supplied by the computer manufacturer. Depending on the computer firm, the extent of these supplied programs may vary greatly. Two examples of program aids are, a floating-point-interpreter and a subroutine to convert numbers in ASCII format into binary numbers.

4. Control Program: This program implements not only the control algorithm, but also is required to scan the process inputs according to a sampling scheme and convert this data into computer acceptable units. After implementing a control algorithm, the results must be output through an output system. Other important parts of a control program include data logging and operator communications; the user must be able to change control settings while the computer is still controlling the process. The control algorithm section is the essential part of a control program and must be written in a modular fashion to permit easy interchanging of control schemes.

When a direct-digital-control computer software package is devised, these four sections must be included. They should be well documented to permit a second user to use the fundamental framework of the developed system and improve upon it.

In the developed software package, the basic framework is modular enough to permit a new user to simply add-on subroutines to suit his own requirements.

All parts of the control program package were written and tested individually; after successful operation they were linked together to form the control program. Extensive program testing was done using the various "Debug" programs available from Datageneral Corp. The control scheme consisted of a proportional-integral (P-I) control algorithm. All operator communications used through the control program presented the various control parameters in their true units. That is to say a proportional gain (Kc) of 1, representing an error of 1 psig, resulted in an increase of the output by 1 psig, similarly the integral constant τ_I, had units of seconds to the power minus one.

The first three of the four requirements needed for a direct-digital-control software package, were met through the use of Datageneral furnished programs. The only exception, relating to software developed by the author, pertained to various instrument handlers.

The last requirement, the control program, will be broken down into subsections and discussed individually. Figure 12 shows the logic flow chart of this program.

1. Reference Table:

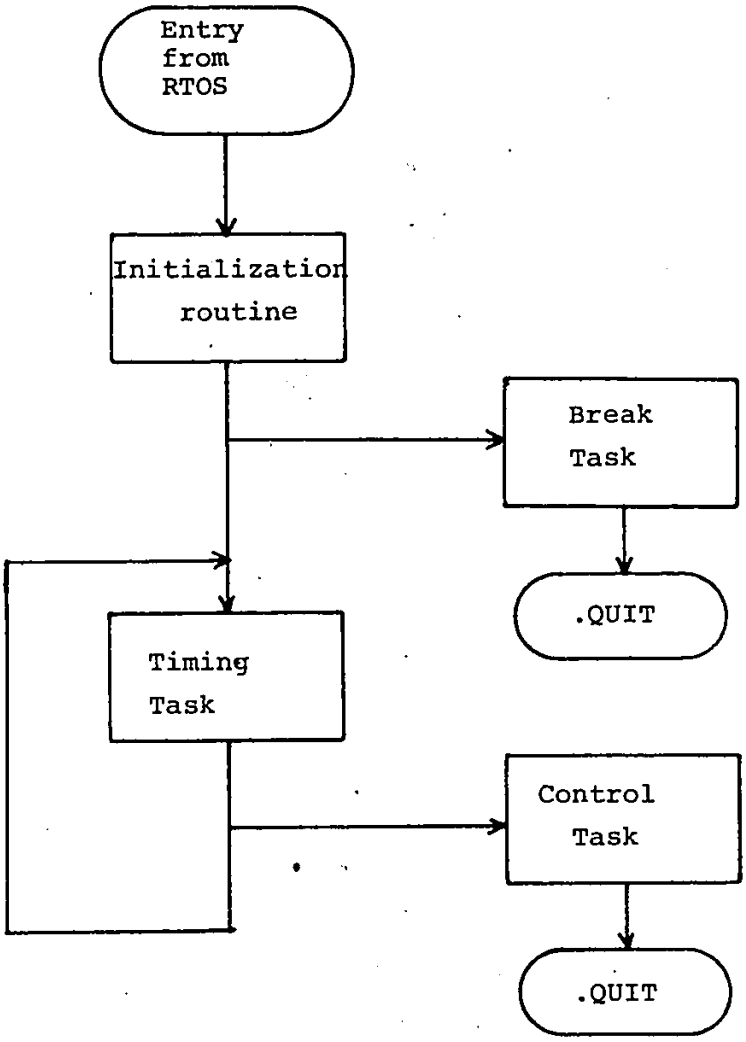
This section is the central data base of the control program. It contains both a page 0 and a page 1 relocatable section. Since page 0 memory can be referred to from anywhere in core memory, it was used as a reference table for all variables and constants. The page 1 memory section was used for data not necessarily referred to in the control program notably the floating-point interpreter work area; this 120 memory location area was used as a small scratch-pad memory for computations within the floating-point interpreter. The data base of this program is easily expandable and permits straight-forward addition or alteration of variables.

2. Initialization Program:

The initialization program serves three functions: it serves as an entry point to the control program from RTOS, it serves as an initialization routine for the control program and finally it is used as a control loop timer. Entry point from RTOS is determined by the use of the label "START:" which is declared an external in RTOS. The initialization subsection assigns a priority to the task for RTOS, calculates various constants and initializes both the floating point interpreter and the digital voltmeter. Before proceeding into the

FIGURE 12

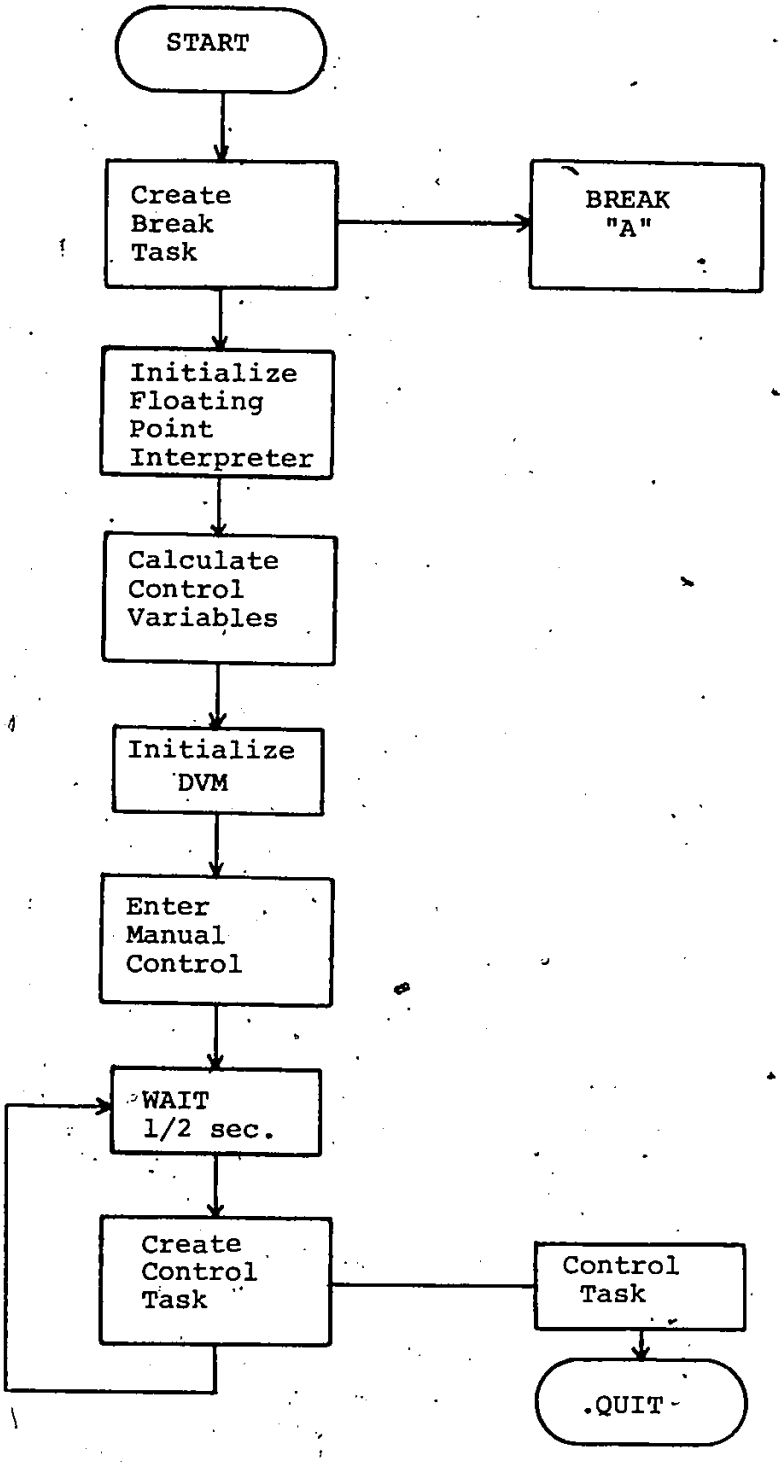
CONTROL PROGRAM OVERALL TASK EXECUTION



control loop timer routine, the initialization program transfers into the manual program to permit process start-up. Upon successful start-up, the operator signals completion and the program returns to the initialization routine to enter the control loop timer. For the present study the control loop timer was set for 0.5 seconds, meaning that control action took place every 0.5 seconds. This time constant is dependent on the complexity of the control algorithm and the time required to implement it. It can easily be changed to accommodate any particular scheme. The mechanism of the control program is simplified by the use of various meta-instructions available from the operating system. Within the timing loop, the control action task is created at a higher priority level than the executing task. A "WAIT" instruction is executed, permitting the task scheduler to executed any other task during the 0.5 seconds. During the .WAIT time control is transferred to the control algorithm task. Upon termination, the computer is idle for the remaining time of the .WAIT meta-instruction. The excess time may be minimized to suit individual needs; in this particular case the excess time was approximately 0.15 seconds. Upon completion of the .WAIT meta-instruction, the control task is reinitialized through the .FORK meta-instruction. Program execution is returned to the .WAIT meta-instruction thereby completing a loop. This scheme permits the user to know the control loop-timing. Figure 13 represents the logical flow path of the

FIGURE 13

CONTROL PROGRAM INITIALIZATION TASK



initialization program.

3. Control Algorithm Program:

The control loop task uses a proportional integral control scheme. Inputs are from the feed rotameter, the reflux rotameter and the steam d/p cell, therefore all inputs are flow measurements. Outputs are limited to the steam and reflux control valves; the feed stream was monitored to permit future program development to include a feed-forward control loop. Each channel or input is monitored before any control action is calculated. Upon completion of the control action calculations and outputs to the valves, a parallel task of lower priority is created through the .FORK meta-command. This lower priority task, upon termination of the control action task, recreates it. This format is necessary if the control loop timer program is to be used and because one task cannot recreate itself while still being executed.

A special section was added to the control action task to permit a "bumpless" transfer from manual operation to computer control. Upon termination of the manual control task and entering the automatic control loop, the first input from each process sensor was used as the loop set point rather than the manual setting output by the user. This routine was necessary since the output to the valve varied greatly with the input from the flow meter during steady-state operation. A subroutine called "DAEXT" was written to insure that the

result of the control action did not exceed process equipment limits. A logic flow chart of the control algorithm routine is shown in Figure 14.

4. User Communication Programs:

The next four sections of the control program can be grouped together since they are, in reality, one task with respect to RTOS. These sections link the user with the control program; they include a set point change routine, two control parameter change routines and a manual operation routine. Each section, although coming from the same origin, will be discussed separately.

a) Break Routine: This routine is entered from the initialization or control tasks by the use of RTOS's "BRK" meta-instruction. The user forces entry into this task by the input of an "A" from the teletype. This constant ("A") may be changed to suit the user. Once the task (break task) is started, a teletype input is demanded. This input determines which type of user communication is needed; an input of AM indicates that the user wants to enter the manual operation task, AR represents a reflux control parameter change and so forth. There are five such tasks, if by mistake an error is committed the task terminates itself permitting the user to restart. From this routine, the "BREAK" task branches to one of the various communications routine. A flow chart of this routine is presented in Figure 15.

FIGURE 14

CONTROL PROGRAM CONTROL TASK

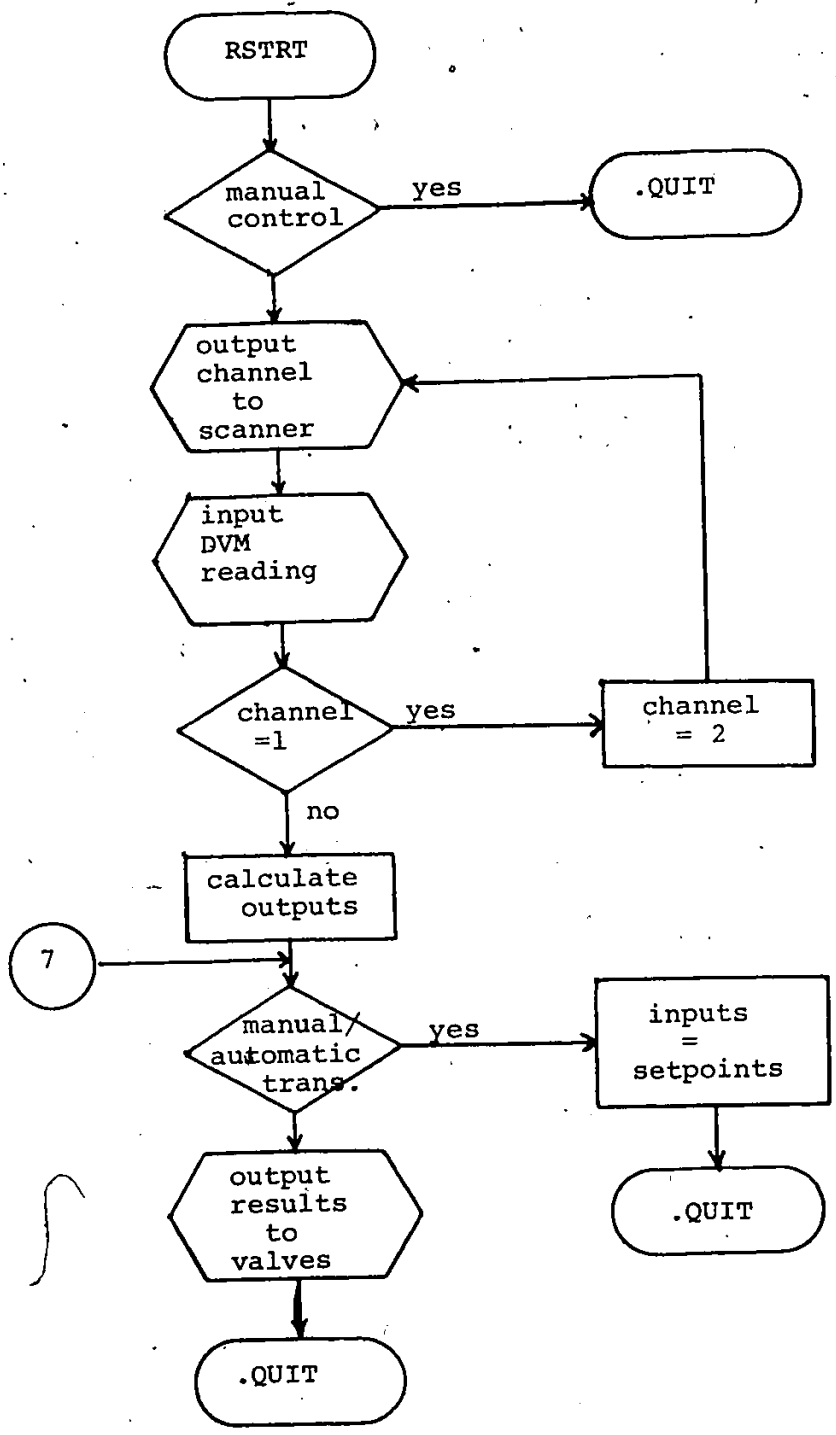
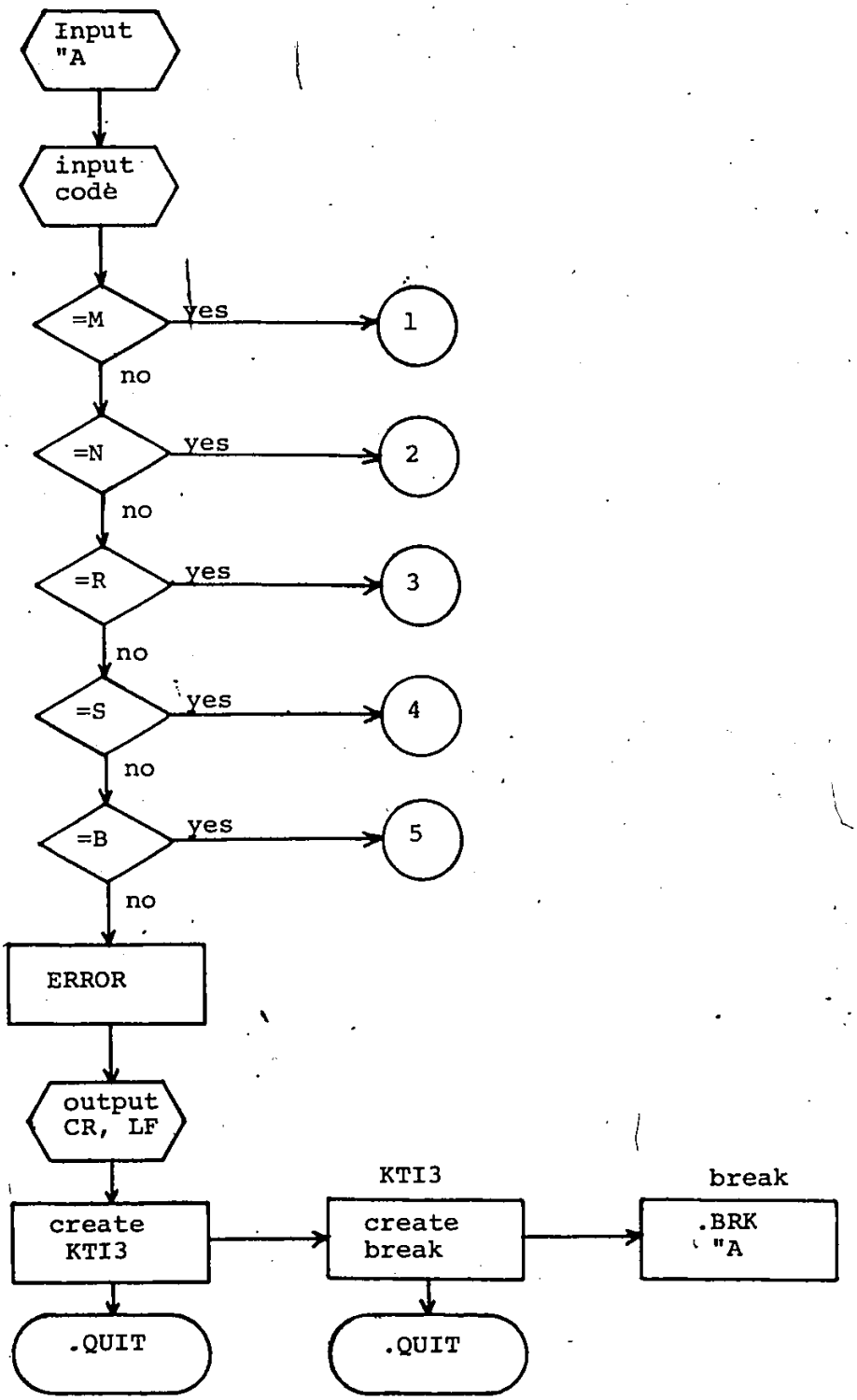


FIGURE 15

CONTROL PROGRAM BREAK TASK



b) Steam k_c, τ_I Changes: Upon input of the letters AB on teletypewriter, program execution is dedicated to the k_c, τ_I steam control variable change routine. This routine outputs the present constants and demands the input of new ones. Since the priority number of this task is higher than that of the control task and control loop timer task, the execution of these last two is unhindered. The steam k_c, τ_I change task is executed simultaneously with the other two, the execution in reality, is done during the excess .WAIT time in the control loop timer. Therefore, to permit the cocurrent execution of the control task and the user communications task, it is necessary to have some "idle" time included in the .WAIT meta-instruction. All changes done through this routine are implemented immediately. A logic flow chart of this routine is shown in Figure 16.

c) Reflux; k_c, τ_I changes: This routine, entered by an input of AR on the teletypewriter is essentially identical to the previous routine except that it pertains to the reflux control loop. An amalgamation of these two routines would not result in any memory savings because each routine, although structurally identical, has its own particular variables. Therefore, for ease of debugging and future alterations they were written separately. Both these tasks upon termination recreate the "BREAK" task. A logic flow chart of this routine is shown in Figure 17.

