BNWL-1795

A MICROPROCESSOR SAMPLED DATA PROCESS CONTROLLER

T. A. Seim

Battelle
Pacific Northwest Laboratories
Richland, Washington 99352

NOVEMBER 1973

Prepared for the U.S. Atomic Energy Commission under Contract AT(45-1):1830

BNWL-1795

NOTICE

PACIFIC NORTHWEST LABORATORY
operated by
BATTELLE
for the
U.S. ATOMIC ENERGY COMMISSION
Under Contract AT(45-1)-1830

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, Virginia 22151
Price: Printed Copy $5.45; Microfiche $1.45

AEC-RL RICHLAND, WASH.

BNWL-1795

A MICROPROCESSOR SAMPLED DATA PROCESS CONTROLLER

by

T. A. Seim

November 1973

BATTELLE
PACIFIC NORTHWEST LABORATORIES
RICHLAND, WASHINGTON

A MICROPROCESSOR SAMPLED DATA PROCESS CONTROLLER

ABSTRACT

A micro-miniaturized digital processor was utilized in the development of a sampled data process controller. While general purpose in nature, the processor was applied specifically to control temperature. Physically the processor was found to be well suited for industrial environments, as it's relatively slow speed and high switching voltage levels made it exceptionally noise immune. Successful application was found to be more dependent on external software development support than anything else. This is because of the very limited nature of the processor in terms of memory and input-output peripherals.

SUMMARY

This report documents the application of an Intel MCS-8 microprocessor to sampled data process control. Microprocessors are small scale digital processors which offer low cost, small size, and high reliability to areas such as process control. The programming and application of microprocessors, however, present problems much different than those associated with minicomputer applications. Hence a major portion of the development was expended solving these problems, most of which centered around software development. It was concluded that software could only be conveniently and inexpensively developed with the use of a larger computer system.

The MCS-8 was found to be a highly reliable unit, and considerably more immune to electrical noise than conventional digital logic. This is due in part to its p-channel MOS construction, and in part to its relatively slow execution speed (higher speed logic is inherently more susceptible to electrical noise).

Because of the low cost of microprocessors, many new application areas will become feasible. But this depends directly on minimizing other development costs as well, notably software development costs. The potential impact of microprocessors on technology and productivity is tremendous if the application problems are known and understood.

CONTENTS

CONTENTS (continued)

ILLUSTRATIONS

A MICROPROCESSOR SAMPLED DATA PROCESS CONTROLLER

INTRODUCTION

Under initial funding from Lawrence Livermore Laboratory, Livermore, California, and later by funding from Dow Chemical, Rocky Flats Division, Golden, Colorado, Battelle was directed to undertake two tasks. First, establish the feasibility of using large scale integrated (LSI) circuit computers known as *microprocessors* in industrial environments; and secondly to implement a digital brazing process controller with a microprocessor. This report is concerned primarily with the later task, but discusses topics of the former where pertinent.

Application Description

The wider goal, of which this development was a part, was to upgrade the performance of a brazing process and minimize uncertainties of the joint parameters. This, in turn, would reduce rejects and improve component reliability, and would ease the problems of a component failure analysis.

The braze is done under high vacuum, with heat applied by an inductively coupled coil. The coil is fed by a three phase motor-generator. Heating energy is controlled by the voltage applied to the generator's primary winding. Presently the brazing temperature is uncontrolled, and its rise time varies from part to part, depending upon the degree of coupling between the heating coil and the part. Also the temperature rise, or overshoot, after the braze occurs is unknown. The object of closed loop control is to have repeatable rise time and minimal overshoot. It was also specified that a post-braze temperature hold be maintained for a selectable period of time.

During discussions of this problem at Battelle in March of 1972 it was agreed that a small digital processor with its control program stored in an unchangeable type of media, termed a *read only memory* (ROM)[1], would satisfy the requirements of the controller for the brazing process. Recent semiconductor developments had made available a digital processing unit in a single integrated circuit (IC), which was ideal for the processor required by the controller. From this base a controller was developed using the microprocessor and analog input and output subsystems. The use of ROM for program storage eliminated the problems of alteration caused by electro-magnetic interference (EMI).[2]

Feasibility

The use of microprocessors in industrial applications have been demonstrated to be a viable alternative to conventional control techniques. They offer improvements in reliability, capability , and cost. Further, such advantages as vastly simplified maintenance techniques (the "throw-away" philosophy) will minimize the necessity of employing skilled technicians to keep this equipment in operating order, which is especially inviting at a time when skilled labor is becoming increasingly scarce. But this is the positive outlook, the negative aspects include the fact that digital processors are high technology devices, and require at least an acquaintence with the numerous technologies they touch on. The difficulty is that many of these areas are simply out of the realm of the average control engineer. In other words, some education and re-orientation of one's thinking are in order. For instance, the idea that reliability is a function of the switching function complexity (i.e. the number of relay contacts) does not hold for microprocessors, where reliability is more a function of the number of interconnections. As the complexity of the

1. *The actual memory used could not only be programmed, but could also be erased and reprogrammed indefinitely. It was read only, or unchangeable, in the sense that this operation could not be done under program control, but took a special programming device which applied programming pulses unavailable to the processor.*

2. *Some programmable controllers used core memory for program storage, and had problems because of EMI.*

individual circuits increase the number of interconnections drop significantly, thus yielding a considerable reliability increase.

Obsolescence of the controller is minimized by (a) the universality of the control element, and (b) the ability of the unit to be reprogrammed. The latter feature is especially important with an in-place process which must be updated to some new specifications. Redesign of an existing relay control panel is a costly and time-consuming task, but the alteration of a control program is a far less formidable undertaking. Testimony to this observation is readily available from numerous special purpose controllers now obsolete, while computers procured at the same time are still performing valuable functions.

Apparent contradictions will be the order of the day for some time for those recently introduced to this area. The initial impression one has of microprocessors is that of unbridled complexity. Yet microprocessors represent simplicity itself. Instead of being confronted with a bewildering array of semi-compatible control elements the microprocessor represents a definite move toward standardization. Instead of hundreds or thousands of unique components a control function can be performed by single component, the digital processor. Many functions can be executed by one processor, whose particular character is governed by the control program.

The control engineer who is unacquainted with computer technology is not the only one caught in the tides of change. Engineers already associated with computer applications must accept some radically different theorems of use ("rules of thumb") concerning microprocessors. For the most part the central processor was treated as a scarce resource which must be utilized to the utmost. This is direct result of the cost of the processor and its associated peripherals. Consequently much effort has been expended to make programs as efficient as possible, and to maximize the number of functions performed by the processor (the most irritating sight a programmer can see is a computer which is not being used). This concept is being invalidated by the microprocessor, whose cost is becoming

an insignificant factor. In other words the processor has changed from a scarce resource to an overabundant resource, which can be used almost at will. The impact of this new factor alone is so immense as to defy full consideration.

If the processor is no longer the scarce resource, what is? Unquestionably this is in the domain of invested man-time, which continues to rise and is accelerated by the rapidly changing technology. If savings are to be realized it will be by minimizing the invested man time in each application. Specific recommendations will be made as how to fully use microprocessors, while minimizing the investment in both man-time and equipment. But it must be expected that a greater development of capabilities is required than, say, relay controller design. It is only through the availability of powerful development tools that the full advantage of microprocessors can be gained.

Sampled Data Control Systems

Originally sampled data control systems implied the existence of a *sampler* in what was otherwise a continuous system. The sampler would measure a continuous signal at discrete points in time for short intervals, thereby approximating the input with a pulse train similar to Fig. 1.

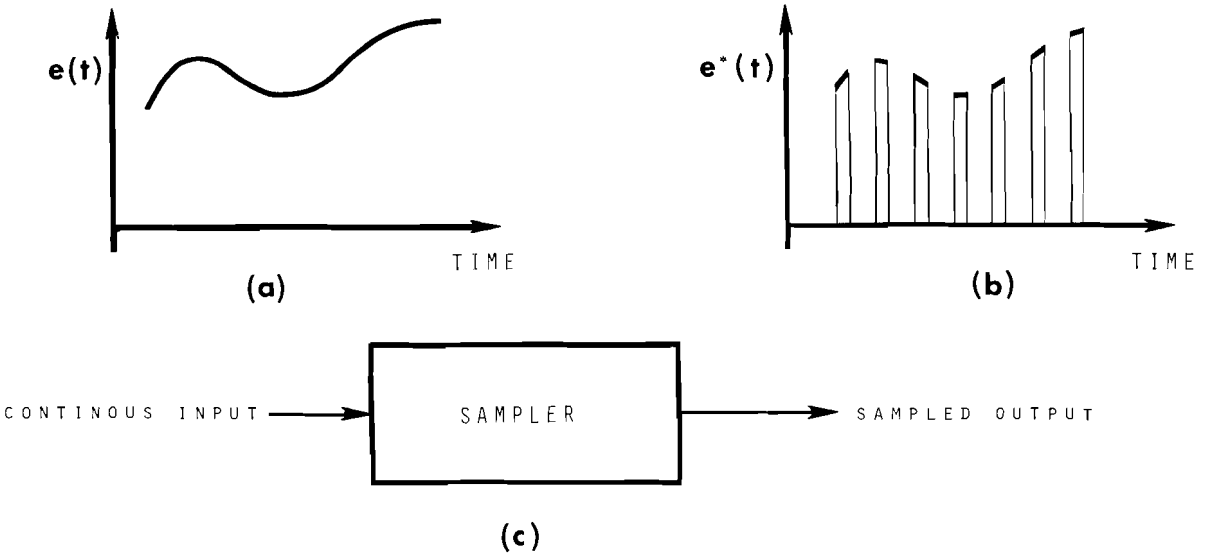


FIGURE 1. Signal sampling

While sampled data control systems still are used which contain continuous type components this term is more often applied to digital control systems which, by the very nature, are discrete. Indeed, it was the advent of practical digital control systems that motivated the theoretical development of sampled data control.

Unfortunately until very recently the cost and reliability of a completely digital system had a hard time competing directly with analog controllers economically. The prime use of computers was not in the control loops, but in a supervisory role performing functions which simply could not be done with analog techniques. Now, however, the economics of LSI make it feasible for general digital controllers to compete directly with their analog counterparts, almost on a loop-for-loop basis.

Feedback Control

The initial goal was to implement a single feedback control loop as diagrammed in Fig. 2.

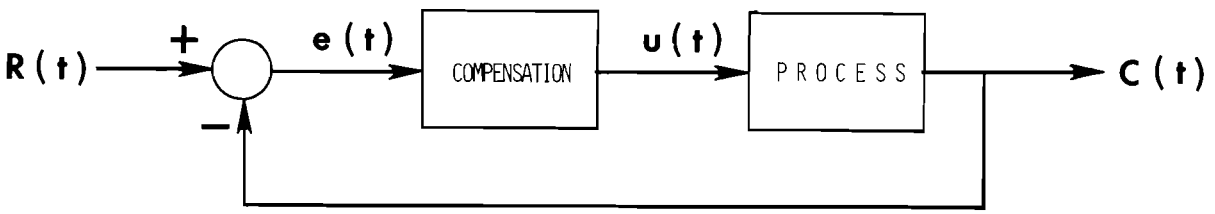


FIGURE 2. Single loop feedback control

The target, or desired control point is R(t), while C(t) is the actual process output, in this case temperature. Their difference, the control error e(t), is what the controller uses to compute the control action, U(t).

As the time functional notation suggests R(t) is not a constant "setpoint," but is rather a function of time, or a time profile. This allows the temperature to be increased in a controlled manner during a brazing cycle and makes the temperature control more stable.

The feedback control system of Fig. 2 takes on the physical formulation of Fig. 3.

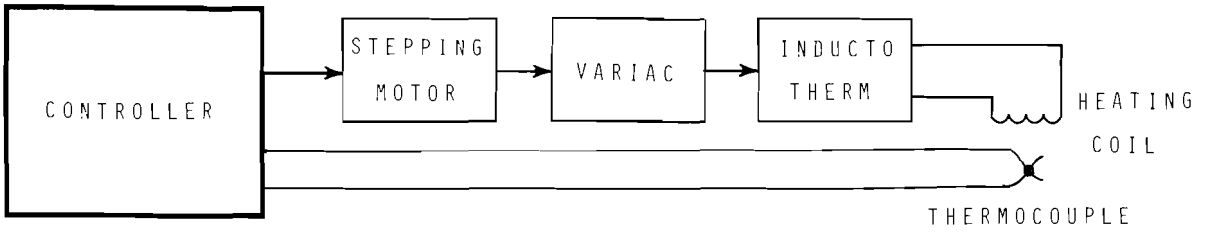


FIGURE 3. Controller internal view

The temperature, which corresponds to C(t), is sensed by a thermocouple, and is amplified and converted to digital form by the controller. The control output, U(t), is in the form of rotary motion derived from a stepping motor. This is converted to electrical energy by a Variac to ultimately supply power to a heating coil. The independent control variable, R(t), is generated internally by the controller from a profile table.

Development Sequence

While the end result is most visible in the hardware form the major part of the development effort was in software. This includes not only the control program, but several other programs written for a Battelle-owned SEL 840. The SEL 840 proved to be a convenient vehicle with which to develop the controller software, most of which was designed and operational before any hardware had been received. Part of the SEL 840 software included a simulator for the microprocessor. With this program the SEL 840 simulated the microprocessor in every detail, allowing it to be used to debug software for use on the microprocessor.

Final design of the hardware was not done until the software was completely operational. This resulted in the least number of hardware design changes, and the most efficient software design. Then the hardware was breadboarded using a device designed for this purpose by the manufacturer of the microprocessor. This breadboard was used to find any errors in the control program, and to check out a data link to the SEL 840. This

data link was used primarily to load programs into the controller (the programs were stored in volatile memory at this time). Because all of these preparatory steps were taken the main task involved with the prototype were final fabrication details, as most all the design problems had been solved. The prototype controller, remote control box, and stepping motor are pictured in Fig. 4.