FIGURE 16

CONTROL PROGRAM STEAM K τ_I CHANGES ROUTINE

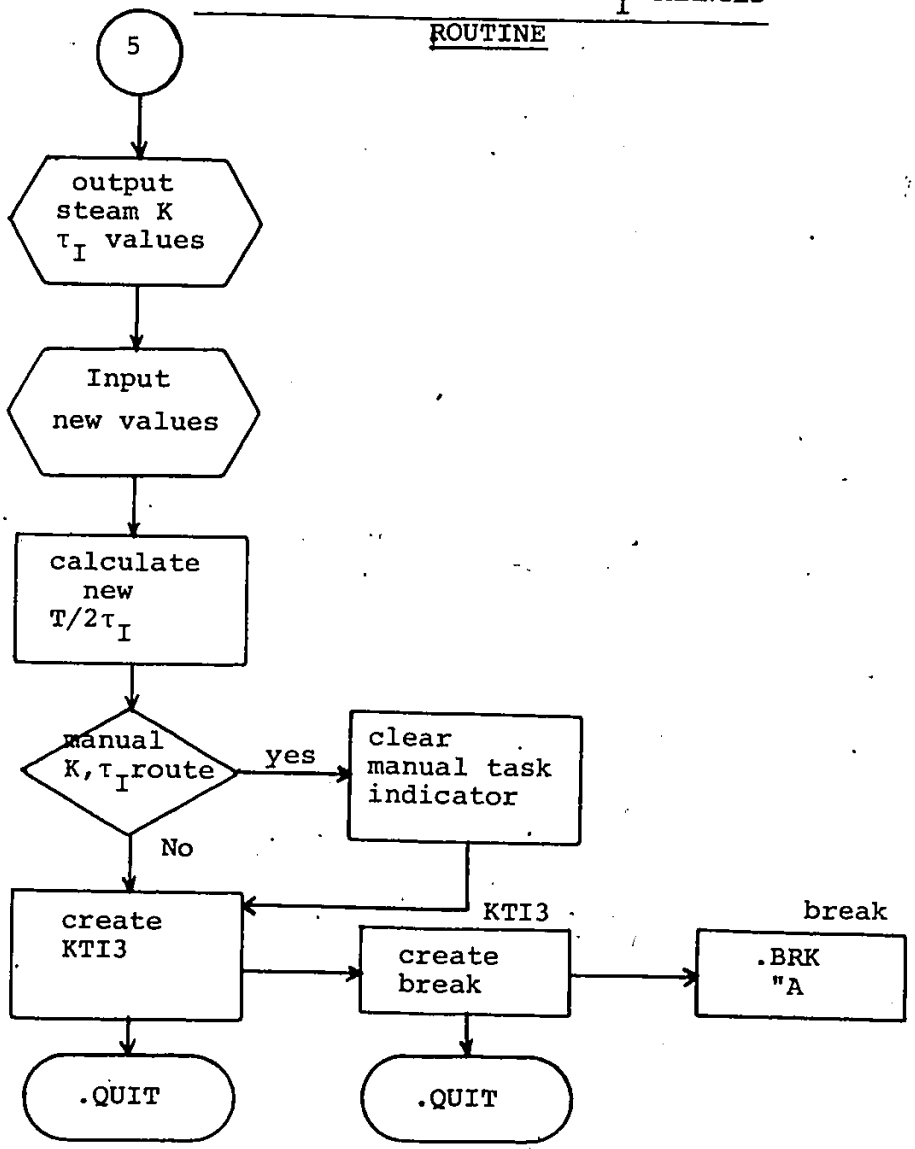
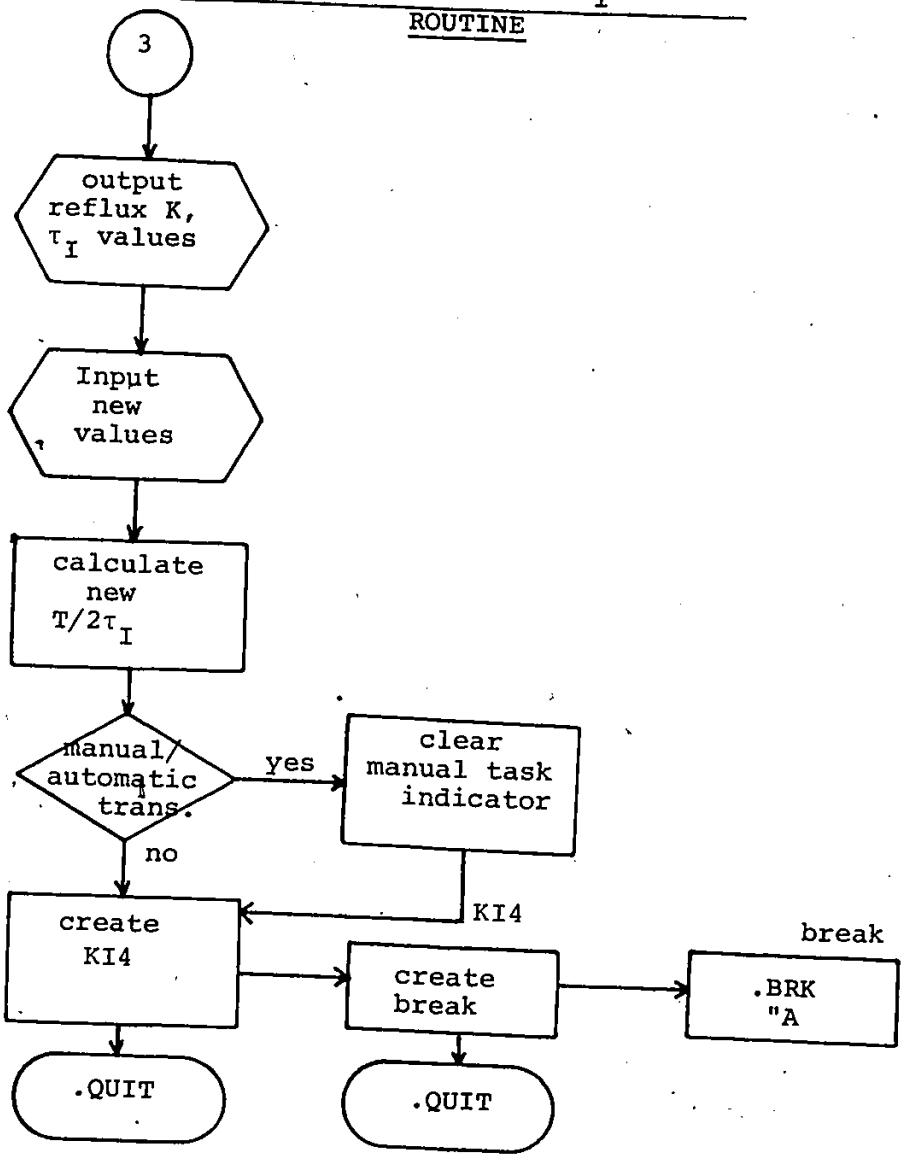


FIGURE 17

CONTROL PROGRAM REFLUX K, τ_I CHANGES ROUTINE



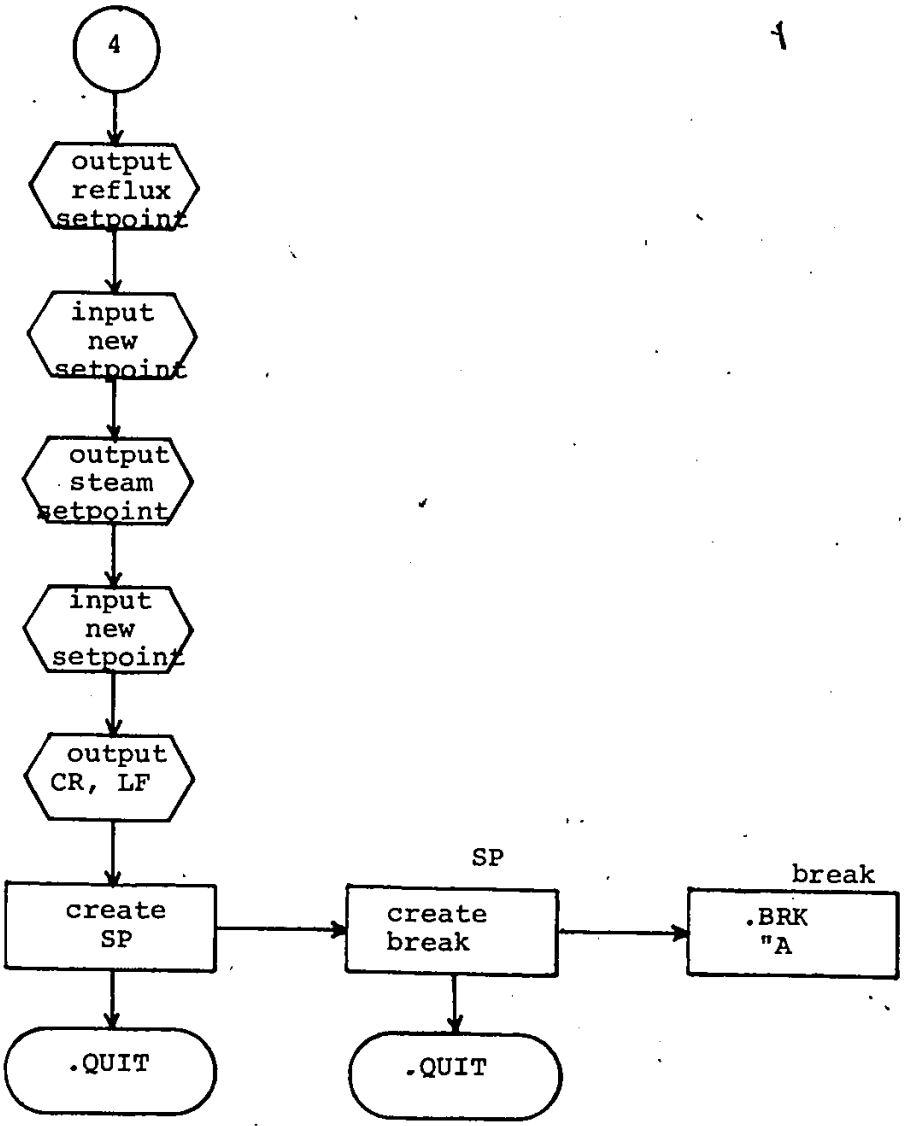
d) Set Point Changes: By the input of AS on the Teletypewriter, the program jumps to a routine which permits set point changes to both the reflux and steam control loops. This routine is not intended to act as a process dynamics testing routine. Having a higher priority number that the control loop and the control loop timer tasks, it is executed during the control loop timer "idle" time. It outputs a message, the present setpoint, and then demands a new set point, the procedure is repeated for the other control loop. This procedure, although very convenient for user interaction is much too slow for process dynamics calculation. Figure 18 shows the routine's logical flow path.

e) Manual Operation: This routine may be entered from two different locations depending on the teletypewriter input; an input of AM results in a listing of operation instructions while an input of AN proceeds directly into the manual control program. After entering the manual control routine, one of the first operations performed is to increment the manual control variable ($\neq$o). This variable, always tested for zero before entering the control loop task, will prevent control loop task execution while on manual control. This contrasts with the simultaneous execution of previous user routines with the automatic control loop.

The structure of the manual control routine is straightforward. The instruction message (if called by AM) states that an input of 0 results in a 3 psig output while

FIGURE 18

CONTROL PROGRAM SETPOINT CHANGES ROUTINE



a 35000 input to the teletypewriter results in a 15 psig output to the control valve. Each control valve is identified by a letter proceeding the numerical input; an "R" represents reflux while an "S" indicates steam. An input of "T" or of any other letter results in the termination of the manual control task. Before the manual control routine termination the user is asked whether any k_c, τ_I changes are in order and if "yes", for which control loop. Upon completion of the k_c, τ_I changes routine (assuming "yes"), the manual control routine again asks whether there are any k_c, τ_I changes to be made. This looping will continue until the user responds with a "no". As with the initialization routine, upon return to automatic control, the first process sensor inputs and not the user's output value are used as the new process set points. Termination of the manual control routine results in the creation of a new task which is used to reinitialize the "Break" task.

A manual control routine is essential for debugging an automatic control loop and for process start-up. A two-point entry system was found to be beneficial for subsequent uses of the manual control routine or for the experienced user for whom the instruction message was repetitive. Figures 19 and 20 represents the manual control routine's logical flow path.

FIGURE 19

CONTROL PROGRAM MANUAL CONTROL ROUTINE 1/2

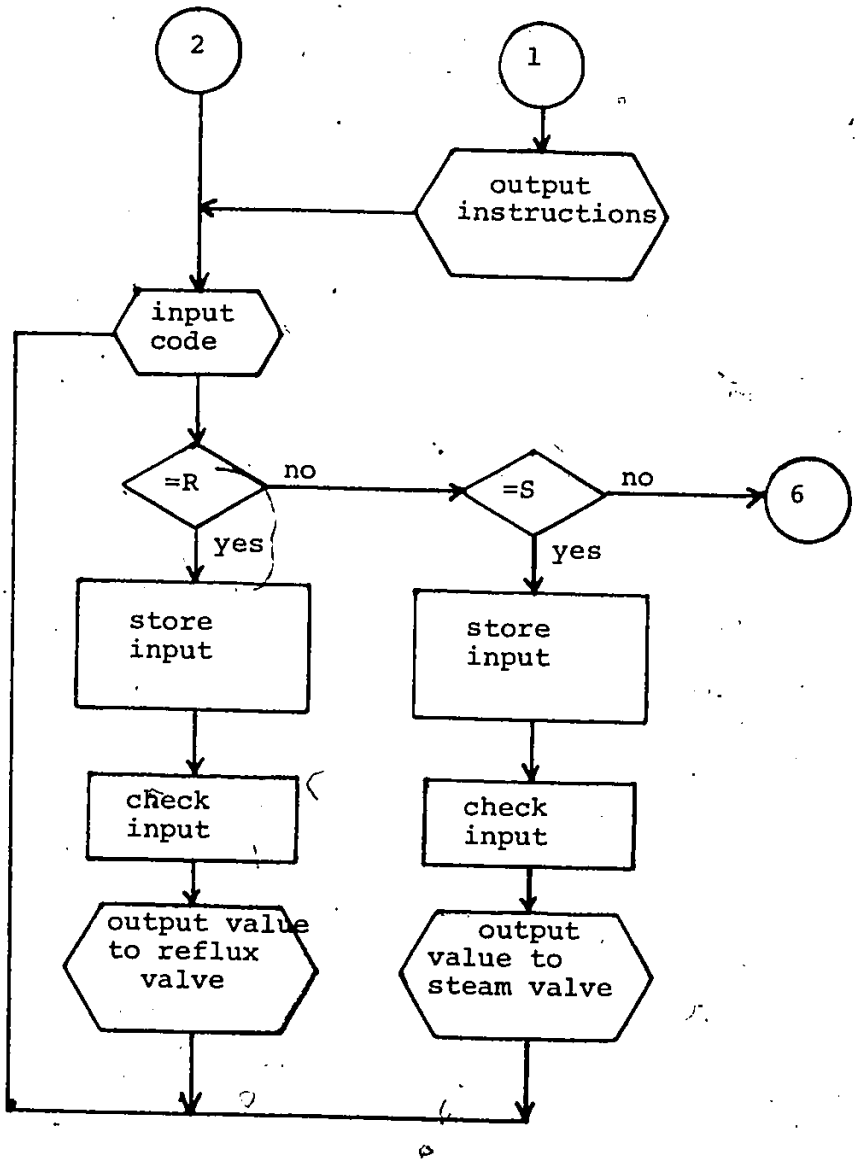
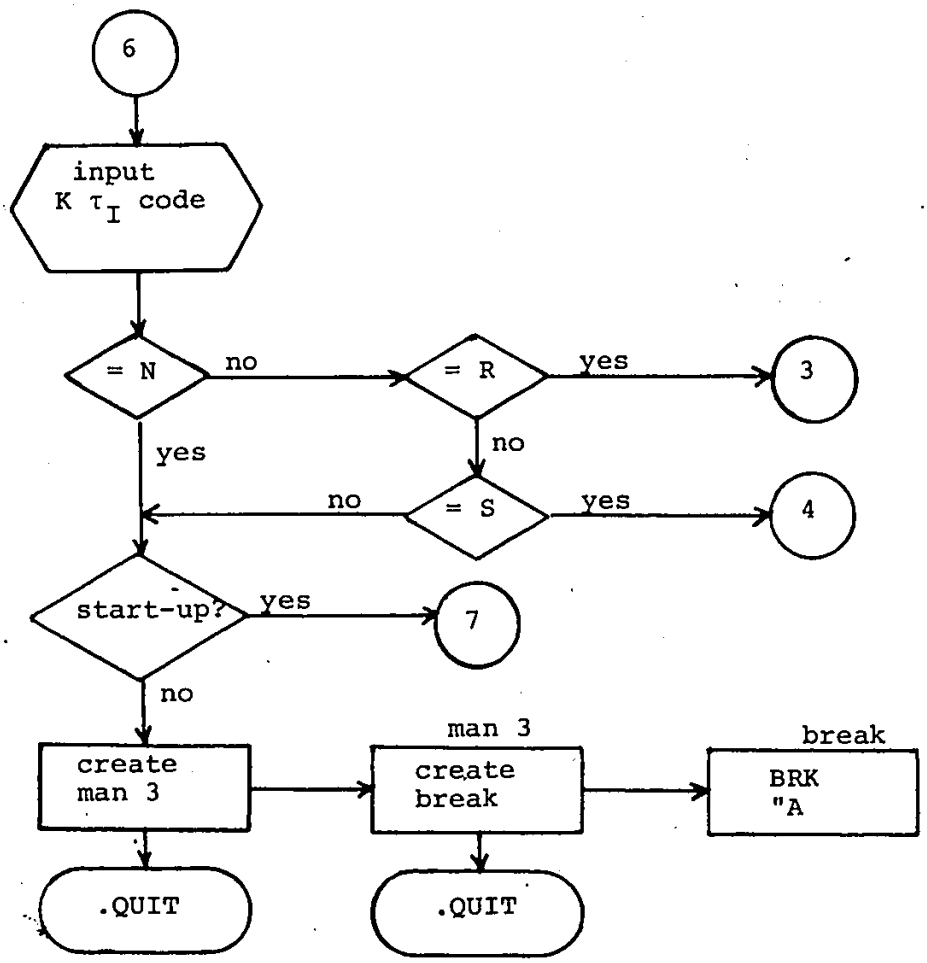


FIGURE 20

CONTROL PROGRAM MANUAL CONTROL ROUTINE 2/2



5. Support Subroutines:

Two relocatable subroutines which could be called from anywhere in the program by using the page 0 reference table, were used in input/output handling. One routine was used to convert an "ASCII" character string to a single precision binary number, the other to convert a single precision binary number to an "ASCII" character string (decimal format) and output it on the teletype. Both subroutines are based on Datageneral supplied software but were modified for use with RTOS.

Another subroutine, stored in the reference section on page 0, was written to permit resetting (setting to 0) of an input storage area. This subroutine, called "CLEAN" has a storage area of 21 memory locations. Still another subroutine, called "DAEXT", was used to check for D/A output extremes. A maximum of 15 psig and a minimum of 3 psig were the permitted limits.

6. Floating Point Interpreter:

The extended version of the Datageneral Floating-Point Interpreter (Datageneral No. 091-000013) was used for mathematical computations. Being a separate program, linking between the floating-point-interpreter and the user software, was achieved through the use of meta-commands; "FENT" to enter the interpreter and "FEXT" to exit it. Rules and regulations concerning the Floating Point Interpreter are well defined in the program manual. Being an interpreter,

instruction execution was slow but the use of the interpreter was still justifiable; it liberates the user of writing numerous mathematical operations subroutines. The use of a floating point format also permits very large numbers to be calculated with a high degree of precision.

7. Programming with the HP-IB

(i) Digital Voltmeter:(DVM) During the initialization section of the control program, the DVM is initialized. This consists of sending a bus-clear instruction which is equivalent to IORST for the NOVA computer. Then talker control was enabled permitting the NOVA to act as both system controller and talker. The attention line is then set high indicating that instruction commands are to be sent on the data-lines. The DVM's listen address, "66", is sent on the HP-IB data-lines, setting the DVM in listen mode. For both the DVM and the Scanner the listen-talk addresses were set by the user. Attention is then set low indicating a data transfer. There are seven variables on the DVM which must be set; the property to be measured, the range, the measurement scheme, the use of the math option, high resolution, auto-calibration and finally the data ready switch. Once these variables are set, the attention line is set high and a universal unlisten command is sent.

During the control loop, to take a reading, only a triggering is required to activate the DVM. This triggering is achieved by setting the NOVA as system talker and attention

high. With the DVM addressed to listen and attention reset low, a manual/hold instruction is sent. This triggers the DVM into taking a measurement. The system controller, the NOVA, is set as a system listener and a start pulse is sent on the HP-1B resulting in an interrupt request. Subsequent measurements are achieved in a similar manner.

(ii) Scanner: To program the Scanner, the NOVA is set as system controller and talker. The attention line is then raised and the Scanner's listen address, in this case "51", is output on the data lines. In the data mode with attention lowered, three instructions are needed to program the Scanner; select the proper channel decade, select the channel needed and execute the proceeding instructions. Execution is achieved by the output of "15" or an "ASCII" carriage return. Attention is then set high and a universal unlisten instruction is sent.

Experimental Run:

To demonstrate the stability of the developed control program and the feasibility of direct digital control, various experimental runs were conducted on a distillation column using a 17 mole % methanol water feed.

The distillation column, as described in previous sections, was interfaced with the minicomputer; the minicomputer was used as a controller for two control loops. The control settings for the proportional-integral control scheme used to control the column's reflux and bottoms flowrates, were determined in an on-line trial and error manner. These settings, by no means optimum, provided an adequate means of testing the control system. In all the trial runs, the control settings were kept constant. Appendix III lists the control settings and various process conditions for 3 runs of a 17 mole % methanol-water solution distillation. These results are appended for matter of record and do not indicate long term control stability or accuracy.

For these experimental trial runs, thermocouples, rotameters, and control valves were not calibrated. This was not done as the essential goal of this work was not to study the dynamic behaviour of the distillation column but rather to demonstrate the feasibility of computer software for control. The computer control system proved itself to be effective and stable in maintaining the distillation column at steady state conditions. The serious noise problems

encountered by Ching[11] were avoided with the new input signal analysis system. Both the DVM and Scanner performed as specified by the manufacturer to filter out distortions.

RESULTS & DISCUSSION

The objectives of this study was to develop a flexible software package for direct-digital control, to develop a hardware configuration not needing constant changes and to implement the first two systems by controlling a ten tray distillation column using a P-I control scheme.

The software package developed included an operating system, support software and a general control program. The real-time operating system, being non-dedicated, can be used in its present form in future work. The support software is well documented and tested, permitting straightforward use. The control program is written in a modular fashion so it can easily be ammended to suit future needs. The three handlers written, the HP-IB, the D/A and the Power Fail/Auto-Restart handlers, work well and need not be changed. Extensive debugging and testing was performed to assure reliability and accuracy. The only change needed would be if other equipment is to be added on to the HP-IB system; the HP-IB handler interrupt servicing routine doesn't have polling or broadcasting capabilities. In the present system this was not necessary since only one instrument talker (DVM) was present on the system.

The hardware package developed is now fairly complete with respect to direct-digital-control. Expansion would be needed if complexity of programming increased or if

a FORTRAN compiler is to be used. If the number of controlled variables is to increase, a new D/A converter would be needed since the D/A converter presently used has only 2 channels.

With respect to the hardware and software system developed, the number of inputs or output variables could easily be increased to accommodate a different control scheme.

Within the developed software package the author's personal contribution included the writting and debugging of three I/O handlers; the HP-IB, the D/A converter and the Power Fail/Auto-Restart handler, and the design and debugging of the control program. RTOS, although a Datageneral Corp. issued program, required additional debugging and had to be revamped to accommodate the three additional handlers; this work was also done by the author.

The hardware system developed centered around the Hewlett-Packard Interface Bus System. The decision to purchase this particular system over others was of great importance since all future process control work was to be based on the chosen hardware configuration. Accuracy and system flexibility were key issues. The evaluation and selection of the HP-IB system was in part the responsibility of the author. All hardware connections and interfacing was also done by the author with the exception of the HP-IB/Nova computer hardware interface which was purchased from Datageneral Corp.

CONCLUSION & RECOMMENDATIONS

The fundamental objective of this study was to enable future work to be carried out in computer process control in a more structured and uniform manner. The hardware configuration developed will enable all future work to be done with a minimum of hardware interfacing. The system developed is simple, accurate and easy to maintain. With this hardware system in place, future control work be concentrated in the actual control as any necessary expansion such as expanding the number of D/A output channels would only be duplicating existing facilities.

The software system developed is quite complete and is built around an accurate yet simple control program. The basic design of this control program permits most changes to be made with a minimum of work.

However, whatever the realizations of this work, all advantages would be lost if future work is not based on past experience; future work should consist of adding to the accumulation of expertise and experience. In the past, all research was done without continuity so expertise and experience were lost.

Future expansion may follow two routes; one being a general system expansion, the other would be the development of a new control approach. A system expansion would consist of the purchase of new equipment; a first priority would be

a new D/A converter. The present D/A converter has only two channels with a maximum expansion up to six channels. Since compatible new channels are very expensive, a new D/A converter, possibly connected to the HP-IB would be cheaper. By using the HP-IB as system interface, only the D/A converter need be purchased thereby saving the cost of the otherwise necessary interface board. Other priorities include process interface instruments such as transducers. The second route would be a radically new system approach; this would necessitate the purchase of an additional 8K of memory, a floppy diskette system and software. This new system approach would permit all future work to be carried out in a high level language such as FORTRAN. By using a FORTRAN compiler, the uninitiated researcher would be alleviated of using ASSEMBLER, a possible saving of 3-6 months in programming time for major work.

REFERENCES

(1) Shinskey, F.G., "Distillation Control", McGraw-Hill (1977).

(2) Smith, C.L., "Digital Process Control", Intext Educational Publishers (1972).

(3) Coughanowr, D.R., and Koppel, L.B., "Process Systems Analysis and Control", McGraw-Hill (1965).

(4) Chan, A.L., Ph.D. Thesis, University of Ottawa, Ottawa Ontario (1973).

(5) Weitzman, Cay, "Minicomputer Systems: Structure, Implementation and Application", Prentice-Hall (1974).

(6) Korn, G.A., "Minicomputers for Engineers and Scientists", McGraw-Hill (1973).

(7) Hewlett-Packard Ltd., Digital Voltmeter model 3455A, "Operating and Service Manual".

(8) Hewlett-Packard Ltd., Scanner model 3495A, "Operating and Service Manual".

(9) Eckhouse, R.H. Jr., "Minicomputers Systems: Organization and Programming (PDP-11)", Prentice-Hall (1975).

(10) Auns, J., Ph.D. Thesis, University of Ottawa, Ottawa Ontario (1972).

(11) Ching, A.K., Masters Thesis, University of Ottawa, Ottawa, Ontario (1977).

(12) Datageneral Corp., "Introduction to Programming the NOVA Computers", (1972).

(13) Datageneral Corp, "Fundamentals of Minicomputer Programming", (1973).

(14) Datageneral Corp., "Relocatable Loader Manual", document No. 093-000039-00, (1969).

(15) Datageneral Corp., "Relocatable Assembler Manual", document No. 093-000040-00, (1969).

(16) Datageneral Corp., "Floating Point Interpreter Manual", document No. 093-000019-01, (1969).

(17) Datageneral Corp., "Real Time Operating System Reference Manual", document No. 093-000056-05, (1971).

APPENDIX I

Glossary of Terms

accumulator: (hardware)

small "scratch pad" memory in CPU used in programming.

actuator relay: (hardware)

switching mechanism not using special relays to protect against signal distortion, used in Scanner.

analog: (hardware)

represented by means of continuously variable physical quantities, contrast with digital.

ASCII code: (software)

American national standard code for information interchange, standard code using a coded character set consisting of 7-bit coded characters.

assembler: (software)

translates an assembler language or a low level language instruction into a machine language instruction on a one-to-one basis.

binary: (software)

refers to a number base; base two

bit: (hardware)

a binary digit, either of the digits 0 and 1, smallest part in a computer word; for the NOVA minicomputer system 16 bits=1 computer word.

bumpless transfer (software)

refers to switch over from manual to automatic control without any signal disturbances.

byte: (hardware)

a binary element string operated upon as a unit, in the case of the NOVA minicomputer equals 8 bits or 1/2 computer word.

central processing unit (hardware)

CPU, a unit of a computer that includes circuits controlling the interpretation and execution of instructions.

compiler: (software)

translates one compiler language or high level language instruction into many machine language instructions.

controller: (hardware)

device connected to HP-IB which controls the operations on the bus.

core memory: (hardware)

a magnetic storage in which the storage medium consists of magnetic cores.

CPU: (hardware)

refers to Central Processing Unit.

current-to-air transducer: (hardware)

instrument which converts an electric current into an air signal, needs an air power supply.

D/A converter: (hardware)

refers to a digital to analog converter

data logging: (software)

orderly storage of information in computer memory or pheripheral such as magnetic tape equipment.

debug: (software)

to correct, name of Datageneral program (Debug I, II, III) which can be used to follow program execution.

decimal: (software)

refers to a number base; base 10

digital: (hardware)

pertains to digits or to the representation of data by digits, contrasts to analog.

digital/analog converter (hardware)

interfaces minicomputer with physical equipment decodes digital signal into analog one.

digital computer: (hardware)

a computer in which discrete representation of data is mainly used.

digital voltmeter: (hardware)

instrument which converts a signal from analog or continuous form to digital or discrete form.

double precision: (software)

number representation within two computer words (32 bits for NOVA).

Editor: (software)

program which makes it possible to create, update and change programs, facilities such as line deletion and word corrections are incorporated.

floating point interpreter: (software)

refers to interpreter program used in floating point number manupulations.

floating point numbers: (software)

number using a floating point format, an example would be 10.735 E + 03, any base power may be used.

handler: (software)

I/O operations are called through a subroutine known as handler, involves interrupt servicing (if applicable) and data transfer, (part of operating system).

handshake: (hardware)

communication linking with timing considerations, concerned that data levels on input or output may not be established long enough or soon enough to complete data transfer.

hardware:

defined as the equipment used in a computer system as opposed to computer programs, procedures, rules and associated documentation, contrasts with software.

HP-IB: (hardware)

Hewlett-Packard Interface Bus, see interface bus.

IEEE:

Institute of Electrical and Electronics Engineers

index register: (hardware)

accumulator or register used in calculating effective address.

interface bus: (hardware)

HP-IB, interface system implementing the IEEE 488-1975 standard, based on the Hewlett Packard interface bus system.

interpreter: (software)

translates one source language statement at a time, executes the resulting machine language instruction or instructions, then translates the next statement.

interrupt: (hardware)

when a device needs servicing, it requests an interrupt to the processor (CPU).

I/O: (hardware)

input/output.

listener: (hardware)

device connected to HP-IB capable of receiving instructions or data from bus.

low thermal relay: (hardware)

switching mechanism using Reed relays providing low signal distortion (scanner).

machine language: (software)

a computer language that is used directly by a machine

minicomputer: (hardware).

arbitrarily defined as a digital computer whose minimum configuration (4K memory, teletypewriter) costs under $20,000 and usually employs short computer words (under 18 bits).

module: (software)

a program unit that is discrete and is identifiable with respect to others.

meta-instruction: (software)

communication between operating system and user tasks is implemented through meta-instructions, a machine language subroutine call.

multiplexer: (hardware)

switching mechanism by which several input signals can be connected to one computer entry port.

object tape: (software)

source program tape translated into machine language program tape, equivalent to binary tape.

octal: (software)

refers to a number base; base 8

operating system: (software)

program that controls the operation of the minicomputer system.

page 0: (hardware)

refers to memory, memory is divided into pages, in NOVA minicomputers, page 0 addresses are referable from anywhere in memory.

page 1: (hardware)

refers to memory, refers to total of memory minus page 0 in NOVA minicomputer system, is not necessarily referable from anywhere in memory.

parallel polling: (software)

method of determining which device requested interrupt, device needing servicing input a digit 1 in input word other input 0, location of 1 determined device needing servicing.

peripherals: (hardware)

any equipment distinct from the central processing unit that may provide the system with outside communications or additional facilities.

power fail/auto restart: (hardware)

optional equipment which warns of power failure, restarts program execution upon power restoration.

program loader: (software)

program used to load into the computer's memory binary paper tapes produced by the assembler program.

real time: (hardware)

pertains to the actual time during which a physical process transpires.

real time clock: (hardware)

a device that measures and indicates time, a device that generates periodic signals from which synchonisms may be maintained.

reentrant: (software)

subroutines designed to work properly when they are interrupted and recalled for interrupt servicing, are called reentrant.

relocatable: (software)

memory position independent, a program which is written to run independent of its specific memory location.

RTOS: (software)

real time operating system, operating system supplied by Datageneral of Canada Ltd.

scanner: (hardware)

device (HP-3495A) which serves as a multiplexer for process inputs to the computer.

scheduler: (software)

task scheduler, routine which is used to determine the order of tasks to be executed by the computer, part of the operating system.

serial polling: (software)

method of determining which device is requesting an interrupt, tests each device one at a time to determine which is requesting interrupt.

single precision: (software)

number representation using one computer word (16 bits for NOVA).

source language: (software)

a language from which statements are translated.

source tape: (software)

program tape written in assembler language.

talker: (hardware)

device connected to HP-IB capable of sending instructions or data from bus.

task scheduler: (software)

routine which is used to determine the order of tasks to be executed by the computer, part of the operating system.

teletype: (hardware)

typewriter-like device capable of producing paper-tape, reading paper tape and typing out on paper, interfaced with computer.

user program: (software)

software which is designed and written by the user, contrasts with manufacturer supplied software.