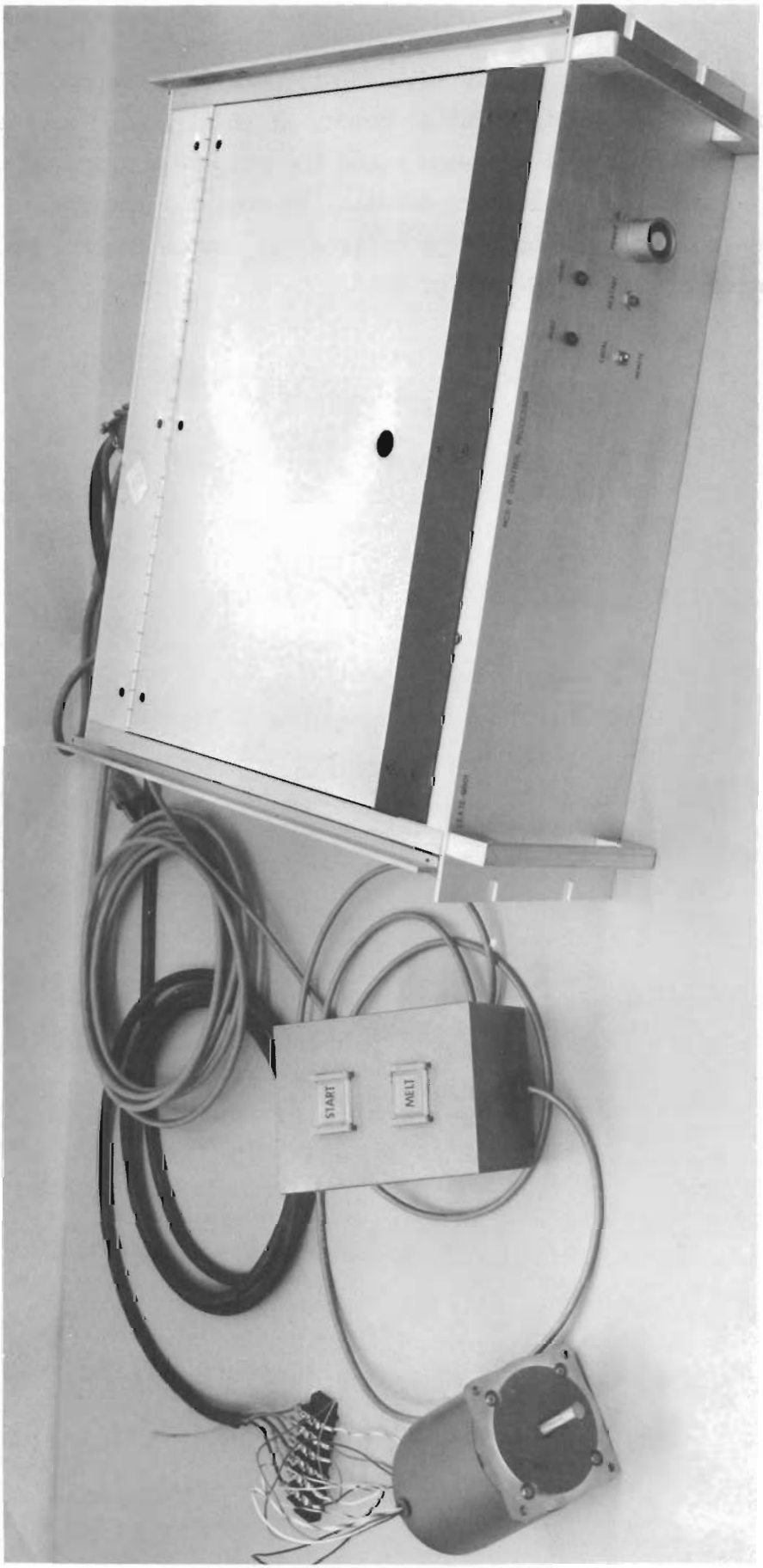


FIGURE 4. Prototype braze controller

CONCLUSIONS AND RECOMMENDATIONS

Reliability

The initial conclusion I have reached based on personal experience with MCS-8 processors at Battelle and other installations is that they are highly reliable and relatively immune to EMI. The only reliability problems I encountered were with the RAM memory units.[1] These exhibited a failure rate which was unusually high (ten percent of exercised units). The sample size, however, was too small (40 IC's) to base a firm judgment on.

The PROM's were never observed to fail when used properly. They are probably a more reliable memory medium than the more common magnetic core memories. The later devices are known to be susceptible to EMI commonly found in industrial environments. The immunity of the PROM's is due to the insulated floating gate on which the data are stored (each integrated circuit has 2048 such gates). This gate is isolated in an extremely high resistance material. Data are stored by avalanche injecting a charge into the gate by relatively high voltage pulses. Since EMI is highly unlikely to duplicate these conditions the only possibility for loss of data is through degradation of the charge stored on the gate by leakage (it can also be erased with a high intensity short wavelength ultraviolet light). This degradation has been measured and found to take years to occur (only about three years data are available, with estimates based on projections past this point). This failure mechanism is considered to be insignificant as annual, or bi-annual, re-programming defeats it.

Reliability of the processor can be further increased by minimizing the digital logic external to the processor. As is explained in the section on the micro computer set, much of this external circuitry is due to the choice of packaging of the MCS-8. The device is pin limited and must multiplex data in and out of the processor. Much of the external circuitry is required to handle this multiplexing.

---

1 *Volatile storage devices used for storing program variables.*

Other Processors

Intel will soon introduce an improved version of the 8008 processor, numbered 8080, which will eliminate these and other problems with the unit. Furthermore, programs written for the 8008 will run on the 8080 (which is what is called "upwards compatible"). The instruction set will be expanded by about 60 percent more instructions than are available with the 8008, and will execute these instructions several times faster than the 8008.

Future applications should definitely consider other available microprocessors. Currently most semiconductor manufacturers are developing microprocessors, some of which are available. An incomplete list of these companies is:

American Microsystems, Inc.
Computer Automation
Fairchild Semiconductor
National Semiconductor
RCA
Rockwell International
Teledyne
Texas Instruments.

Of these only Computer Automation and Rockwell International have units available, while Teledyne repackages chips manufactured by some other firm (as many as 30 companies are packaging the Intel processors, mostly the MCS-4).

Choice of the processor will increasingly depend less on performance and more on software support. This is a simple result of the generally long time constants found in process control (high performance applications, of course, will be an exception). In other words if a processor can perform a task within specified time constraints the major consideration will be what it costs to build and program it. And since the costs of building the processor are decreasing the major cost will increasingly be programming (high volume applications are the exception here).

Cost Reduction

Software costs can be minimized in at least two ways. The first way is to take advantage of vendor supplied software. This typically consists of programs such as assemblers, compilers, and operating systems. Selection of a processor can in part be based on the software available for it and the cost of that software. Intel offers programs written in Fortran to assemble code for the MCS-8 and simulate the MCS-8, either as an outright sale or from a time sharing service. For us neither of those choices was acceptable due to the high cost. It was more economical to develop the same programs for our own computer system. The reason for that lies in the fact that this type of program, when written in Fortran, becomes machine dependent. The effort involved in adapting the Fortran code for our own system plus the acquisition cost made it more economical to design what we wanted to begin with.