APPENDIX II

SOFTWARE PROGRAM PACKAGE

TABLE OF CONTENT

```
                        ;****************************************;
                        ;                                        ;
                        ;       RTOS DEVICE HANDLER              ;
                        ;                                        ;
                        ;       DIGITAL  TO ANALOG CONVERTER     ;
                        ;                                        ;
                        ;       WORD COUNT >1, IGNORED           ;
                        ;       CONTROL WORD XXXRRRYYYYYYYYYY    ;
                        ;               X=CHANNEL #              ;
                        ;               Y=OUTPUT                 ;
                        ;               R=IGNORED                ;
                        ;                                        ;
                        ;****************************************;

                                .TITL   .D2AC
                                .ENT  DACON
                                .EXTD .STAK
                                .EXTN .DISN
                                .NREL

                        ;ADDRESS OF DEVICE CONTROL BLOCK

00000'000030'                   IOXIN

00001'000000'  DACON:   0                       ;NO INTERRUPT POSSIBLE
00002'000003'  CBADR:   CTBLK

                        ;D/A DEVICE CONTROL BLOCK

00003'000000'  CTBLK:   0
00004'000000'           0
00005'001400'           JMP 0,3
00006'000000'           0
00007'000023'           23
00010'000000'           0
00011'000000'           0
00012'000000'           0
00013'000000'           0
00014'000000'           0
00015'000036'           PTPX
00016'000000'           0
00017'000000'           0
00020'000000'           0
00021'000000'           0
00022'000000'           0
00023'000000'           0

00024'160000'  MASK:    160000          ;MASK FOR CHANNEL
00025'001777'  MASK1:   1777            ;MASK FOR OUTPUT
00026'177777'  OUT:     .DISN           ;.DISN+20
00027'000020'  CONST:   20              ;JMP IOEND TO TERMINATE

                        ;ENTRY POINT FROM .IOX CALL
                        ;C(AC2) = ADDR. OF CREATED QUEUE BLOCK
                        ;C(AC0) = DEVICE CONTROL WORD

00030'034752'  IOXIN:   LDA 3,CBADR     ;LOAD ADDRESS OF CONTROL BLOCK
00031'011407'           ISZ DUBSY,3     ;INDICATE QUEUE BUSY
00032'025400'           LDA 1,DTCBA,3   ;GET FIRST ENTRY
```

```
00033'125004            MOV 1,1,SER     ;UNIT AVAILABLE
00034'002001$           JMP @.STAK      ;NO, GO STACK REQUEST

                ;UNIT IS AVAILABLE - OUTPUT

00035'051400            STA 2,DTCBA,3   ;ENTER QUEUE ADDRESS IN DUCB
00036'024767  PTPX:     LDA 1,MASK1     ;LOAD OUTPUT MASK
00037'111000            MOV 0,2
00040'123400            AND 1,0         ;D/A OUTPUT IN AC0
00041'024763            LDA 1,MASK      ;LOAD CHANNEL MASK
00042'133400            AND 1,2
00043'151120            MOVZL 2,2
00044'151100            MOVL 2,2
00045'151100            MOVL 2,2
00046'151100            MOVL 2,2        ;D/A CHANNEL IN AC2
00047'072023            DOB 2,DACV      ;OUTPUT CHANNEL
00050'061023            DOA 0,DACV      ;OUTPUT VOLTAGE
00051'030731            LDA 2,CBADR     ;LOAD ADDR. OF DUCB
00052'034754            LDA 3,OUT       ;LOAD ADDR. OF .DISN
00053'020754            LDA 0,CONST     ;LOAD CONSTANT OF 20
00054'117000            ADD 0,3         ;ADDR. OF IOEND
00055'102440            SUBO 0,0        ;CLEAR AC0
00056'041007            STA 0,DQBSY,2   ;INDICATE QUEUE AVAILABLE
00057'001400            JMP 0,3         ;JMP TO IOEND TO END OUTPUT
      000007  DQBSY=7                   ;QUEUE BUSY INDICATOR
      000000  DTCBA=0                   ;TCB ADDRESS
                        .END
```

```
                ;*****************************************;
                ;                                         ;
                ;       RTOS DEVICE HANDLER               ;
                ;                                         ;
                ;       POWER FAIL - AUTO RESTART         ;
                ;             ROUTINE                     ;
                ;                                         ;
                ;         POWER SWITCH MUST BE            ;
                ;           IN 'LOCK' POSITION            ;
                ;                                         ;
                ;*****************************************;


                        .TITL PFAIL
                        .ENT .PWR
                        .EXTN .LOX, PFSET
                        .NREL

                ;ENTRY POINT FROM INTERRUPT PROCESSER

00000'060277    .PWR:   INTDS           ;DISABLE INTERRUPTS

00001'040420            STA 0,PFAC0     ;SAVE AC0
00002'044420            STA 1,PFAC1     ;SAVE AC1
00003'050420            STA 2,PFAC2     ;SAVE AC2
00004'054420            STA 3,PFAC3     ;SAVE AC3
00005'175100            MOVL 3,3
00006'054412            STA 3,PFCRY     ;SAVE CARRY
00007'020000            LDA 0,0,0       ;SAVE RETURN ADDRESS
00010'040415            STA 0,PFADR     ;STORE IT
00011'020415            LDA 0,.PFST     ;LOAD RESTART ADDRESS
00012'042415            STA 0,@PFIND    ;STORE PAGE '0', POINTER
00013'020414            LDA 0,PFIND     ;LOAD PAGE '0' ADDRESS
00014'024414            LDA 1,INST      ;LOAD RESTART INSTRUCTION
00015'123000            ADD 1,0         ;ADD TO FORM JMP @PFSET
00016'040000            STA 0,0,0       ;STORE ADDRESS '0'

00017'063077            HALT            ;STOP THE SYSTEM

00020'000000    PFCRY:  0               ;STORAGE FOR CARRY
00021'000000    PFAC0:  0               ;  "      "   AC0
00022'000000    PFAC1:  0               ;  "      "   AC1
00023'000000    PFAC2:  0               ;  "      "   AC2
00024'000000    PFAC3:  0               ;  "      "   AC3
00025'000000    PFADR:  0               ;  "      "   RETURN ADDRESS
00026'000031'   .PFST:  PFRST           ;POINTER FOR RESTART ROUTINE
00027'177777    PFIND:  PFSET           ;POINTER POWER FAIL INDICATOR
00030'002000    INST:   2000            ;INDIRECT (@) INSTRUCTION

                ;RESTART ROUTINE TO RESTORE SYSTEM

00031'020774    PFRST:  LDA 0,PFADR     ;RESTORE RETURN ADDRESS
00032'040000            STA 0,0,0
00033'102400            SUB 0,0         ;CLEAR AC0
00034'101400            INC 0,0         ;INCREMENT AC0
00035'042772            STA 0,@PFIND    ;INDICATE POWER FAILURE
00036'020762            LDA 0,PFCRY     ;  "    CARRY
00037'101200            MOVR 0,0
00040'020761            LDA 0,PFAC0     ;  "    AC0
```

```
00041'024761            LDA 1,@FAC1       ;   "     AC1
00042'030761            LDA 2,@FAC2       ;   "     AC2
00043'034761            LDA 3,@FAC3       ;   "     AC3

00044'060177            INTEN             ;ENABLE INTERRUPTS

00045'002000            JMP @0            ;RETURN

                        .END
```

```
                ;****************************************;
                ;                                        ;
                ;        RTOS DEVICE HANDLER             ;
                ;                                        ;
                ;        HEWLETT-PACKARD ASCII           ;
                ;        BUS INTERFACE                   ;
                ;        IEEE 488-1975                   ;
                ;        HP-IB                           ;
                ;                                        ;
                ;****************************************;

                        .TITL HPBUS
                        .ENT  HPBUS
                        .EXTD .SERV,.STAK
                        .EXTN .CHRI,.DISN
                        .NREL

                ;ADDRESS OF INITIALIZATION ROUTINE

00000'000107'           IOXIN

                ;INTERRUPT SERVICING ROUTINE

00001'006001R HPBUS:    JSR @.SERV      ;REARRANGE PRIORITIES
00002'007777            7777            ;NEW INTERRUPT MASK
00003'000000            0               ;OLD MASK STORAGE
00004'000000            0               ;CARRY
00005'000000            0               ;AC0
00006'000000            0               ;AC1
00007'000000            0               ;AC2
00010'000000            0               ;AC3
00011'000000            0               ;PC

00012'000052' CBADR:    CTBLK           ;ADDR. OF HPIB CONTROL BLOCK
00013'004554            JSR MARC        ;CHECK IF BUS AVAILABLE
00014'065442            DIB 1,AIB       ;INPUT SERVICE MESSAGE
00015'024460            LDA 1,CAST      ;LOAD OUTPUT MESSAGE
00016'004551            JSR MARC        ;CHECK IF BUS AVAILABLE
00017'066042            DOB 1,AIB       ;OUTPUT FOR ACTIVE LISTENING
00020'020462            LDA 0,PMASK     ;LOAD OUTPUT MESSAGE
00021'062042            DOB 0,AIB       ;SET DATA MODE FOR INPUT
00022'020454            LDA 0,MASK      ;LOAD MASK FOR 0
00023'030451            LDA 2,IEND      ;LOAD MASK FOR 12
00024'004543            JSR MARC        ;CHECK IF BUS AVAILABLE
00025'034454            LDA 3,CONST     ;LOAD CONSTANT 20
00026'054445            STA 3,CNTER     ;STORE COUNTER
00027'060277            INTDS
00030'004537 .IN:       JSR MARC        ;CHECK IF BUS AVAILABLE
00031'065542            DIBS 1,AIB      ;INPUT DATA D101-D108
00032'046452            STA 1,@ STORE   ;STORE @POINTER
00033'010451            ISZ STORE       ;INCREMENT POINTER
00034'034437            LDA 3,CNTER     ;LOAD COUNTER
00035'174400            NEG 3,3         ;SUBTRACT 1 FROM AC3
00036'174005            COM 3,3,SNR     ;ZERO RESULT?
00037'000406            JMP .OUT        ;YES, JUMP OUT
00040'054433            STA 3,CNTER     ;STORE LOOP COUNTER
00041'107415            AND# 0,1,SNR    ;ZERO (D101-8) ?
00042'000766            JMP .IN         ;YES, RESTART INPUT
00043'147414            AND# 2,1,SZR    ;LF OR "12 (D101-8) ?
```

```
00044'000764            JMP IN          ;NO,RESTART INPUT
00045'030745  .OUT:     LDA 2,CBADR     ;YES, LOAD ADDR. DUCB
00046'060242            NIOC AIB        ;CLEAR AIB
00047'060177            INTEN
00050'002401            JMP @.+1        ;JMP TO .CHRI
00051'177777            .CHRI           ;TO END INTERRUPT

                ;HP-IB DEVICE CONTROL BLOCK

00052'000000  CTBLK:    0
00053'000000            0
00054'001400            JMP 0,3
00055'000000            0
00056'000042            42
00057'000000            0
00060'000000            0
00061'000000            0
00062'000000
00063'000000            0
00064'000132'           BUS
00065'000000            0
00066'000000            0
00067'000004'           HPBUS+3
00070'000000            0
00071'000000            0
00072'000000  TRIG:     0               ;DVM TRIGGER INDICATOR
00073'000000  CNTER:    0               ;LOOP COUNTER FOR INTERRUPT
00074'000365  IEND:     365             ;MASK FOR INPUT = LF (12)
00075'104000  CAST:     104000          ;OUTPUT FOR ACTIVE LISTENING
00076'000377  MASK:     377             ;MASK FOR INPUT = 0
00077'160000  MASK2:    160000          ;MASK FOR DATA/CONTROL WORD
00100'177777  OUT:      .DISN           ;.DISN+20
00101'000020  CONST:    20              ;JMP IOEND TO TERMINATE
00102'020100  PMASK:    20100           ;DATA MODE FOR INPUT ON INT.
00103'000001  C1:       1               ;CONSTANT 1
00104'000000  STORE:    0               ;INPUT ADDR. POINTER
00105'000000  COUNT:    0               ;DATA COUNTER
00106'000000  POINT:    0               ;OUTPUT ADDR. POINTER

                ;ENTRY POINT FROM .IOX CALL
                ;C(AC2)=ADDR. OF CREATED QUEUE BLOCK
                ;C(AC0)=DEVICE CONTROL WORD (INPUT IND.)

00107'034703  IOXIN:    LDA 3,CBADR     ;GET ADDRESS OF CONTROL BLOCK
00110'011407            ISE DUBSY,3     ;INDICATE QUEUE BUSY
00111'025400            LDA 1,DTCBA,3   ;GET FIRST ENTRY
00112'125004            MOV 1,1,SZR     ;UNIT AVAILABLE
00113'002002$           JMP @.STAK      ;NO, GO STACK REQUEST
00114'051400            STA 2,DTCBA,3   ;ENTER QUEUE ADDR IN DUCB
00115'031005            LDA 2,5,2       ;RESTORE ADDRESS .IOX +1
00116'025003            LDA 1,3,2       ;GET DATA COUNT + TRIG MODE
00117'125100            MOVL 1,1        ;CHECK FOR TRIG MODE
00120'125102            MOVL 1,1,SZC    ;BIT1=1 FOR TRIGGER
00121'044751            STA 1,TRIG      ;SET TRIGGER NOT = 0
00122'125220            MOVZR 1,1       ;RESET CORRECT
00123'125220            MOVZR 1,1       ;DATA COUNTER
00124'044761            STA 1,COUNT     ;SET COUNT = TO # OF OUTPUT WORDS
00125'025002            LDA 1,2,2       ;GET DATA POINTER
00126'044760            STA 1,POINT     ;STORE IN HANDLER
00127'040755            STA 0,STORE     ;DEVICE CONTROL WORD EQUALS
```

```
                                        ;INPUT STORE ADDRESS
00130'060277            INTDS           ;DISABLE INTERRUPTS
00131'020754            LDA 0,COUNT     ;GET DATA COUNTER
00132'101005  BUS:      MOV 0,0,SNR     ;DATA COUNT EQUALS ZERO
00133'000416            JMP END         ;YES, GO TO END:
00134'004433            JSR MARC        ;NO, CHECK IF BUS AVAILABLE
00135'036751            LDA 3,@POINT    ;GET OUTPUT WORD
00136'030741            LDA 2,MASK2     ;LOAD MASK
00137'173414            AND# 3,2,SZR    ;IS IT A CONTROL WORD ?
00140'000403            JMP .+3         ;YES
00141'076142            DOBS 3,AIB      ;NO, DATA WORD (D101-8)
00142'000402            JMP .+2         ;CONTINUE
00143'076042            DOB 3,AIB       ;OUTPUT CONTROL WORD
00144'030737            LDA 2,C1        ;C1=CONSTANT (1)
00145'142400            SUB 2,0         ;DECREASE COUNT BY 1
00146'125400            INC 1,1         ;INCREMENT POINTER
00147'044737            STA 1,POINT     ;STORE NEW POINTER ADDR.
00150'000762            JMP BUS
00151'004416  END:      JSR MARC        ;CHECK IF BUS AVAILABLE
00152'060242            NIOC AIB        ;CLEAR AIB
00153'060177            INTEN           ;ENABLE INTERRUPTS
00154'030716            LDA 2,TRIG      ;GET TRIGGER
00155'151014            MOV# 2,2,SZR    ;TRIGGER EQUALS 0 ?
00156'060142            NIOS AIB        ;NO, TRIGGER AIB
00157'030633            LDA 2,CBADR     ;YES, LOAD ADDR OF CONTROL BLOCK
00160'034720            LDA 3,OUT       ;LOAD ADDR. OF .DISN
00161'020720            LDA 0,CONST     ;LOAD CONSTANT OF 20
00162'117000            ADD 0,3         ;ADDRESS OF IOEND
00163'102440            SUBO 0,0        ;CLEAR AC0
00164'040706            STA 0,TRIG      ;STORE IN TRIG
00165'041007            STA 0,DQBSY,2   ;INDICATE QUEUE AVAILABLE
00166'001400            JMP 0,3         ;JMP TO IOEND TO END OUTPUT

00167'063542  MARC:     SKPBZ AIB       ;UNIT BUSY ?
00170'000777            JMP .-1         ;YES
00171'001400            JMP 0,3         ;NO, RETURN

      000007  DQBSY=7                   ;QUEUE BUSY INDICATOR
      000000  DTCBA=0                   ;TCB ADDRESS
      000042  AIB=42                    ;HP-IB DEVICE UNIT NUMBER

                      .END
```

```
;  DIRECT DIGITAL CONTROL OF A BINARY DISTILLATION
;  COLUMN USING REFLUX AND STEAM FLOWRATES AS
;  CONTROL VARIABLES, DISTILLATE AND BOTTOMS
;  ARE ON ANALOG LEVEL CONTROLLERS.

                .TITL CONTR

                .ENT START, PUTC, GETC, WSA, PFSET
                .EXTN .IOX, .PTY, .QUIT, .BRK, FENT, FINT
                .EXTN .RCV, .XMIT, .WAIT, .FORK
                .EXTN INIT

                .ZREL                   ;PAGE 0 REFERENCE TABLE.

00000-000000  N1:     0                 ;TI REFLUX CONSTANT
00001-013560          6000.             ; IN FL. PT. NOTATION (*100.)
00002-013560  TITOP:  6000.             ; IN INTEGER NOTATION (*100.)

00003-000000  N2:     0                 ;K REFLUX CONSTANT
00004-000001          1                 ; IN FL. PT. NOTATION (*100.)
00005-000001  KTOP:   1                 ; IN INTEGER NOTATION (*100.)

00006-000000  N3:     0                 ;TI STEAM CONSTANT
00007-013560          6000.             ; IN FL. PT. NOTATION (*100.)
00010-013560  TIBOT:  6000.             ; IN INTEGER NOTATION (*100.)

00011-000000  N4:     0                 ;K STEAM CONSTANT
00012-000001          1                 ; IN FL. PT. NOTATION (*100.)
00013-000001  KBOT:   1                 ; IN INTEGER NOTATION (*100.)

00014-000000  N5:     0                 ;LOOP SAMPLING TIME CONSTANT
00015-000000          0                 ; IN FL. PT. NOTATION (*100.)
00016-000000  TCONS:  0                 ; IN INTEGER NOTATION (*100.)

00017-000000  N6:     0                 ;CONSTANT FOR REFLUX, T/2TI
00020-000000  DTCON:  0                 ; IN FL. PT. NOTATION

00021-000000  N7:     0                 ;CONSTANT FOR STEAM, T/2TI
00022-000000  BTCON:  0                 ; IN FL. PT. NOTATION

00023-000000  N8:     0                 ;REFLUX SETPOINT (V*1000.)
00024-000000          0                 ; FL. PT. # (METER REFERENCE)
00025-000000  SPTOP:  0                 ; INTEGER # (VALVE REFERENCE)

00026-000000  N9:     0                 ;STEAM SETPOINT (V*1000.)
00027-000000          0                 ; FL. PT. # (METER REFERENCE)
00030-000000  SPBOT:  0                 ; INTEGER # (VALVE REFERENCE)

00031-000000  N10:    0                 ;ERROR IN REFLUX FLOW
00032-000000  ERTOP:  0                 ; IN FL. PT. NOTATION, NEW ERROR

00033-000000  N11:    0                 ;ERROR FOR STEAM FLOW
00034-000000  ERBOT:  0                 ; IN FL. PT. NOTATION, NEW ERROR

00035-000000  N12:    0                 ;ERROR FOR REFLUX FLOW
00036-000000  N13:    0                 ; IN FL. PT. NOTATION, OLD ERROR
```

```
00037-000000  N14:    0          ;ERROR FOR STEAM FLOW
00040-000000  N15:    0          ; IN FL. PT. NOTATION, OLD ERROR

00041-000000  SINPT:  0          ;STEAM INPUT SIGNAL
00042-000000          0          ; IN FL. PT. NOTATION

00043-000000  RINPT:  0          ;REFLUX INPUT SIGNAL
00044-000000          0          ; IN FL. PT. NOTATION

00045-000000  TI      0          ;TIME LOOP COUNTER

00046-000000  ACONS:  0          ;VALUE OF A IN Y=AX+B
00047-000000          0          ; OUTPUT GAIN, FL. PT. NOTATION

00050-000000  BCONS:  0          ;VALUE OF B IN Y=AX+B
00051-000000          0          ; OUTPUT GAIN, FL. PT. NOTATION

00052-000000  FTENS:  0          ;CONSTANT "10."
00053-000012          10.        ; FLOATING POINT NUMBER

00054-000000  MANAU:  0          ;MANUAL TO AUTO CONTROL INDICATOR

00055-000000  STMAN:  0          ;START-UP INDICATOR
00056-000000  INMAN:  0          ;RETURN ADDRESS OF SUBROUTINE
00057-000060  C4:     60         ;ASCII "0"
00060-000060  CH1:    60         ;CHANNEL # FOR SCANNER
00061-020000  CHA1:   20000      ;CHANNEL #1 FOR D/A OUTPUT
00062-002000  NEGDA:  2000       ;CONSTANT TO NEGATE D/A OUTPUT
00063-000000  DTOUT:  0          ;PRESENT OUTPUT LEVEL REFLUX
00064-000000  BTOUT:  0          ;PRESENT OUTPUT LEVEL STEAM
00065-000000  MBOI:   0          ;DELTA M STEAM
00066-000000          0          ; IN FLOATING POINT NOTATION
00067-000000  MDIST:  0          ;DELTA M REFLUX
00070-000000          0          ; IN FLOATING POINT NOTATION

00071-000000  C100:   0          ;CONSTANT '100.'
00072-000000          0          ; IN FLOATING POINT NOTATION
00073-000144  K100:   100.       ;CONSTANT '100.'


00074-000457' DBIN1:  DBIN       ;POINTER DBIN  SUBROUTINE
00075-000401' BNDE1:  BNDEC      ;POINTER BNDEC SUBROUTINE
00076-001016' DAMEX:  DAEXT      ;POINTER DAEXT SUBROUTINE
00077-002156' MKIRT:  LOPMN      ;POINTER, RETURN K,TI TO MANUAL

00100-102000  SELEC:  102000     ;ENABLE TALKER CONTROL
00101-020300          20300      ;ATN HIGH
00102-000051          51         ;SCANNER LISTEN ADDRESS
00103-020100          20100      ;ATN LOW
00104-000060          60         ;DECADE ADDRESS
00105-000060  CSEL:   60         ;CHANNEL ADDRESS
00106-000015          15         ;EXECUTE
00107-020300          20300      ;ATN HIGH
00110-000077          77         ;UNLISTEN ADDRESS
00111-000000' PUTC:   ERRER
00112-000000' GETC:   ERRER
00113-000042' WSA:    WSA1       ;POINTER FL. PT. INT. WORK AREA
00114-000000  CHEK:   0          ;CONTROL LOOP BUSY INDICATOR
00115-000000  MCTRL:  0          ;MANUAL CONTROL ON INDICATOR
00116-000000  MAKI:   0          ;MANUAL TO KI CHANGE INDICATOR
```

```
00117-001762' MAN1:   MAN              ;POINTER MANUAL CONTROL (TEXT)
00120-001772' MAN2:   MANN             ;POINTER MANUAL CONTROL (NO TEXT)
00121-001166' KTI:    KTI1             ;POINTER REFLUX K,TI CHANGE TASK
00122-002446' STAM:   STEM             ;POINTER STEAM K,TI CHANGE TASK
00123-001503' SP2:    SP1              ;POINTER SETPOINT CHANGE TASK
00124-000102  BINP:   "B               ;ASCII 'B'
00125-000115  MINP:   "M               ;ASCII 'M'
00126-000116  NINP:   "N               ;ASCII 'N'
00127-000122  RINP:   "R               ;ASCII 'R'
00130-000123  SINP:   "S               ;ASCII 'S'
00131-000124  TINP:   "T               ;ASCII 'T'
00132-000131  YES:    "Y               ;ASCII 'Y'

00133-000134- POM:    TEMSP            ;POINTER STORAGE AREA
      000025  TEMSP:  .BLK 25          ;STORAGE AREA
00161-000000  RET1:   0                ;STORAGE OF SUBROUTINE RETURN
00162-000024  C01:    24               ;CONSTANT '24'
00163-000000  MODMA:  0                ;STORAGE TTY INPUT
00164-000000  PFSET:  0                ;INDICATOR POWER FAILURE
00165-177773  MSOUT:  177773           ;MASK OUT ALL DEVICES EXCEPT RTC

00166-054161- CLEAN:  STA 3,RET1       ;STORE RETURN
00167-024162-         LDA 1,C01        ;LOAD COUNTER
00170-034133-         LDA 3,POM        ;LOAD POINTER
00171-102400          SUB 0,0          ;CLEAR AC0
00172-041400  LOP1:   STA 0,0,3        ;STORE AC0 AT POINTER ADDRESS
00173-175400          INC 3,3          ;INCREMENT POINTER
00174-124400          NEG 1,1          ;SUBTRACT 1 FROM COUNTER
00175-124004          COM 1,1,SZR      ;SKIP IF ZERO RESULT
00176-000172-         JMP LOP1         ;NO, CONTINUE
00177-002161-         JMP @RET1        ;YES, RETURN


                      .NREL            ;PAGE 1

00000'177777  ERRER:  .IOX             ;FL. PT. INTERPRETER EXIT
00001'000000          0                ;ERROR MESSAGE OUPUT
00002'100000          100000
00003'000007'         MS2
00004'000100          100
00005'000006'         .+1

00006'063077          HALT             ;STOP OPERATION

00007'051105  MS2:    .TXT ?ER
00010'047522  RO
00011'026122  R,
00012'040440   A
00013'052124  TT
00014'046505  EM
00015'052120  PT
00016'042105  ED
00017'052040   T
00020'020117  O
00021'052517  OU
00022'050124  TP
00023'052125  UT
00024'052440   U
```

```
00025'044523  SI
00026'043516  NG
00027'043040   F
00030'027114  L.
00031'050040   P
00032'027124  T.
00033'044440   I
00034'051516  NS
00035'051124  TR
00036'041525  UC
00037'044524  TI
00040'047117  ON
00041'000123  S?

      000170  WSA1:   .BLK 170          ;FL PT INTERPRETER WORK AREA

                      .EOT              ;END OF REFERENCE TABLE
```

```
;INITIALIZATION PROGRAM, ENTER FROM RTOS

00232'177777  START:  .PRI          ;SET INITIAL PRIORITY
00233'000025          25            ; TO 25

00234'177777          .FORK         ;CREAT PARALLEL TASK
00235'000020          20            ;PRIORITY 20
00236'002415'         BREAK         ;THIS IS THE BREAK (CHANGE) TASK

00237'177777          FINT          ;INITIALIZE FL. PT. INTERPRETER

00240'020474          LDA 0,MCTIM+1 ;LOAD LOOP TIME (*100.)
00241'040015-         STA 0,N5+1    ;STORE IN TABLE (FL. PT. #)
00242'040016-         STA 0,TCONS   ;STORE IN TABLE (INTEGER #)

00243'176400          SUB 3,3       ;CLEAR AC3
00244'054045-         STA 3,TIME    ;STORE IN TIME LOOP COUNTER

00245'020504          LDA 0,A1CONS  ;CONSTANT FOR Y=AX+B
00246'024505          LDA 1,C1CONS  ;NEEDS TO BE CALCULATED
00247'040047-         STA 0,ACONS+1 ;BECAUSE IT IS SO SMALL
00250'044051-         STA 1,BCONS+1 ;A=A1CONS/(C1CONS**2)

00251'020165-         LDA 0,MSOUT   ;LOAD INTERRUPT MASK
00252'062077          DOB 0,CPU     ;MASK OUT ALL DEVICES EXCEPT RTC
00253'177777          FENT          ;ENTER INTERPRETER
00254'060000-         FFLO N1       ;FLOAT TI REFLUX
00255'060003-         FFLO N2       ;FLOAT K REFLUX
00256'060006-         FFLO N3       ;FLOAT TI STEAM
00257'060011-         FFLO N4       ;FLOAT K STEAM
00260'060014-         FFLO N5       ;FLOAT LOOP SAMPLING TIME
00261'060023-         FFLO N8       ;FLOAT REFLUX SETPOINT
00262'060026-         FFLO N9       ;FLOAT STEAM SETPOINT
00263'060035-         FFLO N12      ;FLOAT OLD REFLUX ERROR (=0)
00264'060037-         FFLO N14      ;FLOAT OLD STEAM ERROR (=0)
00265'060052-         FFLO FTENS    ;FLOAT CONSTANT "10."
00266'060046-         FFLO ACONS    ;FLOAT CONSTANTS FOR
00267'060050-         FFLO BCONS    ; Y=AX+B
00270'024046-         FLDA 1,ACONS  ;LOAD CONSTANT (A1CONS)
00271'030050-         FLDA 2,BCONS  ;LOAD CONSTANT (C1CONS)
00272'144200          FDIV 2,1      ;DIVIDE (A1CONS/C1CONS)
00273'144200          FDIV 2,1      ;DIVIDE (A1CONS/C1CONS**2)
00274'044046-         FSTA 1,ACONS  ;STORE VALUE FOR "A"
00275'020000-         FLDA 0,N1     ;LOAD TI REFLUX CONSTANT
00276'024014-         FLDA 1,N5     ;LOAD LOOP SAMPLING TIME
00277'104200          FDIV 0,1      ;DIVIDE, T/TI
00300'124300          FHLV 1,1      ;HALF RESULT, T/2TI
00301'044017-         FSTA 1,N6     ;STORE RESULT
00302'020006-         FLDA 0,N3     ;LOAD TI STEAM CONSTANT
00303'024014-         FLDA 1,N5     ;LOAD LOOP SAMPLING TIME
00304'104200          FDIV 0,1      ;DIVIDE, T/TI
00305'124300          FHLV 1,1      ;HALF RESULT, T/2TI
00306'044021-         FSTA 1,N7     ;STORE RESULT
00307'100000          FEXT          ;EXIT INTERPRETER

00310'020442          LDA 0,B1CONS  ;LOAD CONSTANT "B" (INTEGER)
00311'126400          SUB 1,1       ;CLEAR AC1
00312'044050-         STA 1,BCONS   ;CLEAR FOR FL. PT. #
00313'040051-         STA 0,BCONS+1 ;STORE CONSTANT "B"
```

```
00314'000253'           FENT              ;ENTER INTERPRETER
00315'060050-           FFLO BCONS        ;FLOAT CONSTANT "B"
00316'100000            FEXT              ;EXIT INTERPRETER
00317'102400            SUB 0,0           ;CLEAR AC0
00320'062077            DOB 0,CPU         ;ENABLE INTERRUPT FOR ALL DEVICES

00321'000000'           .IOX              ;OUTPUT MESS  TO DVM
00322'000003            3                 ;(DIGITAL VOLTMETER)
00323'000000            0                 ;TO INITIATE IT
00324'000354'           OUTPUT
00325'000025            25
00326'000327'           .+1

00327'010055-           ISZ STMAN         ;INCREMENT START-UP INDICATOR
00330'006117-           JSR @MAN1         ;JUMP TO MANUAL
00331'102400            SUB 0,0           ;CLEAR AC0
00332'040055-           STA 0,STMAN       ;CLEAR START-UP INDICATOR

00333'177777  MCTIM:    .WAIT             ;WAIT 100.*50. HZ
00334'000062            50.               ; OR 0.5 SECOND

00335'020045-           LDA 0,TIME        ;LOAD LOOP COUNTER
00336'101400            INC 0,0           ;INCREMENT COUNTER
00337'040045-           STA 0,TIME        ;STORE TIME LOOP COUNTER

00340'020114-           LDA 0,CHEK        ;LOAD CONTROL LOOP CHECK
00341'101004            MOV 0,0,SZR       ;BUSY ?
00342'000771            JMP MCTIM         ;YES, CONTINUE
00343'101400            INC 0,0           ;NO, INCREMENT AC0
00344'040114-           STA 0,CHEK        ;STORE AND INDICATE BUSY

00345'000234'           .FORK             ;CREAT PARALLEL CONTROL
00346'000030            30                ;TASK WITH PRIORITY 30
00347'000555'           RSTRT

00350'000763            JMP MCTIM         ;RETURN


00351'022106  A1CONS:   22106             ;CONSTANT 9286, BASE 10
00352'000144  B1CONS:   144               ;CONSTANT 144 ="B"
00353'001750  C1CONS:   1750              ;CONSTANT (10**-3) FOR CALC. OF A

00354'040000  OUTPUT:   40000             ;INITIATE BUS CLEAR
00355'102000            102000            ;ENABLE TALKER CONTROL
00356'020300            20300             ;ATN HIGH
00357'000066            66                ;DVM LISTEN ADDRESS
00360'020100            20100             ;ATN LOW
00361'000106            106               ;SET DC VOLTS
00362'000061            61
00363'000122            122               ;SET RANGE = 1 VOLT
00364'000062            62
00365'000124            124               ;SET HOLD/MANUAL
00366'000063            63
00367'000115            115               ;SET MATH OFF
00370'000063            63
00371'000101            101               ;SET AUTO-CAL OFF
00372'000060            60
00373'000110            110               ;SET HIGH RESOLUTION OFF
00374'000060            60
```

```
00375'000104        104             ;SET DATA READY ON, RQS
00376'000061        61
00377'020300        20300           ;ATN HIGH
00400'000077        77              ;UNLISTEN ADDRESS

                    .EOT            ;END OF MAIN PROGRAM TAPE
```

```
                        ;SUBROUTINE TO CONVERT A SINGLE PRECISION
                        ;BINARY NUMBER TO AN ASCII (DECIMAL)
                        ;CHARACTER STRING AND OUTPUT IT ON
                        ;THE TELETYPE

                        ;       AC1 HAS BINARY NUMBER TO BE OUTPUT

00401'054445  BNDEC:    STA 3,SAVE          ;SAVE ROUTINE ADDRESS
00402'102440            SUBO 0,0            ;CLEAR AC0
00403'040444            STA 0,NUMB          ;CLEAR COUNTER
00404'034444            LDA 3,PNTE1         ;LOAD POINTER ADDRESS
00405'054444            STA 3,PNTER         ;STORE POINTER ADDRESS
00406'034436            LDA 3,INST          ;SET UP LDA COMMAND
00407'054401            STA 3,.+1
00410'000000  LOOP:     0                   ;AC2=POWER OF 10
00411'020434            LDA 0,C60           ;AC0=ASCII ZERO
00412'146443            SUBO 2,1,SNC        ;STILL PLUS IF NO CARRY
00413'101401            INC 0,0,SKP         ;INC ASCII CHARACTER
00414'147001            ADD 2,1,SKP         ;TOO MUCH ADD BACK
00415'000775            JMP .-3
00416'034433            LDA 3,PNTER         ;LOAD POINTER
00417'175400            INC 3,3             ;INC POINTER
00420'054431            STA 3,PNTER         ;STORE POINTER
00421'041400            STA 0,0,3           ;STORE RESULT
00422'010425            ISZ NUMB            ;INC COUNTER
00423'010765            ISZ LOOP            ;INC LDA COMMAND
00424'151203            MOVR 2,2,SNC        ;LAST DIGIT?
00425'000763            JMP LOOP            ;NO,CONTINUE
00426'020421            LDA 0,NUMB          ;LOAD COUNTER
00427'040405            STA 0,.+5           ;YES,OUTPUT
00430'000321'           .IOX                ;OUTPUT ASCII CHARACTER STRING
00431'000000            0
00432'130000            130000
00433'000452'           PNTE1+2
00434'000000            0
00435'000436'           .+1

00436'002410            JMP @SAVE           ;RETURN

      000012            .RDX 10             ;CHANGE RADIX TO 10
00437'023420  TENS:     10000
00440'001750            1000
00441'000144            100
00442'000012            10
00443'000001            1
      000010            .RDX 8              ;CHANGE BACK TO 8
00444'030427  INST:     LDA 2,.+TENS-LOOP
00445'000060  C60:      60                  ;ASCII '0', CONSTANT "60"
00446'000000  SAVE:     0                   ;SUBROUTINE RETURN ADDRESS
00447'000000  NUMB:     0                   ;COUNTER
00450'000451' PNTE1:    PNTER               ;POINTER TO 'PNTER'
00451'000451' PNTER:    PNTER               ;POINTER FOR NEXT RESULT
00452'000000            0                   ;ASCII CHARACTER STRING
00453'000000            0                   ;STORAGE AREA
00454'000000            0
00455'000000            0
00456'000000            0

                        .EOT                ;END OF BNDEC TAPE
```

```
                        ;SUBROUTINE TO CONVERT AN ASCII CHARACTER
                        ;STRING TO A SINGLE PRECISION BINARY NUMBER
                        ;       INPUT: AC1 HAS DATA POINTER
                        ;       OUTPUT:AC1 HAS BINARY INTEGER NUMBER

                        ;ACCUMULATORS DESTROYED:
                        ;       AC3 (SUBROUTINE CALL)
                        ;       AC1 (DATA POINTER).

                        ;TERMINATION:
                        ;       BY INPUT OF CHARACTER NOT IN 0-9 RANGE

00457'054441  DBIN:     STA 3,.EC03     ;SAVE RETURN
00460'050437            STA 2,.EC02     ;SAVE AC2
00461'044440            STA 1,.EC01     ;SAVE POINTER
00462'040434            STA 0,.EC00     ;SAVE AC0
00463'102400            SUB 0,0         ;CLEAR AC0
00464'040436            STA 0,.EC11     ;CLEAR SUM
00465'034434  .EC96:    LDA 3,.EC01     ;LOAD AC3 WITH POINTER
00466'021400            LDA 0,0,3       ;LOAD AC0 WITH INPUT
00467'024434            LDA 1,.EC22     ;ASCII "0"
00470'030434            LDA 2,.EC23     ;ASCII "9"
00471'034434            LDA 3,MASK      ;LOAD 6 BIT MASK
00472'163400            AND 3,0         ;ISOLATE ASCII #
00473'142033            ADCZ# 2,0,SNC   ;SKIP IF >9
00474'106032            ADCZ# 0,1,SZC   ;SKIP IF >=0
00475'000407            JMP .EC95       ;NOT A DIGIT THEREFORE A
                                        ;BREAK CHARACTER
00476'122400            SUB 1,0         ;REDUCE DIGIT TO 0-9
                                        ;BINARY RANGE
00477'024423            LDA 1,.EC11     ;SUM WORD
00500'004410            JSR .EC50       ;MULTIPLY BY 10 & ADD
00501'044421            STA 1,.EC11     ;SAVE SUM
00502'010417            ISZ .EC01       ;INCREMENT POINTER
00503'000762            JMP .EC96       ;GET NEXT CHARACTER
00504'024416  .EC95:    LDA 1,.EC11     ;RESULT IN AC1
00505'020411            LDA 0,.EC00     ;RESTORE AC0
00506'030411            LDA 2,.EC02     ;RESTORE AC2
00507'002411            JMP @.EC03      ;RETURN
00510'131120  .EC50:    MOVZL 1,2       ;N*2
00511'151120            MOVZL 2,2       ;N*4
00512'147000            ADD 2,1         ;N*5
00513'125120            MOVZL 1,1       ;N*5*2=N*10
00514'107000            ADD 0,1         ;ADD AC0
00515'001400            JMP 0,3         ;RETURN

00516'000000  .EC00:    0               ;AC0 STORAGE
00517'000000  .EC02:    0               ;AC2 STORAGE
00520'000000  .EC03:    0               ;RETURN
00521'000000  .EC01:    0               ;POINTER ADDRESS
00522'000000  .EC11:    0               ;SUM
00523'000060  .EC22:    "0              ;ASCII 0
00524'000071  .EC23:    "9              ;ASCII 9
00525'000077  MASK:     77              ;MASK WORD


                        .EOT            ;END OF DBIN TAPE
```

```
                        ;CONTROL LOOP TASK
                        ; INPUT: CH0 = REFLUX FLOW SIGNAL
                        ;        CH1 = STEAM FLOW SIGNAL
                        ;        CH2 = FEED FLOW SIGNAL


00526'000004  FOUR:   4                     ;CONSTANT "4"
00527'000530' IN2:    IN1                   ;DVM INPUT POINTER
      000024  IN1:    .BLK 20.              ;DVM INPUT STORAGE
00554'001013' QUIT:   MQUIT                 ;POINTER FOR END OF TASK

00555'020115- RSTRT:  LDA 0,MCTRL           ;LOAD MANUAL INDICATOR
00556'101004          MOV 0,0,SZR           ;SKIP IF 0
00557'002775          JMP @ QUIT            ;ON MANUAL, JMP TO END

00560'020060-         LDA 0,CH1             ;LOAD CHANNEL
00561'040105-         STA 0,CSEL            ;STORE CHANNEL IN SCANNER OUTPUT

00562'000430'         .IOX                  ;SELECT SCANNER CHANNEL
00563'000003          3
00564'000000          0
00565'000100-         SELEC
00566'000011          11
00567'000570'         .+1

00570'000562'         .IOX                  ;TRIGGER DVM
00571'000003          3
00572'000530'         IN1
00573'001027'         OUT1
00574'040012          40012
00575'000576'         .+1

00576'024731          LDA 1,IN2             ;LOAD AC1 WITH POINTER
00577'020727          LDA 0,FOUR            ;LOAD CONSTANT "4"
00600'107000          ADD 0,1               ;ADD, = POINTER ADDRESS +4
00601'006074-         JSR @DBIN1            ;JSR ASCII=DECIMAL CONVERSION
00602'176400          SUB 3,3               ;CLEAR AC3
00603'020060-         LDA 0,CH1             ;LOAD CHANNEL
00604'030057-         LDA 2,C4              ;GET "60", ASCII 0
00605'142405          SUB 2,0,SNR           ;GET CHANNEL #, CHANNEL 0?
00606'000407          JMP CHAN0             ;YES
00607'101205          MOVR 0,0,SNR          ;CHANNEL 1?
00610'000410          JMP CHAN1             ;YES
00611'101205          MOVR 0,0,SNR          ;CHANNEL 2?
00612'000402          JMP CHAN2             ;YES
00613'000407          JMP EX                ;ERROR, CONTINUE

00614'000406  CHAN2:  JMP EX                ;CHAN. 2 = FEED, NOT YET NEEDED

00615'044044- CHAN0:  STA 1,RINPT+1         ;CHAN. 0 = REFLUX, STORE INPUT
00616'054043-         STA 3,RINPT           ;CLEAR FOR FL. PT. #

00617'000403          JMP EX                ;CONTINUE

00620'044042- CHAN1:  STA 1,SINPT+1         ;CHAN. 1 = STEAM, STORE INPUT
00621'054041-         STA 3,SINPT           ;CLEAR FOR FL. PT. #

00622'020060- EX:     LDA 0,CH1             ;LOAD CHANNEL #
00623'024057-         LDA 1,C4              ;LOAD '60', ASCII "0"
00624'122400          SUB 1,0               ;REDUCE TO CHANNEL #
```

```
00625'101235            MOVŹR# 0,0,SNR  ;AC0 LESS THAN 2?
00626'000404            JMP CONTA       ;YES, CONTINUE
00627'020057-           LDA 0,C4        ;NO, RESET
00630'040060-           STA 0,CH1       ;STORE CHANNEL #0 IN CH1
00631'000406            JMP CONTB       ;CONTINUE

00632'101400  CONTA:    INC 0,0         ;INCREMENT CHANNEL #
00633'024057-           LDA 1,C4        ;LOAD '60', ASCII "0"
00634'123000            ADD 1,0         ;ADD TO CHANNEL #
00635'040060-           STA 0,CH1       ;STORE CHANNEL #
00636'000717            JMP RSTRT       ;RESTART LOOP

00637'020165- CONTB:    LDA 0,MSOUT     ;LOAD INTERRUPT MASK
00640'062077            DOB 0,CPU       ;MASK OUT ALL DEVICES EXCEPT RTC
00641'000314'           FENT            ;ENTER INTERPRETER
00642'060043-           FFLO RINPT      ;FLOAT REFLUX INPUT SIGNAL
00643'060041-           FFLO SINPT      ;FLOAT STEAM INPUT SIGNAL
00644'020023-           FLDA 0,N8       ;LOAD REFLUX SETPOINT
00645'024043-           FLDA 1,RINPT    ;LOAD REFLUX INPUT SIGNAL
00646'122400            FSUB 1,0        ;SUBTRACT, = NEW REFLUX ERROR
00647'040031-           FSTA 0,N10      ;STORE NEW REFLUX ERROR