Computer Automation, a minicomputer manufacturer as opposed to a semiconductor manufacturer, based their micro design on a previous minicomputer series. Since the instruction sets were compatible, software developed for the earlier series can be used with the micro series. This presents the tremendous advantage of a large body of existing software. It must be understood that the Computer Automation processor is considerably more powerful than the MCS-8, being comparable to most minicomputers in all respects with the exception of speed. Thus with a reasonable amount of memory (at least 4K words) it is possible to operate an assembler with the unit. The primary disadvantage here is with the lack of high speed peripherals and mass storage. This can be overcome by using it in conjunction with another computer system already possessing those peripherals, communicating by means of a medium speed data link (an asynchronous 9600 baud interface will do nicely).

Another approach to take is to develop a simulator for the target computer, and execute the assembler with simulator on the host computer. This has proven satisfactory in the past, and has the great advantage of not requiring the availability of the target computer. The value of a good simulator, in this and other respects, should not be under-estimated.

Almost all of the software development for this project was done in this fashion, with few changes when run on the actual processor. If this approach had not been taken a delay of between one and two months would have resulted. In addition, program changes can be tested with the simulator before they are installed in the field. If the process operation is critical, or the time to deliver and install a change lengthy, this alone can be justification for the simulator's development.

The other area where software development costs can be saved falls in the category of remote site assistance. Generally the application process will be physically separated from the research and development location, perhaps by thousands of miles. Thus it may be inconvenient, or patently impractical, to make repeated trips to the host computer for program changes and re-assemblies. Yet the use of these facilities is very useful for such purposes as

- Program debug
- Equipment installation
- Program changes
- Process alterations.

The obvious conclusion is some sort of data link between the host computer and the process control site. Where it is infeasible to install a dedicated communications line the telephone lines can be used. This requires a data modulator/demodulator (modem) at each location. The use of switched telephone lines for data transmission involves well known problems. These problems can and have been solved by the use of error checking techniques. The teletype interface can be used to complete the data link from the modem to the MCS-8 processor (the serial data format is compatible with most modems). In this case the Teletype cannot be used while the modem is connected to the processor. A second telephone line would be necessary for the operators at each end to communicate. Another alternative is the addition of an interface dedicated to the modem so that the Teletype would be on-line with the processor while the communications link is established.

The primary use of the data link would be to load programs in writable storage (RAM), but other uses come to mind. For instance, by means of the necessary "operating system" (not to be confused with operating systems developed for larger systems) the system developer could monitor and change program data and parameters during operation. At present this can only be done by stopping the processor and, hence, the process.

Packaging

Packaging and interface hardware can easily dwarf the cost of the microprocessor, its peripheral logic, and memory. And adapting a particular microprocessor to a package configuration consistent with the user's usual electronics packaging can be an expensive proposition. Packaging alone can sway the choice from a lower cost, higher performance processor, merely because the older processor is already available in a desirable package.

INSTALLATION

Controller

The controller rack mounts and is equipped with ball bearing slides with a tilt-lock mechanism. Other than physical proximity to the brazing process the only consideration need be given toward placement is the availability of free air flow behind the unit. The box has two fans, one for blowing cool air in and the other for exhausting hot air. The input fan has no filter so if installation is in a dusty area it may be wise to add a filter to this fan.

Cables

The unit has three cables which mate to the controller via two connectors. The analog input uses a special twisted pair shielded cable and three conductor coaxial connector (both manufactured by Trompeter Electronics), J56, for minimal electrical noise pickup (Fig. 5).

The other two cables are for the remote control box and the stepping motor. They both use J53, control output, for mating to the controller, and are both ten feet long.

Stepping Motor

The HS-50 stepping motor requires a 15 volt, 6 amp power supply, connected by terminal strip J55 (Fig. 5). If desired, the power can be provided by two 15 volt, 3 amp supplies. Good regulation of this power is not necessary, and a brute force regulated supply is more than sufficient. Sixteen gauge wire should be used in the power distribution.

The program is designed to have the motor connect directly to a Variac, or equivalent. The control output is limited to 180 degrees maximum rotation for this reason. If the 85 oz-in torque of the motor is insufficient to rotate the shaft, gearing can be added, but this will require a program change.

Because there is no feedback from the stepping motor it is necessary for the program to rotate the motor far enough in CCW direction to reach a travel limit. After this initialization has been accomplished the program

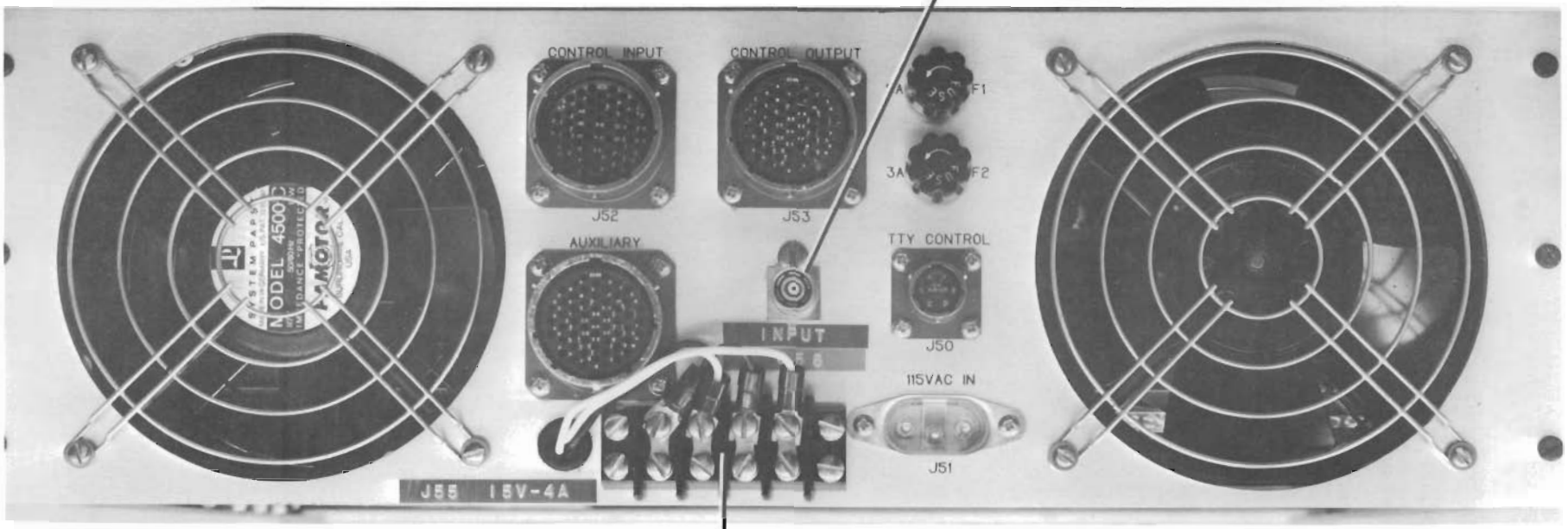


FIGURE 5. Input/output connectors

maintains the motors position by counting all output pulses, until heat is to be shutoff following the braze melt. At this time initialization is repeated to assure no accumulating error. A limit switch is sensed by the program to prevent excessive rotation of the Variac.

OPERATING INSTRUCTIONS

Normal Operation

When power is first applied to the controller it will be necessary to restart the unit by pressing the switch on the front panel. This will also initialize the control program and return the stepping motor to home position, so that in the event of an emergency automatic control can be shut off by pushing the restart switch.

Usually power will be on and the program running. Under these conditions the operator need only be concerned with the pushbutton switches on the remote control box. A brazing cycle is initiated by pressing the START switch. This commands the controller to raise the temperature up to the brazing point at a preset rate (see Adjustments section for discussion of this and other control settings). Once at the brazing temperature the controller will hold indefinitely at this temperature until the operator signals that melt has occurred by pressing the MELT switch. This causes the controller to continue holding for a pre-set time period, after which the heat is shut completely off.

Adjustments

Controller adjustments are located inside of the enclosure towards the front on the right hand side (see Fig. 6). Adjustments include three gain control potentiometers for the feedback compensation and two data entry switches for profile control. The parameters determined by the later two switches set the rate of rise (in minutes to reach brazing temperature) and the hold time following the braze melt (in tens of seconds) before heat shut-off. The hold time may *not* be set at zero, as the program will treat this situation incorrectly.

Control Response Adjustments

The gain constants associated with each of the three different control actions: proportional, integral, and differential, are set by the previously mentioned potentiometers. For satisfactory operation these controls must be set correctly, or either under-control or unstable response will result. Many references in the literature describe techniques

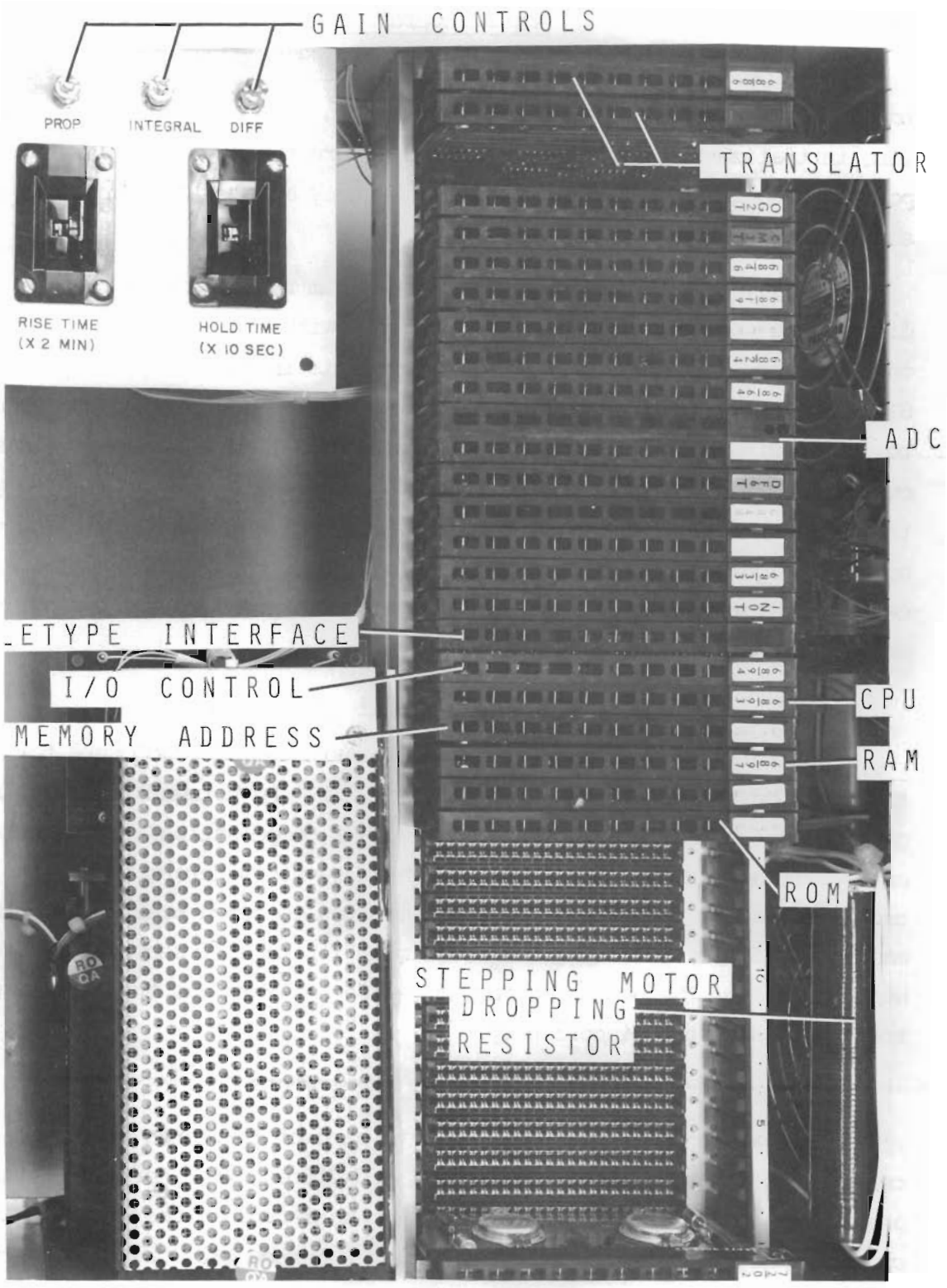


FIGURE 6. Controller internal view.

for setting these controls, but most assume detailed characterization of the controlled process. The method given here anticipates the absence of this knowledge, and only requires a strip chart recorder attached to the temperature feedback signal (full scale should be 50 mv).

One note of caution: the response of any particular brazing setup will be affected by the degree of coupling between the heating coil and the part. The control response should take this into account by being set on the conservative side. Otherwise a test part with adequate coupling may control well when a production part with poorer coupling is partially unstable. Strip chart records of the temperature profile should be maintained for the first few parts following an adjustment for this reason.

Proportional Gain. The bulk of the control response will undoubtedly be proportional, that is the control output (the amount of heat being applied to the part) will be proportional to the difference between the desired profile temperature and the actual (measured) temperature. The important thing to note is that there must be some temperature error to produce any proportional response. Thus for any temperature other than ambient with proportional response only, a steady state temperature offset will exist. Increasing proportional gain will decrease the amount of the offset, but can *never* fully eliminate it. At some point further increases in proportional gain will cause the system to go unstable and oscillate (this is called the ultimate proportional gain). A reasonable proportional gain setting is half this value. Fig. 7 represents the effects of increasing proportional gain. The third adjustment is the ultimate gain setting, and the process is in continuous oscillation at this point.

Integral Gain. The offset error that accompanies proportional-only control can be eliminated by the introduction of integral control. The control output in this mode is proportional to the integral of the control error. Thus even a small offset will integrate over some time period to a sizable control response. In fact, any degree of integral control has infinite gain for dc control errors. The integral gain merely determines the rate at which integral accumulates.