00650'024035-           FLDA 1,N12      ;LOAD OLD ERROR IN REFLUX
00651'122400            FSUB 1,0        ;E(NEW)-E(OLD)= *E= DELTA E
00652'030031-           FLDA 2,N10      ;LOAD NEW ERROR IN REFLUX
00653'133000            FADD 1,2        ;E(NEW)+E(OLD)
00654'024017-           FLDA 1,N6       ;LOAD T/2TI, REFLUX CONSTANT
00655'130100            FMPY 1,2        ;T/2TI(E(NEW)+E(OLD))
00656'143000            FADD 2,0        ;*E+T/2TI(E(NEW)+E(OLD))
00657'024003-           FLDA 1,N2       ;LOAD K REFLUX CONSTANT
00660'120100            FMPY 1,0        ;K(*E+T/2TI(E(NEW)+E(OLD)))
00661'024046-           FLDA 1,ACONS    ;A CONSTANT IN Y=AX+B
00662'120100            FMPY 1,0        ; AX (B CANCEL SINCE DELTA)
00663'030052-           FLDA 2,FTENS    ;LOAD CONSTANT "10."
00664'140200            FDIV 2,0        ; *M (Y)=FAC0/10.
00665'040067-           FSTA 0,MDIST    ;STORE RESULT *M (Y)
00666'074067-           FFIX MDIST      ;CONV. FL PT #=D. PRES. INTEGER
00667'020031-           FLDA 0,N10      ;LOAD NEW ERROR
00670'040035-           FSTA 0,N12      ;STORE IN OLD ERROR FOR NEXT LOOP

00671'020026-           FLDA 0,N9       ;LOAD STEAM SETPOINT
00672'024041-           FLDA 1,SINPT    ;LOAD STEAM INPUT SIGNAL
00673'122400            FSUB 1,0        ;SUBTRACT, = NEW STEAM ERROR
00674'040033-           FSTA 0,N11      ;STORE NEW STEAM ERROR
00675'024037-           FLDA 1,N14      ;LOAD OLD STEAM ERROR
00676'122400            FSUB 1,0        ;E(NEW)-E(OLD)= *E=  DELTA E
00677'030033-           FLDA 2,N11      ;LOAD NEW ERROR IN STEAM
00700'133000            FADD 1,2        ;E(NEW)+E(OLD)
00701'024021-           FLDA 1,N7       ;LOAD T/2TI STEAM CONSTANT
00702'130100            FMPY 1,2        ;T/2TI(E(NEW)+E(OLD))
00703'143000            FADD 2,0        ;*E+T/2TI(E(NEW)+E(OLD))
00704'024011-           FLDA 1,N4       ;LOAD K STEAM CONSTANT
00705'120100            FMPY 1,0        ;K(*E+T/2TI(E(NEW)+E(OLD)))
00706'024046-           FLDA 1,ACONS    ;A CONSTANT IN Y*AX+B
00707'120100            FMPY 1,0        ; AX (B CANCEL SINCE DELTA)
00710'030052-           FLDA 2,FTENS    ;LOAD CONSTANT "10."
00711'140200            FDIV 2,0        ; *M (Y)=FAC0/10.
00712'040065-           FSTA 0,MBOT     ;STORE RESULT *M (Y)
00713'074065-           FFIX MBOT       ;CONV. FL PT #=D. PRES. INTEGER
00714'020033-           FLDA 0,N11      ;LOAD NEW ERROR
00715'040037-           FSTA 0,N14      ;STORE IN OLD ERROR FOR NEXT LOOP
```

```
00716'100000            FEXT            ;EXIT INTERPRETER
00717'102400            SUB 0,0         ;CLEAR AC0
00720'062077            DOB 0,CPU       ;ENABLE INTERRUPT FOR ALL DEVICES

00721'020054-           LDA 0,MANAU     ;LOAD MANUAL TO AUTO INDICATOR
00722'101005            MOV 0,0,SNR     ;SKIP IF IN TRANSITION LOOP
00723'000432            JMP NOTTR       ;NOT IN TRANSITION, CONTINUE

00724'020165-           LDA 0,MSK1      ;LOAD INTERRUPT MASK
00725'062077            DOB 0,CPU       ;MASK OUT ALL DEVICES EXCEPT RTC
00726'102400            SUB 0,0         ;CLEAR AC0
00727'040037-           STA 0,N14       ;CLEAR OLD STEAM FLOW ERROR
00730'040040-           STA 0,N15       ; FLOATING POINT NUMBER
00731'040035-           STA 0,N12       ;CLEAR OLD REFLUX FLOW ERROR
00732'040036-           STA 0,N13       ; FLOATING POINT NUMBER
00733'000641'           FENT            ;ENTER FL. PT. INTERPRETER
00734'060035-           FFLO N12        ;FLOAT OLD REFLUX ERROR (=0)
00735'020043-           FLDA 0,RINPT    ;LOAD REFLUX INPUT SIGNAL
00736'040023-           FSTA 0,N8       ;STORE  REFLUX SETPT (METER REF.)

00737'060037-           FFLO N14        ;FLOAT OLD STEAM ERROR (=0)
00740'020041-           FLDA 0,SINPT    ;LOAD STEAM INPUT SIGNAL
00741'040026-           FSTA 0,N9       ;STORE  STEAM SETPT (METER REF.)
00742'074041-           FFIX SINPT      ;FIX STEAM INPUT SIGNAL (SETPT)
00743'074043-           FFIX RINPT      ;FIX REFLUX INPUT SIGNAL (SETPT)
00744'100000            FEXT            ;EXIT INTERPRETER

00745'020042-           LDA 0,SINPT+1   ;LOAD INTEGER SETPOINT
00746'024044-           LDA 1,RINPT+1   ;LOAD INTEGER REFLUX SETPOINT
00747'040030-           STA 0,SPBOT     ;STORE AS INTEGER VALUE
00750'044025-           STA 1,SPTOP     ;STORE AS INTEGER VALUE
00751'102400            SUB 0,0         ;CLEAR AC0
00752'040054-           STA 0,MANAU     ;CLEAR MANUAL/AUTO INDICATOR
00753'062077            DOB 0,CPU       ;ENABLE INTERRUPT FOR ALL DEVICES

00754'000437            JMP MQUIT       ;RESTART LOOP

00755'020070- NOTTR:    LDA 0,MDIST+1   ;LOAD DELTA M REFLUX
00756'024063-           LDA 1,DTOUT     ;LOAD PRESENT OUTPUT LEVEL (M)
00757'123000            ADD 1,0         ;ADD CHANGE, M +DELTA M
00760'004436            JSR DAEXT       ;JSR TO CHECK D/A EXTREMES
00761'040063-           STA 0,DTOUT     ;M +DELTA M = NEW M
00762'024062-           LDA 1,NEGDA     ;REVERSE D/A OUTPUT
00763'106400            SUB 0,1         ;CHANGE "+" TO "-"
00764'044403            STA 1,.+3       ;D/A OUTPUT REFLUX

00765'000570'           .IOX            ;D/A OUTPUT REFLUX
00766'000004            4
00767'000000            0
00770'000001            1
00771'000001            1
00772'000773'           .+1

00773'020066-           LDA 0,MBOT+1    ;LOAD DELTA M STEAM
00774'024064-           LDA 1,BTOUT     ;LOAD PRESENT OUTPUT LEVEL (M)
00775'123000            ADD 1,0         ;ADD CHANGE, M +DELTA M
00776'004420            JSR DAEXT       ;JSR TO CHECK D/A EXTREMES
00777'040064-           STA 0,BTOUT     ;M +DELTA M = NEW M
01000'030062-           LDA 2,NEGDA     ;REVERSE D/A OUTPUT
01001'112400            SUB 0,2         ;CHANGE "+" TO "-"
```

```
01002'024061-           LDA 1,CHA1      ;CHANNEL #1 MODE
01003'133000            ADD 1,2         ;ADD TO OUTPUT
01004'050403            STA 2,.+3       ;D/A OUTPUT STEAM

01005'000765'           .IOX            ;D/A OUTPUT STEAM
01006'000004            4
01007'000000            0
01010'000001            1
01011'000001            1
01012'001013'           .+1

01013'102400  MQUIT:    SUB 0,0         ;CLEAR AC0
01014'040114-           STA 0,CHEK      ;STORE AND INDICATE FREE

01015'177777            .QUIT


01016'024410  DAEXT:    LDA 1,DAMIN     ;D/A MINIMUM OUTPUT
01017'122512            SUBL# 1,0,SEC   ;BELOW MINIMUM
01020'020406            LDA 0,DAMIN     ;YES, OUTPUT MINIMUM
01021'024404            LDA 1,DAMAX     ;D/A MAXIMUM OUTPUT
01022'122513            SUBL# 1,0,SNC   ;ABOVE MAXIMUM
01023'020402            LDA 0,          ;YES, OUTPUT MAXIMUM
01024'001400            JMP 0,3         ;RETURN

01025'000651  DAMAX:    651             ;D/A MAXIMUM OUTPUT
01026'000144  DAMIN:    144             ;D/A MINIMUM OUTPUT

01027'102000  OUT1:     102000          ;TO TRIGGER DVM
01030'020300            20300
01031'000066            66
01032'020100            20100
01033'000124            124
01034'000063            63
01035'020300            20300
01036'000126            126
01037'020320            20320
01040'101000            101000


                        .EOT            ;END OF CONTROL LOOP TASK
```

```
                ;K,TI CHANGES IN REFLUX SETTINGS

01041'006412  MESS1:   .TXT ?<12><15>
01042'050012  <12>P
01043'047522  RO
01044'047520  PO
01045'052122  RT
01046'047511  IO
01047'040516  NA
01050'020114  L
01051'040507  GA
01052'047111  IN
01053'024040   (
01054'024513  K)
01055'043040   F
01056'051117  OR
01057'051040   R
01060'043105  EF
01061'052514  LU
01062'020130  X
01063'047503  CO
01064'052116  NT
01065'047522  RO
01066'046114  LL
01067'051105  ER
01070'024040   (
01071'030452  *1
01072'030060  00
01073'024456  .)
01074'020040  <40><40>
01075'000040  <40>?

01076'006412  MESS2:   .TXT ?<12><15>
01077'044412  <12>I
01100'052116  NT
01101'043505  EG
01102'040522  RA
01103'020114  L
01104'044524  TI
01105'042515  ME
01106'043040   F
01107'051117  OR
01110'051040   R
01111'043105  EF
01112'052514  LU
01113'020130  X
01114'047503  CO
01115'052116  NT
01116'047522  RO
01117'046114  LL
01120'051105  ER
01121'024040   (
01122'030452  *1
01123'030060  00
01124'024456  .)
01125'020040  <40><40>
01126'000040  <40>?

01127'006412  MESS3:   .TXT ?<12><15>
01130'044412  <12>I
```

```
01131'050116  NP
01132'052125  UT
01133'047040   N
01134'053505  EW
01135'050040   P
01136'047522  RO
01137'047520  PO
01140'052122  RT
01141'047511  IO
01142'040516  NA
01143'020114  L
01144'040507  GA
01145'047111  IN
01146'024040   (
01147'030452  *1
01150'030060  00
01151'024456  .)
01152'005015  <15><12>
01153'052040   T
01154'051105  ER
01155'044515  MI
01156'040516  NA
01157'042524  TE
01160'053440   W
01161'052111  IT
01162'020110  H
01163'051103  CR
01164'020040  <40><40>
01165'000040  <40>?


01166'001005' KTI1:   .IOX            ;OUTPUT MESSAGE 1
01167'000000          0
01170'100000          100000
01171'002102"         MESS1*2
01172'000100          100
01173'001174'         .+1

01174'024005-         LDA 1,KTOP      ;GET K REFLUX VALUE
01175'006075-         JSR @BNDE1      ;OUTPUT VALUE

01176'001166'         .IOX            ;OUTPUT MESSAGE 2
01177'000000          0
01200'100000          100000
01201'002174"         MESS2*2
01202'000100          100
01203'001204'         .+1

01204'024002-         LDA 1,TITOP     ;GET TI REFLUX VALUE
01205'006075-         JSR @BNDE1      ;OUTPUT VALUE

01206'001176'         .IOX            ;OUTPUT MESSAGE 3
01207'000000          0
01210'100000          100000
01211'002256"         MESS3*2
01212'000100          100
01213'001214'         .+1

01214'001206'         .IOX            ;INPUT NEW VALUE
01215'000000          0
```

```
01216'020000          20000
01217'000134-         TEMSP
01220'000020          20
01221'001222'         .+1

01222'024133-         LDA 1,POM          ;LOAD POINTER
01223'006074-         JSR @DBIN1         ;CONVERT ASCII=BINARY
01224'044005-         STA 1,KTOP         ;STORE NEW K (INTEGER #)

01225'004166-         JSR CLEAN          ;CLEAR TABLE TEMSP

01226'001214'         .IOX               ;OUTPUT MESSAGE 4
01227'000000          0
01230'100000          100000
01231'002646"         MESS4*2
01232'000100          100
01233'001234'         .+1

01234'001226'         .IOX               ;INPUT NEW VALUE
01235'000000          0
01236'020000          20000
01237'000134-         TEMSP
01240'000020          20
01241'001242'         .+1

01242'024133-         LDA 1,POM          ;LOAD POINTER
01243'006074-         JSR @DBIN1         ;CONVERT ASCII=BINARY
01244'044002-         STA 1,TITOP        ;STORE NEW TI (INTEGER #)
01245'020165-         LDA 0,MSOUT        ;LOAD INTERRUPT MASK
01246'062077          DOB 0,CPU          ;MASK OUT ALL DEVICES EXCEPT RTC
01247'044001-         STA 1,N1+1         ;STORE NEW TI (FL. PT. #)
01250'024073-         LDA 1,K100         ;LOAD CONSTANT 100
01251'044072-         STA 1,C100+1       ;STORE CONSTANT
01252'024005-         LDA 1,KTOP         ;LOAD NEW K REFLUX (INTEGER)
01253'044004-         STA 1,N2+1         ;STORE NEW K (FL. PT. #)
01254'102400          SUB 0,0            ;CLEAR AC0
01255'040003-         STA 0,N2           ;CLEAR OLD FL PT K VALUE
01256'040000-         STA 0,N1           ;CLEAR OLD FL PT TI VALUE
01257'040071-         STA 0,C100         ;CLEAR OLD FL PT CONSTANT

01260'004166-         JSR CLEAN          ;CLEAR TABLE TEMSP

01261'000733'         FENT               ;ENTER INTERPRETER
01262'060000-         FFLO N1            ;FLOAT TI REFLUX
01263'060003-         FFLO N2            ;FLOAT K REFLUX
01264'020000-         FLDA 0,N1          ;LOAD TI CONSTANT REFLUX
01265'024014-         FLDA 1,N5          ;LOAD LOOP TIME CONSTANT
01266'104200          FDIV 0,1           ;DIVIDE, T/TI
01267'124300          FHLV 1,1           ;HALF RESULT
01270'044017-         FSTA 1,N6          ;STORE, T/2TI
01271'060071-         FFLO C100          ;FLOAT CONSTANT 100.
01272'020071-         FLDA 0,C100        ;LOAD CONSTANT
01273'024003-         FLDA 1,N2          ;LOAD K REFLUX
01274'104200          FDIV 0,1           ;DIVIDE BY 100.
01275'044003-         FSTA 1,N2          ;STORE CORRECT K REFLUX
01276'100000          FEXT               ;EXIT INTERPRETER
01277'102400          SUB 0,0            ;CLEAR AC0
01300'062077          DOB 0,CPU          ;ENABLE INTERRUPT FOR ALL DEVICES

01301'001234'         .IOX               ;OUTPUT CR, LF
```

```
01302'000000            0
01303'110000            110000
01304'002646'           MESS4*2
01305'000003            3
01306'001307'           .+1

01307'020116-           LDA 0,MAKI      ;LOAD MANUAL TO KTI INDICATOR
01310'101005            MOV 0,0,SNR     ;SKIP IF NOT = 0
01311'000402            JMP .+2         ;EXIT
01312'002077-           JMP @MKIRT      ;RETURN TO MANUAL

01313'000345'           .FORK           ;CREAT NEW TASK
01314'000015            15              ;TO REINITIALIZE
01315'001317'           KI4             ;PREVIOUS TERMINATING TASK

01316'001015'           .QUIT

01317'001313' KI4:      .FORK           ;CREAT NEW TASK
01320'000020            20              ;SAME PRIORITY AND
01321'002415'           BREAK           ;SAME TASK AS PREVIOUS

01322'001316'           .QUIT

01323'006412  MESS4:  .TXT ?<12><15>
01324'044412  <12>I
01325'050116  NP
01326'052125  UT
01327'044440   I
01330'052116  NT
01331'043505  EG
01332'040522  RA
01333'020114  L
01334'044524  TI
01335'042515  ME
01336'041440   C
01337'047117  ON
01340'052123  ST
01341'047101  AN
01342'020124  T
01343'025050  (*
01344'030061  10
01345'027060  0.
01346'005051  )<12>
01347'020015  <15><40>
01350'052040  <40>T
01351'051105  ER
01352'044515  MI
01353'040516  NA
01354'042524  TE
01355'053440   W
01356'052111  IT
01357'020110  H
01360'051103  CR
01361'020040  <40><40>
01362'000040  <40>?

                        .EOT            ;END OF REFLUX K-TI TAPE
```

```
                ;SETPOINT CHANGES FOR REFLUX & STEAM CONTROLLERS

01363'006412  MESS9:   .TXT ?<12><15>
01364'051412  <12>S
01365'052105  ET
01366'047520  PO
01367'047111  IN
01370'020124  T
01371'043117  OF
01372'051040    R
01373'043105  EF
01374'052514  LU
01375'020130  X
01376'047503  CO
01377'052116  NT
01400'047522  RO
01401'046114  LL
01402'051105  ER
01403'024040    (
01404'053115  MV
01405'030452  *1
01406'030060  00
01407'030060  00
01410'030060  00
01411'024456  .)
01412'020040  <40><40>
01413'000040  <40>?

01414'006412  MES10:   .TXT ?<12><15>
01415'044412  <12>I
01416'050116  NP
01417'052125  UT
01420'047040    N
01421'053505  EW
01422'051440    S
01423'052105  ET
01424'047520  PO
01425'047111  IN
01426'020124  T
01427'042450  (E
01430'042116  ND
01431'053440    W
01432'052111  IT
01433'020110  H
01434'051103  CR
01435'005051  )<12>
01436'020015  <15><40>
01437'046440  <40>M
01440'047111  IN
01441'030075  =0
01442'046454  .M
01443'054101  AX
01444'031475  =3
01445'030065  50
01446'030060  00
01447'020056  .<40>
01450'020040  <  )><40>
01451'  00  ?
```

```
01452'006412  MES11:   .TXT ?<12><1 > o
01453'051412  <12>S
01454'052105  ET
01455'047520  PO
01456'047111  IN
01457'  124   T
01460'043117  OF
01461'051440   S
01462'042524  TE
01463'046501  AM
01464'041440   C
01465'047117  ON
01466'051124  TR
01467'046117  OL
01470'042514  LE
01471'020122  R
01472'046450  (M
01473'025126  V*
01474'030061  10
01475'030060  00
01476'030060  00
01477'027060  0.
01500'020051  )<40>
01501'020040  <40><40>
01502'000000  ?

01503'001301' SP1:     .IOX              ;OUTPUT MESSAGE 9
01504'000000           0
01505'100000           100000
01506'002746"          MESS9*2
01507'000100           100
01510'001511'          .+1

01511'024025-          LDA 1,SPTOP       ;LOAD REFLUX SETPOINT
01512'006075-          JSR @BNDE1        ;OUTPUT VALUE

01513'001503'          .IOX              ;OUTPUT MESSAGE 10
01514'000000           0
01515'100000           100000
01516'003030"          MES10*2
01517'000100           100
01520'001521'          .+1

01521'001513'          .IOX              ;INPUT NEW VALUE
01522'000000           0
01523'020000           20000
01524'000134-          TEMSP
01525'000020           20
01526'001527'          .+1

01527'024133-          LDA 1,POM         ;LOAD POINTER
01530'006074-          JSR @DBIN1        ;CONVERT ASCII=BINARY
01531'044025-          STA 1,SPTOP       ;STORE NEW REFLUX SETPOINT

01532'004166-          JSR CLEAN         ;CLEAR TABLE TEMSP

01533'001521'          .IOX              ;OUTPUT MESSAGE 11
01534'000000           0
01535'100000           100000
01536'003124"          MES11*2
```

```
01537'000100            100
01540'001541'           .+1

01541'024030-           LDA 1,SPBOT     ;LOAD STEAM SETPT.
01542'006075-           JSR @BNDE1      ;OUTPUT VALUE

01543'001533'           .IOX            ;OUTPUT MESSAGE 10
01544'000000            0
01545'100000            100000
01546'003030"           MES10*2
01547'000100            100
01550'001551'           .+1

01551'001543'           .IOX            ;INPUT NEW VALUE
01552'000000            0
01553'020000            20000
01554'000134-           TEMSP
01555'000020            20
01556'001557'           .+1

01557'024133-           LDA 1,POM       ;LOAD POINTER
01560'006074-           JSR @DBIN1      ;CONVERT ASCII=BINARY
01561'044030-           STA 1,SPBOT     ;STORE NEW STEAM SETPT.
01562'020165-           LDA 0,MSOUT     ;LOAD INTERRUPTS MASK
01563'062077            DOB 0,CPU       ;MASK OUT ALL DEVICES EXCEPT RTC
01564'004166-           JSR CLEAN       ;CLEAR TABLE TEMSP
01565'024030-           LDA 1,SPBOT     ;LOAD NEW STEAM SETPT.
01566'044027-           STA 1,N9+1      ;STORE SETPT. (FL. PT. #)
01567'020025-           LDA 0,SPTOP     ;LOAD NEW REFLUX SETPT.
01570'040024-           STA 0,N8+1      ;STORE SETPT. (FL. PT. #)
01571'102400            SUB 0,0         ;CLEAR AC0
01572'040023-           STA 0,N8        ;CLEAR OLD REFLUX FL PT SETPT.
01573'040026-           STA 0,N9        ;CLEAR OLD STEAM FL PT SETPT.

01574'001261'           FENT            ;ENTER INTERPRETER
01575'060023-           FFLO N8         ;FLOAT REFLUX SETPT.
01576'060026-           FFLO N9         ;FLOAT STEAM SETPT.
01577'100000            FEXT            ;EXIT INTERPRETER
01600'102400            SUB 0,0         ;CLEAR AC0
01601'062077            DOB 0,CPU       ;ENABLE INTERRUPT FOR ALL DEVICES

01602'001551'           .IOX            ;OUTPUT CR, LF
01603'000000            0
01604'110000            110000
01605'003124"           MES11*2
01606'000003            3
01607'001610'           .+1

01610'001317'           .FORK           ;CREAT NEW TASK
01611'000015            15              ;TO REINITIALIZE
01612'001614'           SP              ;PREVIOUS TERMINATING TASK

01613'001322'           .QUIT

01614'001610' SP:       .FORK           ;CREAT NEW TASK
01615'000020            20              ;SAME PRIORITY AND
01616'002415'           BREAK           ;SAME TASK AS PREVIOUS

01617'001613'           .QUIT
```

```
.EOT                    ;END OF SETPOINT TAPE
```

```
;MANUAL CONTROL OF REFLUX & STEAM CONTROLLERS

01620'006412  MSS:    .TXT ?<12><15>''
01621'047012  <12>N
01622'053517  OW
01623'047440   O
01624'020116  N
01625'040515  MA