The infinite dc gain of integral (also known as reset) control allows it to reduce the steady state error to zero, but the 90 degree phase lag it adds tends to make the closed loop system more unstable. For this reason it should be used sparingly. After the proportional gain is set and the steady state error is known, the integral gain can be increased to observe its effects. This is illustrated by Fig. 8, where the cycle was allowed to reach the holding temperature before the integral gain was adjusted. This allows for a constant *setpoint* temperature, thus avoiding the introduction of another variable parameter during the adjustment. When the integral gain is first increased it will begin integrating the offset error which presently exists. This will increase the control output, causing the temperature to increase. When the temperature reaches the setpoint of the profile the error has gone to zero, but the integral may have accumulated more than necessary. To reduce the excessive integral the temperature must go *above* the profile to integrate in the opposite direction. The net effect of this is the damped oscillation indicated by Fig. 8. Further increases of integral gain will make the oscillations continuous and, then, unstable. The ideal integral gain (or integration rate) will match the response of the process such that little or no overshoot results from this adjustment.

Differential Gain. This control mode responds to the differential of the control error. Its primary use is to optimize the dynamic performance of the controller to fast changing setpoints and processes. This application does not require such performance so its use may not be necessary. Certainly until more operating experience is gained it should not be used because interaction of a third parameter makes adjustment more difficult. However, because differential control has a 90 degree phase lead it can, to some degree, counteract the instability of the integral control's phase lag, if this is necessary (it probably will not be).

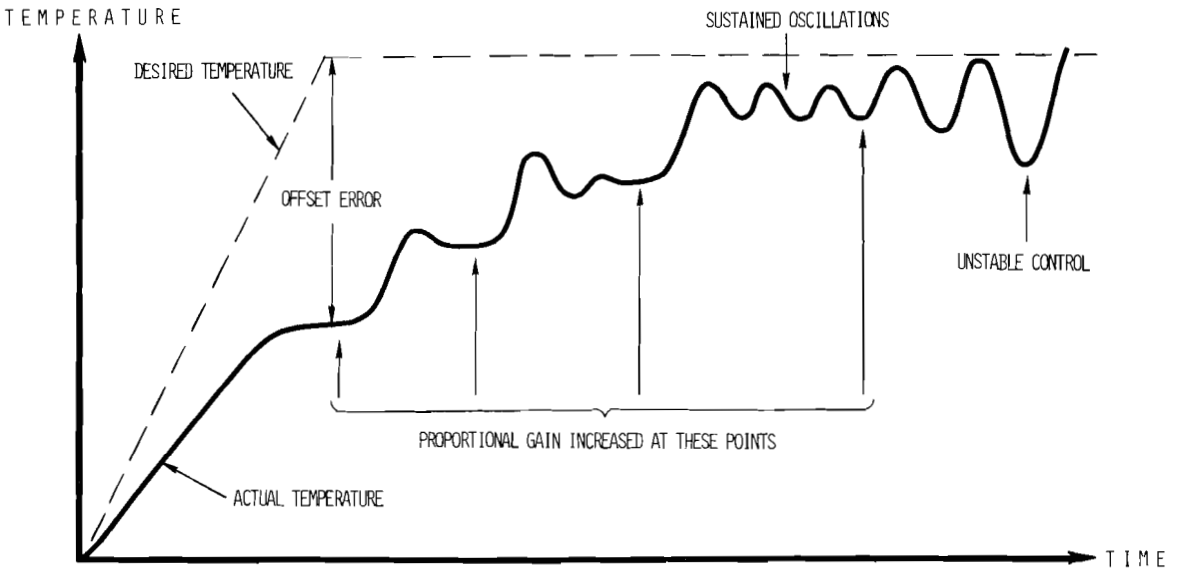


FIGURE 7. Effects of various proportional gain settings.

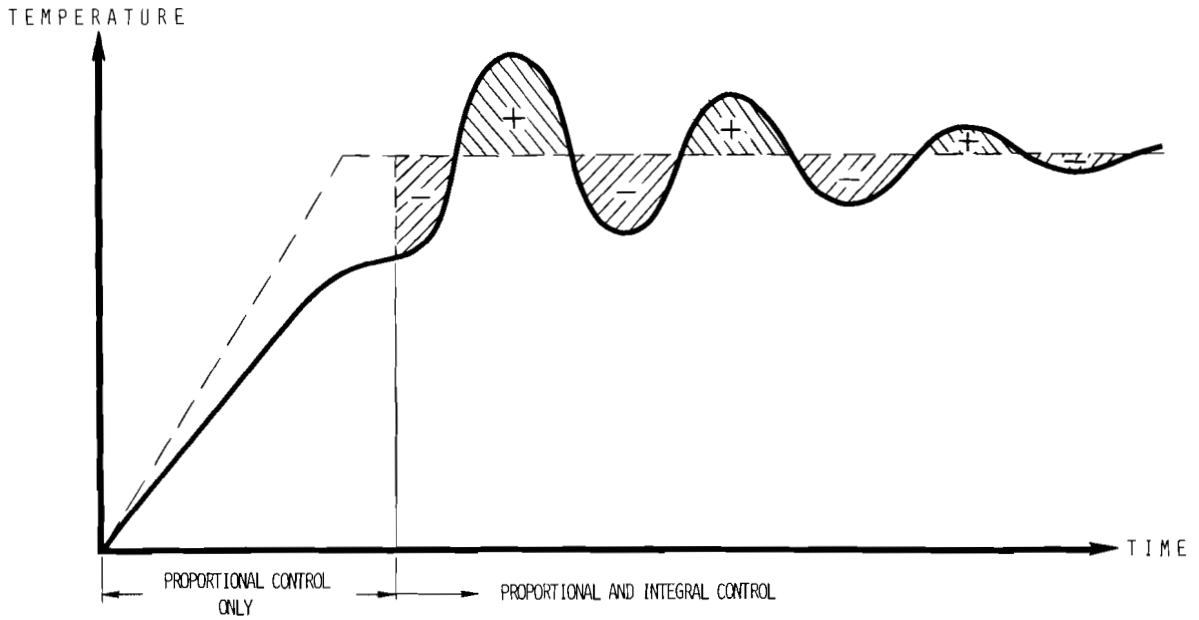


FIGURE 8. Adjusting integral control.

MAINTENANCE

The only periodic preventive maintenance which the controller requires is occasional oiling of the cooling fans and analog-to-digital converter calibration. All other maintenance will involve either "tuning" the control parameters (detailed in the Operating Instructions) or repairing a failure. Because the controller is considerably more sophisticated than it first appears a comprehensive set of procedures for verifying operating and detecting failures is included.

Digital Process Simulator

The immediate concern in the event of a malfunction is the positive isolation of the source of the difficulty. For this and other reasons a digital process simulation was designed and fabricated which permits most of the controller functions to be checked out independent of the other components in the system. While not intended to be foolproof, this technique can identify most controller failures.

Connection of the simulator is made through a cable card connector which plugs into card position 37 (Fig. 9) accessed through the top cover. It is necessary to turn the controller power off and disconnect the temperature input cable (J56). *The simulator should never be connected or disconnected with controller power on.* If it is necessary to disable the stepping motor for any reason, do so by turning off the translator power to terminal strip J55, and *not* by disconnecting only the motor.

The analog output (corresponding to temperature) of the simulator is made available by the two terminal posts on the simulator's box (Fig. 10), and varies between 0 and +10 volts. The best method for observing this signal is with a strip chart recorder.