01626'052516  NU
01627'046101  AL
01630'041440   C
01631'047117  ON
01632'051124  TR
01633'046117  OL
01634'005015  <15><12>
01635'052040  <40>T
01636'020117  O
01637'042524  TE
01640'046522  RM
01641'047111  IN
01642'052101  AT
01643'020105  E
01644'047111  IN
01645'052520  PU
01646'020124  T
01647'052047  'T
01650'005047  '<12>
01651'005015  <15><12>
01652'047524  TO
01653'041440   C
01654'040510  HA
01655'043516  NG
01656'020105  E
01657'040526  VA
01660'053114  LV
01661'020105  E<40>
01662'042523  SE
01663'052124  TT
01664'047111  IN
01665'051507  GS
01666'005015  <15><12>
01667'044440  <40>I
01670'050116  NP
01671'052125  UT
01672'023440   '
01673'023523  S'
01674'024040   (
01675'047506  FO
01676'020122  R
01677'052123  ST
01700'040505  EA
01701'024515  M)
01702'047440   O
01703'020122  R
01704'051047  'R
01705'020047  '
01706'043505  CF
01707'051117  OR
01710'051040   R
01711'043105  EF
```

```
01712'052514  LU
01713'024530  X)
01714'005015  <15><12>
01715'040440  <40>A
01716'042116  ND
01717'047040   N
01720'053505  EW
01721'051440   S
01722'052105  ET
01723'047520  PO
01724'047111  IN
01725'026124  T,
01726'040440   M
01727'047111  IN
01730'030075  =0
01731'020054  ,
01732'040515  MA
01733'036530  X=
01734'032463  35
01735'030060  00
01736'027060  0.
01737'024040   (
01740'053115  MV
01741'030452  *1
01742'030060  00
01743'030060  00
01744'027060  0.
01745'006451  )<15>
01746'020012  <12><40>
01747'042524  TE
01750'046522  RM
01751'047111  IN
01752'052101  AT
01753'020105  E
01754'044527  WI
01755'044124  TH
01756'041440   C
01757'005122  R<12>
01760'005015  <15><12>
01761'000000  ?


01762'054056- MAN:    STA 3,INMAN      ;STORE SUBROUTINE RETURN
01763'001602'         .IOX             ;OUTPUT INSTRUCTION MESSAGE
01764'000000          0
01765'100000          100000
01766'003440"         MSS*2
01767'000350          350
01770'001771'         .+1

01771'000407          JMP .+7          ;SKIP UNLESS ENTER FROM MANN

01772'001763' MANN:   .IOX             ;OUTPUT MESSAGE, 'NOW ON
01773'000000          0                ; MANUAL CONTROL'
01774'100000          100000
01775'003440"         MSS*2
01776'000033          33
01777'002000'         .+1
```

```
02000'010115-          ISZ MCTRL        ;INDICATE MANUAL CONTROL

02001'001772' MANRT:   .IOX             ;OUTPUT CR & LF
02002'000000           0
02003'100000           100000
02004'003440"          MSS*2
02005'000003           3
02006'002007'          .+1

02007'002001'          .IOX             ;INPUT CODE
02010'000000           0
02011'020000           20000
02012'000163-          MODMA
02013'000001.          1
02014'002015'          .+1

02015'002007'          .IOX             ;INPUT VALUE
02016'000000           0
02017'020000           20000
02020'000134-          TEMSP
02021'000015           15
02022'002023'          .+1

02023'020163-          LDA 0,MODMA      ;LOAD INPUT FROM TTY
02024'024130-          LDA 1,SINP       ;LOAD "S"
02025'030127-          LDA 2,RINP       ;LOAD "R"
02026'142415           SUB# 2,0,SNR     ;SKIP IF NOT = "R"
02027'000404           JMP FIRST        ;JUMP REFLUX CONTROL
02030'122415           SUB# 1,0,SNR     ;SKIP IF NOT = "S"
02031'000444           JMP SCON         ;JUMP STEAM CONTROL
02032'000507           JMP OUTM         ;ASSUME "T"

02033'024133- FIRST:   LDA 1,POM        ;LOAD POINTER
02034'006074-          JSR @DBIN1       ;CONVERT ASCII=BINARY
02035'044025-          STA 1,SPTOP      ;STORE NEW REFLUX SETPOINT
02036'044024-          STA 1,N8+1       ;STORE IN FL PT #
02037'102400           SUB 0,0          ;CLEAR AC0
02040'040023-          STA 0,N8         ;CLEAR OLD FL PT #

02041'004166-          JSR CLEAN        ;CLEAR TABLE TEMSP

02042'020165-          LDA 0,MSOUT      ;LOAD INTERRUPT MASK
02043'062077           DOB 0,CPU        ;MASK OUT ALL DEVICES EXCEPT RTC
02044'001574'          FENT             ;ENTER FL PT INTERPRETER
02045'060023-          FFLO N8          ;FLOAT NEW REFLUX SETPOINT
02046'020023-          FLDA 0,N8        ;LOAD NEW REFLUX SETPOINT
02047'024046-          FLDA 1,ACONS     ; A CONSTANT IN Y=AX+B
0205 '030050           FLDA 2,BCONS     ; B   STANT IN Y=AX+B
02051'104100           FMP              ; AX
02052'133000           FADD             ; AX+B
02053'050557           FST  OUTDM       ;STORE OUTPUT (VALVE POSITION)
   54'074556           FFIX OUTDM       ;CONV FL PT # = D PRES. INTEGER
02055'100000           FEXT             ;EXIT INTERPRETER
02056'102400           SUB 0,0          ;CLEAR AC0
02057'062077           DOB 0,CPU        ;ENABLE INTERRUPT FOR ALL DEVICES

02060'020553           LDA 0,OUTDM+1    ;LOAD OUTPUT
02061'006076-          JSR @DAMEX       ;CHECK D/A LIMITS
02062'040063-          STA 0,DTOUT      ;STORE NEW VALVE POSITION IN "M"
02063'024062-          LDA 1,NEGDA      ;REVERSE D/A OUTPUT
```

```
02064'106400            SUB 0,1         ;CHANGE "+" TO "-"
02065'044403            STA 1,.+3

02066'002015'           .IOX            ;OUTPUT REFLUX VALVE POSITION
02067'000004            4
02070'000000            0
02071'000001            1
02072'000001            1
02073'002074'           .+1

02074'000705            JMP MANRT       ;RESTART

02075'024133- SCON:     LDA 1,POM       ;LOAD POINTER
02076'006074-           JSR @DBINI      ;CONVERT ASCII=BINARY
02077'044030-           STA 1,SPBOT     ;STORE NEW STEAM SETPT.
02100'044027-           STA 1,N9+1      ;STORE IN FL. PT. #
02101'102400            SUB 0,0         ;CLEAR AC0
02102'040026-           STA 0,N9        ;CLEAR OLD FL. PT. #

02103'004166-           JSR CLEAN       ;CLEAR TABLE TEMSP

02104'020165-           LDA 0,MSOUT     ;LOAD INTERRUPT MASK
02105'062077            DOB 0,CPU       ;MASK OUT ALL DEVICES EXCEPT RTC
02106'002044'           FENT            ;ENTER FL PT INTERPRETER
02107'060026-           FFLO N9         ;FLOAT NEW STEAM SETPOINT
02110'020026-           FLDA 0,N9       ;LOAD NEW STEAM SETPOINT
02111'024046-           FLDA 1,ACONS    ; A CONSTANT IN Y=AX+B
02112'030050-           FLDA 2,BCONS    ; B CONSTANT IN Y=AX+B
02113'104100            FMPY 0,1        ; AX
02114'133000            FADD 1,2        ; AX+B
02115'050515            FSTA 2,OUTDM    ;STORE OUTPUT (VALVE POSITION)
02116'074514            FFIX OUTDM      ;CONV FL PT # = D PRES. INTEGER
02117'100000            FEXT            ;EXIT INTERPRETER
02120'102400            SUB 0,0         ;CLEAR AC0
02121'062077            DOB 0,CPU       ;ENABLE INTERRUPT FOR ALL DEVICES

02122'020511            LDA 0,OUTDM+1   ;LOAD OUTPUT
02123'006076-           JSR @DAMEX      ;CHECK D/A LIMITS
02124'040064-           STA 0,BTOUT     ;STORE NEW VALVE POSITION IN "M"
02125'030062-           LDA 2,NEGDA     ;REVERSE D/A OUTPUT
02126'112400            SUB 0,2         ;CHANGE "+" TO "-"
02127'024061-           LDA 1,CHA1      ;LOAD CHANNEL #1 MODE
02130'133000            ADD 1,2         ;ADD TO OUTPUT
02131'050403            STA 2,.+3

02132'002066'           .IOX            ;OUTPUT STEAM VALVE POSITION
02133'000004            4
02134'000000            0
02135'000001            1
02136'000001            1
02137'002140'           .+1

02140'000641            JMP MANRT       ;RESTART

02141'102400  OUTM:     SUB 0,0         ;CLEAR AC0
02142'040035-           STA 0,N12       ;CLEAR OLD ERROR FOR REFLUX
02143'040036-           STA 0,N13
02144'040037-           STA 0,N14       ;CLEAR OLD ERROR FOR STEAM
02145'040040-           STA 0,N15
```

```
02146'020165-          LDA 0,MSOUT      ;LOAD INTERRUPT MASK
02147'062077           DOB 0,CPU        ;MASK OUT ALL DEVICES EXCEPT RTC
02150'002106'          FENT             ;ENTER FL. PT. INTERPRETER
02151'060035-          FFLO N12         ;FLOAT OLD REFLUX ERROR
02152'060037-          FFLO N14         ;FLOAT OLD STEAM ERROR
02153'100000-          FEXT             ;EXIT INTERPRETER
02154'102400           SUB 0,0          ;CLEAR AC0
02155'062077           DOB 0,CPU        ;ENABLE INTERRUPT FOR ALL DEVICES

02156'002132' LOPMN:   .IOX             ;OUTPUT MESSAGE 9
02157'000000           0
02160'100000           100000
02161'004470"          MS9*2
02162'000150           150
02163'002164'          .+1

02164'002156'          .IOX             ;INPUT CHARACTER
02165'000000           0
02166'020000           20000
02167'000163-          MODMA
02170'000001           1
02171'002172'          .+1

02172'020163-          LDA 0,MODMA      ;LOAD INPUT CHARACTER
02173'024127-          LDA 1,RINP       ;LOAD 'R', (REFLUX)
02174'030130-          LDA 2,SINP       ;LOAD 'S', (STEAM)
02175'034126-          LDA 3,NINP       ;LOAD 'N', (NO)
02176'162415           SUB# 3,0,SNR     ;SKIP IF NOT = 'N'
02177'000406           JMP .+6          ;EXIT
02200'010116-          ISZ MAKI         ;INDICATE MANUAL TO KI CHANGE
02201'122415           SUB# 1,0,SNR     ;SKIP IF NOT = 'R'
02202'002121-          JMP @KTI         ;JUMP TO KTI1
02203'142415           SUB# 2,0,SNR     ;SKIP IF NOT = 'S'
02204'002122-          JMP @STAM        ;JUMP TO STEM

02205'002164'          .IOX             ;ASSUME 'N' FOR NO
02206'000000           0                ;OUTPUT CR, LF
02207'100000           100000
02210'004470"          MS9*2
02211'000003           3
02212'002213'          .+1

02213'102400           SUB 0,0          ;CLEAR AC0
02214'040116-          STA 0,MAKI       ;CLEAR K,TI INDICATOR
02215'040115-          STA 0,MCTRL      ;INDICATE MANUAL OFF
02216'010054-          ISZ MANAU        ;INDICATE MANUAL TO AUTO TRANSFER

02217'024055-          LDA 1,STMAN      ;LOAD START-UP INDICATOR
02220'125004           MOV 1,1,SZR      ;START-UP ?
02221'002056-          JMP @INMAN       ;YES, RETURN TO INITIALIZATION
                                        ;NO, CONTINUE

02222'001614'          .FORK            ;CREAT NEW TASK
02223'000015           15               ;TO REINITIALIZE
02224'002226'          MAN3             ;PREVIOUS TERMINATING TASK

02225'001617'          .QUIT

02226'002222' MAN3:    .FORK            ;CREAT NEW TASK
02227'000020           20               ;SAME PRIORITY AND
```

```
02230'002415'                BREAK          ;SAME TASK AS PREVIOUS

02231'002225'                .QUIT


02232'000000  OUTDM:         0              ;D/A OUTPUT
02233'000000                 0              ; IN FLOATING PT. NOTATION

02234'006412  MS9:           .TXT !<12><15>
02235'045412  <12>K
02236'052054  ,T
02237'020111  I
02240'044103  CH
02241'047101  AN
02242'042507  GE
02243'020123  S
02244'006477  ?<15>
02245'020012  <12><40>
02246'047111  IN
02247'052520  PU
02250'020124  T
02251'047047  'N
02252'020047  '
02253'047050  (N
02254'024517  O)
02255'020054  ,
02256'051047  'R
02257'020047  '
02260'051050  (R
02261'043105  EF
02262'052514  LU
02263'024530  X)
02264'047440   O
02265'020122  R
02266'051447  'S
02267'020047  '
02270'051450  (S
02271'042524  TE
02272'046501  AM
02273'020051  )<40>
02274'020040  <40><40>
02275'000000  !

                             .EOT           ;END OF MANUAL CONTROL TAPE
```

```
                ;K,TI CHANGES IN STEAM SETTINGS

02276'006412    MESS5:  .TXT ?<12><15>
02277'050012    <12>P
02300'047522    RO
02301'047520    PO
02302'052122    RT
02303'047511    IO
02304'040516    NA
02305'020114    L
02306'040507    GA
02307'047111    IN
02310'024040     (
02311'024513    K)
02312'043040     F
02313'051117    OR
02314'051440     S
02[redacted]'042524    TE
02[redacted]'046501    AM
02317'041440     C
02320'047117    ON
02321'051124    TR
02322'046117    OL
02323'042514    LE
02324'020122    R
02325'025050    (*
02326'030061    10
02327'027060    0.
02330'020051    )<40>
02331'020040    <40><40>
02332'000000    ?

02333'006412    MESS6:  .TXT ?<12><15>
02334'044412    <12>I
02335'052116    NT
02336'043505    EG
02337'040522    RA
02340'020114    L
02341'044524    TI
02342'042515    ME
02343'043040     F
02344'051117    OR
02345'051440     S
02346'042524    TE
02347'046501    AM
02350'024040     (
02351'030452    *1
02352'030060    00
02353'024456    .)
02354'020040    <40><40>
02355'000040    <40>?

02356'006412    MESS7:  .TXT ?<12><15>
02357'044412    <12>I
02360'050116    NP
02361'052125    UT
02362'047040     N
02363'053505    EW
02364'050040     P
02365'047522    RO
```

```
02366'047520  PO
02367'052122  RT
02370'047511  IO
02371'040516  NA
02372'020114  L
02373'040507  GA
02374'047111  IN
02375'024040   (
02376'030452  *1
02377'030060  00
02400'024456  .)
02401'005015  <15><12>
02402'052040  <40>T
02403'051105  ER
02404'044515  MI
02405'040516  NA
02406'042524  TE
02407'053440   W
02410'052111  IT
02411'020110  H
02412'051103  CR
02413'020040  <40><40>
02414'000040  <40>?

02415'177777  BREAK:  .BRK              ;BREAK AT INPUT "A".
02416'000101          101

02417'002205'         .IOX              ;INPUT CODE LETTER
02420'000000          0
02421'020000          20000
02422'002603'         MODEM
02423'000001          1
02424'002425'         .+1

02425'0245            LDA 1,
02426'020125-         LDA 0,
02427'030127-         LDA 2,RINP        ;LOAD "R" FOR REFLUX
02430'034130-         LDA 3,SINP        ;LOAD "S" FOR SETPT
02431'106415          SUB# 0,1,SNR      ;SKIP IF NOT = "M"
02432'002117-         JMP @MAN1         ;JUMP TO MAN
02433'020126-         LDA 0,NINP        ;LOAD "N" FOR MANUAL (NO TEXT)
02434'106415          SUB# 0,1,SNR      ;SKIP IF NOT = "N"
02435'002120-         JMP @MAN2         ;JUMP TO MANN
02436'146415          SUB# 2,1,SNR      ;SKIP IF NOT = "R"
02437'002121-         JMP @KTI          ;JUMP TO KTI1
02440'166415          SUB# 3,1,SNR      ;SKIP IF NOT = "S"
02441'002123-         JMP @SP2          ;JUMP TO SP1
02442'030124-         LDA 2,BINP        ;LOAD "B" FOR BOTTOMS
02443'146415          SUB# 2,1,SNR      ;SKIP IF NOT = "B"
02444'000402          JMP STEM          ;JUMP TO STEM
02445'000514          JMP ERROR         ;ERROR, EXIT

02446'002417' STEM:   .IOX              ;OUTPUT MESSAGE 5
02447'000000          0
02450'100000          100000
02451'004574"         MESS5*2
02452'000100          100
02453'002454'         .+1

02454'024013-         LDA 1,KBOT        ;GET K STEAM VALUE
```

```
02455'006075-          JSR @BNDE1          ;OUTPUT VALUE

02456'002446'          .IOX                ;OUTPUT MESSAGE 6
02457'000000           0
02460'100000           100000
02461'004666"          MESS6*2
02462'000100           100
02463'002464'          .+1

02464'024010-          LDA 1,TIBOT         ;GET TI STEAM VALUE
02465'006075-          JSR @BNDE1          ;OUTPUT VALUE

02466'002456'          .IOX                ;OUTPUT MESSAGE 7
02467'000000           0
02470'100000           100000
02471'004734"          MESS7*2
02472'000100           100
02473'002474'          .+1

02474'002466'          .IOX                ;INPUT NEW VALUE
02475'000000           0
02476'020000           20000
02477'000134-          TEMSP
02500'000020           20
02501'002502'          .+1

02502'024133-          LDA 1,POM           ;LOAD POINTER
02503'006074-          JSR @DBIN1          ;CONVERT ASCII=BINARY
02504'044013-          STA 1,KBOT          ;STORE NEW K (INTEGER #)

02505'004166-          JSR CLEAN           ;CLEAN TABLE TEMSP

02506'002474'          .IOX                ;OUTPUT MESSAGE 8
02507'000000           0
02510'100000           100000
02511'005410"          MESS8*2
02512'000100           100
02513'002514'          .+1

02514'002506'          .IOX                ;INPUT NEW VALUE
02515'000000           0
02516'020000           20000
02517'000134-          TEMSP
02520'000020           20
02521'002522'          .+1

02522'024133-          LDA 1,POM           ;LOAD POINTER
02523'006074-          JSR @DBIN1          ;CONVERT ASCII=BINARY
02524'044010-          STA 1,TIBOT         ;STORE NEW TI (INTEGER #)
02525'020077           LDA 0,CPU           ;LOAD INTERRUPT MASK
02526'062077           DOB 0,CPU           ;MASK OUT ALL DEVICES EXCEPT RTC
02527'044007-          STA 1,N3+1          ;STORE NEW TI (FL. PT. #)
02530'024073-          LDA 1,K100          ;LOAD CONSTANT 100.
02531'044072-          STA 1,C100+1        ;STORE CONSTANT
02532'024013-          LDA 1,KBOT          ;LOAD NEW K (INTEGER #)
02533'044012-          STA 1,N4+1          ;STORE NEW K (FL. PT. #)
02534'102400           SUB 0,0             ;CLEAR AC0
02535'040011-          STA 0,N4            ;CLEAR OLD FL. PT. K VALUE
02536'040006-          STA 0,N3            ;CLEAR OLD FL. PT. TI VALUE
02537'040071-          STA 0,C100          ;CLEAR OLD CONSTANT
```

```
02540'004166-          JSR CLEAN        ;CLEAN TABLE TEMSP

02541'002150'          FENT             ;ENTER INTERPRETER
02542'060006-          FFLO N3          ;FLOAT TI STEAM
02543'060011-          FFLO N4          ;FLOAT K STEAM
02544'020006-          FLDA 0,N3        ;LOAD TI CONSTANT STEAM
02545'024014-          FLDA 1,N5        ;LOAD LOOP TIME CONSTANT
02546'104200           FDIV 0,1         ;DIVIDE, T/TI
02547'124300           FHLV 1,1         ;HALF RESULT
02550'044021-          FSTA 1,N7        ;STORE, T/2TI
02551'060071-          FFLO C100        ;FLOAT CONSTANT 100.
02552'020071-          FLDA 0,C100      ;LOAD CONSTANT
02553'024011-          FLDA 1,N4        ;LOAD K STEAM
02554'104200           FDIV 0,1         ;DIVIDE BY 100.
02555'044011-          FSTA 1,N4        ;STORE CORRECT K STEAM
02556'100000           FEXT             ;EXIT INTERPRETER
02557'102400           SUB 0,0          ;CLEAR AC0
02560'062077           DOB 0,CPU        ;ENABLE INTERRUPT FOR ALL DEVICES

02561'002514' ERROR:   .IOX             ;OUTPUT CR, LF
02562'000000           0
02563'110000           110000
02564'005410"          MESS8*2
02565'000003           3
02566'002567'          .+1

02567'020116-          LDA 0,MAKI       ;LOAD MANUAL TO KTI INDICATOR
02570'101005           MOV 0,0,SNR      ;SKIP IF NOT = 0
02571'000402           JMP .+2          ;EXIT
02572'002077-          JMP @MKIRT       ;RETURN TO MANUAL

02573'002226'          .FORK            ;CREAT NEW TASK
02574'000015           15               ;TO REINITIALIZE
02575'002577'          KTI3             ;PREVIOUS TERMINATING TASK

02576'002231'          .QUIT

02577'002573' KTI3:    .FORK            ;CREAT NEW TASK
02600'000020           20               ;SAME PRIORITY AND
02601'002415'          BREAK            ;SAME TASK AS PREVIOUS

02602'002576'          .QUIT

02603'000000  MODEM:   0                ;INPUT STORAGE

02604'006412  MESS8:   .TXT ?<12><15>
02605'044412  <12>I
02606'050116  NP
02607'052125  UT
02610'044440   I
02611'052116  NT
02612'043505  EG
02613'040522  RA
02614'020114  L
02615'044524  TI
02616'042515  ME
02617'041440   C
02620'047117  ON
02621'052123  ST
```

```
02622'047101   AN
02623'020124   T
02624'025050   (*
02625'030061   10
02626'027060   0.
02627'005051   )<12>
02630'020015   <15><40>
02631'052040   <40>T
02632'051105   ER
02633'044515   MI
02634'040516   NA
02635'042524   TE
02636'053440    W
02637'052111   IT
02640'020110   H
02641'051103   CR
02642'020040   <40><40>
02643'000040   <40>?

                    .END                    ;END OF STEAM K>TI TAPE
```