Test Programs

The interface circuitry can be largely checked by utilizing the processor and a Teletype. This is done by using the Octal Debug Technique program (ODT) to enter short test programs into random access memory (RAM) from the Teletype and execute them. Fig. 11 shows the controller connected to a video terminal in place of a Teletype, along with instrumentation used during calibration.

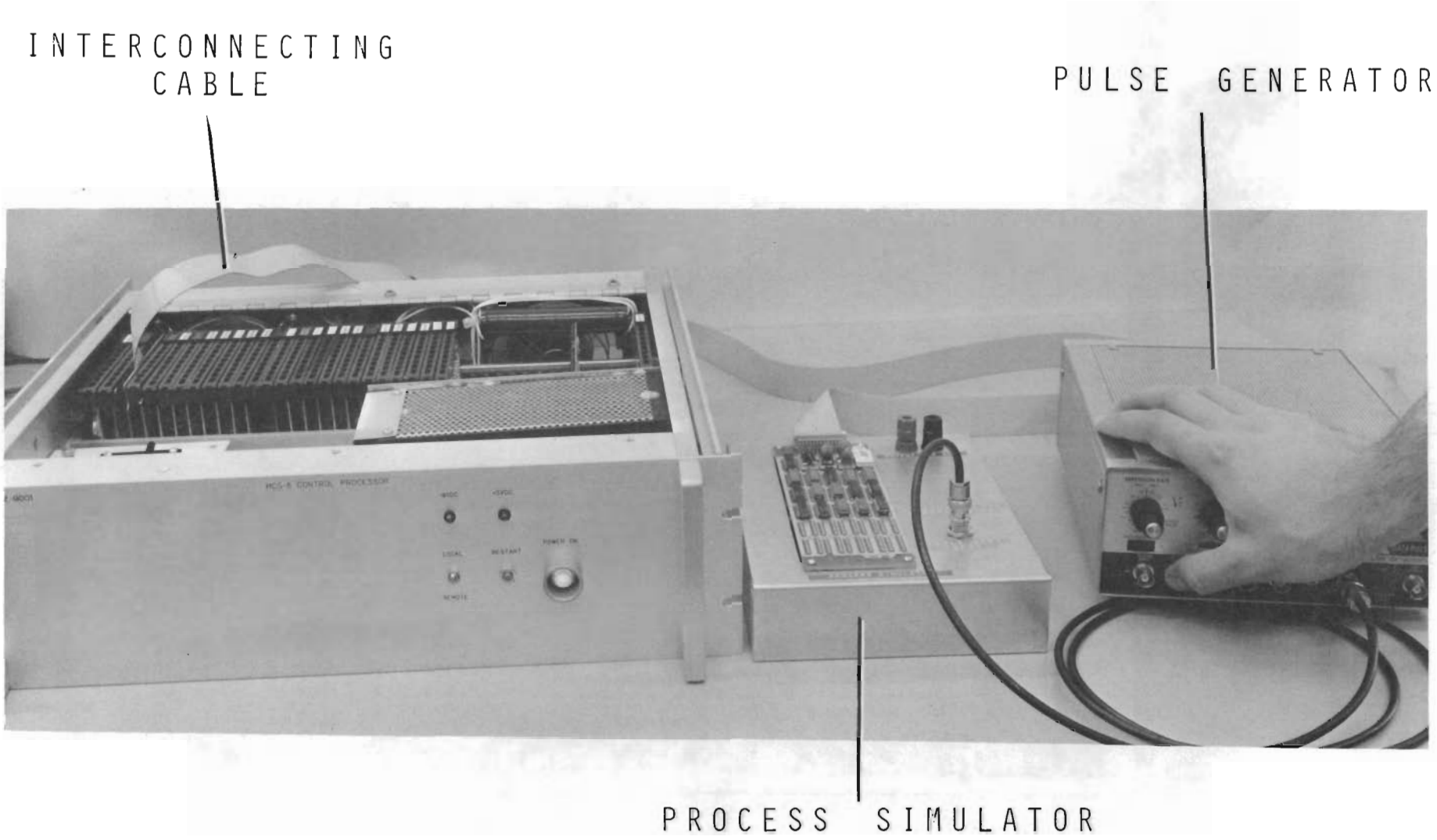


FIGURE 9. Process simulator connected to the controller.

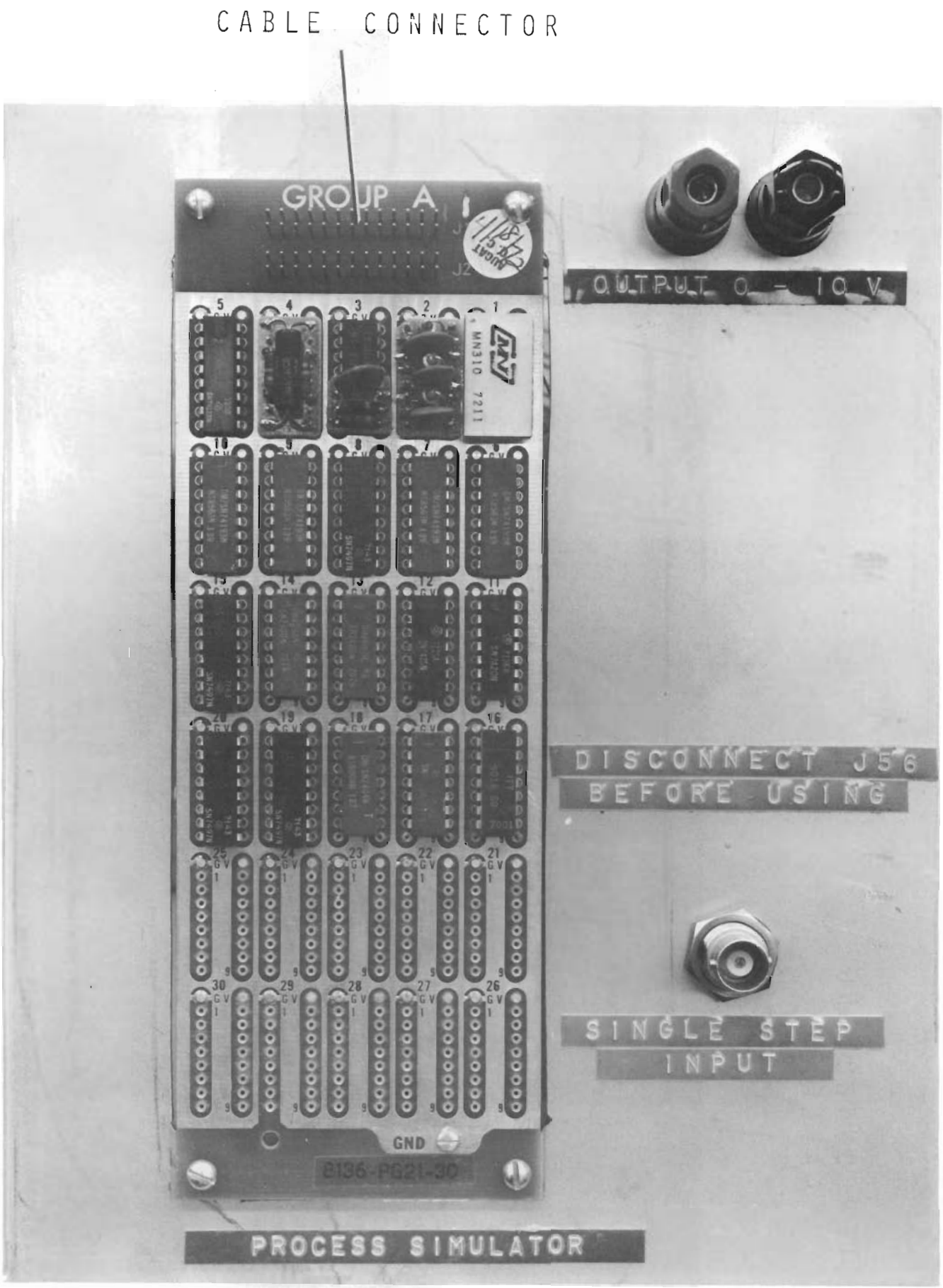


FIGURE 10. Process simulator.