APPENDIX III

EXPERIMENTAL RUN DATA

TABLE 2 CONTROL SETTINGS FOR EXPERIMENTAL RUN

	K_c	τ_I
Steam Flowrate	0.2	0.3 sec^{-1}
Reflux Flowrate	0.05	6.0 sec^{-1}

TABLE 3

EXPERIMENTAL RUN No. 1

STEADY STATE CONDITIONS

Feed Flowrate	=	0.5 USGPM
Bottoms Flowrate	=	0.39 USGPM
Distillate Flowrate	=	0.11 USGPM
Reflux Flowrate	=	9000 as computer setpoint (0.17 USGPM)
Steam Flowrate	=	18000 as computer setpoint (9.2 psig signal to control valve)

TEMPERATURE	PROFILE
Tray No. 1	202°F
Tray No. 2	199°F
Tray No. 3	196°F
Tray No. 4	191°F
Feed	189°F
Tray No. 5	192°F
Tray No. 6	191°F
Tray No. 7	187°F
Tray No. 8	179°F
Tray No. 9	169°F
Tray No. 10	160°F

TABLE 4

EXPERIMENTAL RUN NO. 2

STEADY STATE CONDITIONS

Feed Flowrate	=	0.48	USGPM
Bottoms Flowrate	=	0.38	USGPM
Distillate Flowrate	=	0.10	USGPM
Reflux Flowrate	=	8000	as computer setpoint (0.12 USGPM)
Steam Flowrate	=	18000	as computer setpoint (9.2 psig signal to control valve)

TEMPERATURE	PROFILE
Tray No. 1	209°F
Tray No. 2	201°F
Tray No. 3	198°F
Tray No. 4	192°F
Feed	193°F
Tray No. 5	194°F
Tray No. 6	194°F
Tray No. 7	193°F
Tray No. 8	191°F
Tray No. 9	183°F
Tray No. 10	168°F

TABLE 5

EXPERIMENTAL RUN NO. 3

STEADY STATE CONDITIONS

Feed Flowrate	=	0.50	USGPM
Bottoms Flowrate	=	0.44	USGPM
Distillate Flowrate	=	0.09	USGPM
Reflux Flowrate	=	10000	as computer setpoint (0.20 USGPM)
Steam Flowrate	=	18500	as computer setpoint (9.6 psig signal to control valve)

TEMPERATURE	PROFILE
Tray No. 1	198°F
Tray No. 2	195°F
Tray No. 3	195°F
Tray No. 4	193°F
Feed	190°F
Tray No. 5	183°F
Tray No. 6	174°F
Tray No. 7	165°F
Tray No. 8	158°F
Tray No. 9	155°F
Tray No. 10	153°F