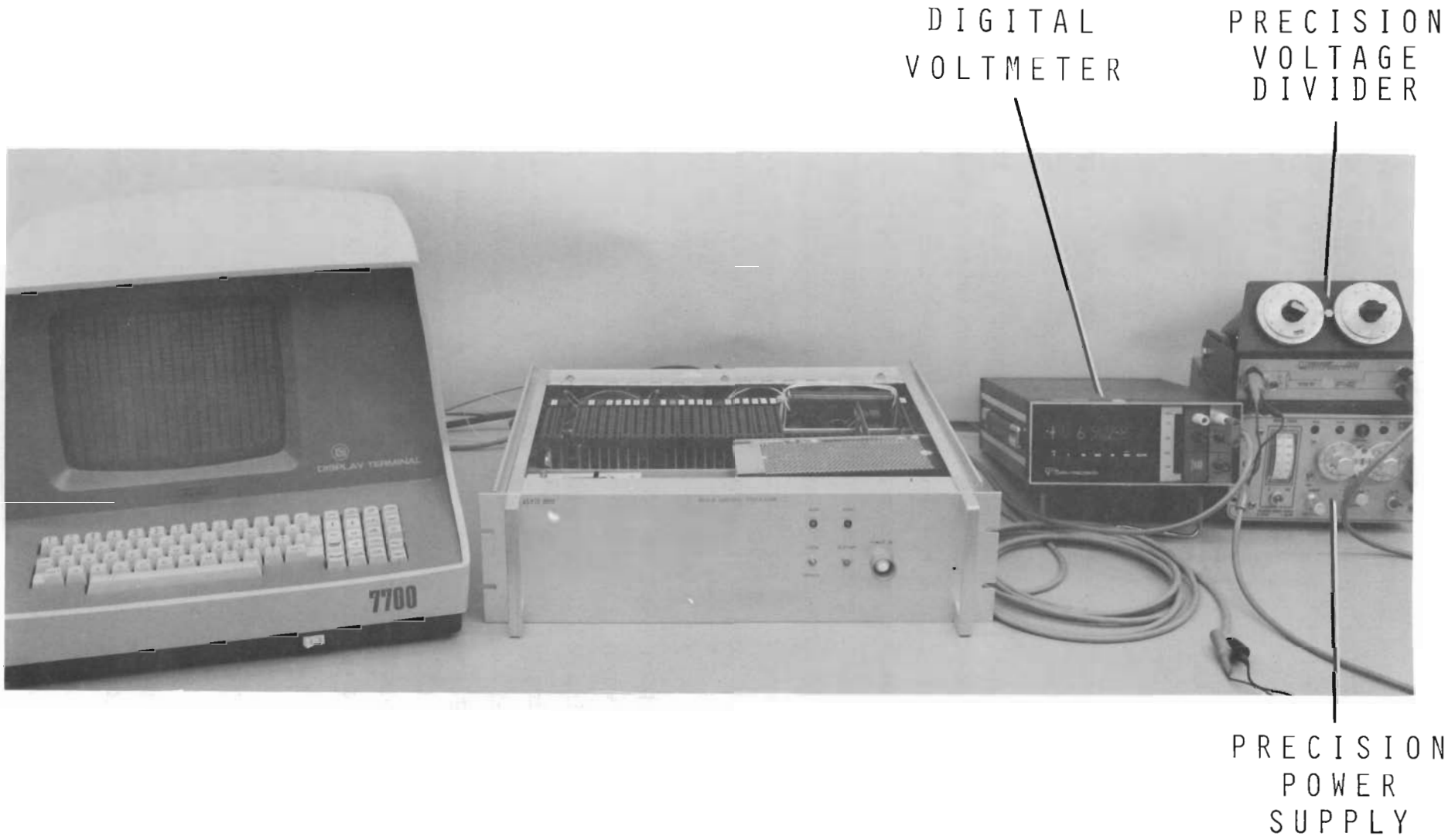


FIGURE 11. Controller setup for maintenance and calibration.

Testing of the input facilities is eased by use of the octal output routine, OCTALP, contained in the ODT program itself. This routine types a space and the value of the A register as a three digit octal number. It is executed by any subroutine call, such as

```
106     CAL OCTALP
307
000.
```

The BCD and control switches, for instance, are examined merely by inputting their setting and typing it out in octal.

Exercising the MUX-ADC is a little more involved. It requires setting the multiplexer and commanding a conversion. A BCD switch is conveniently used to select the input line (only 0 through 3 are used) by reading its value and outputting it to the multiplexer control register. The real-time clock is tested simultaneously with the ADC by waiting for it to set before commanding a conversion and resetting the clock flip-flop. A Teletype, however, is not fast enough to keep up with the clock rate (10 Hz), but the real time clock must be operational for any output to occur.

The stepping motor has no feedback which can be examined, but can be tested in two ways. First the motor can be single stepped in both directions by pushing "A" for CCW and "B" for CW rotation. It may be convenient to use a masking tape "pointer" on the motor shaft as a single step (1.6 degrees) is difficult to observe. A typical motor failure will involve a single winding (usually the power driver is the component which goes). In this situation the motor may continue to rotate, but in an erratic fashion. Single stepping will reveal this as two steps in one direction, one reversed, and one with no motion at all (four steps are required to complete a full sequence). The motor still turns because the net sum of motion will be in the commanded direction, but at one fourth the correct amount. Speed is also limited so that it may stall when operated at normal speeds.

Continuous motion is best checked with the motor disconnected from the load. Friction clamping can simulate the load's torque requirements,

but will not usually be necessary. The test program reads the Teletype input character (which is available until another is typed) and outputs it as a stepping command. As before an "A" causes CCW motion and a "B" CW motion, a "D" will stop all motion. The stepping rate, 200 Hz, is fixed by a software delay of slightly less than 5 milliseconds.

Calibration

To assure an accurate correlation between the thermocouple temperature and the internal digital representation the input amplifier and analog-digital converter need periodic calibration of approximately once a year. This task will require the following pieces of equipment (see Fig. 11):

Reference power supply, 0.01%
Precision divider, 0.01%
Digital voltmeter, 0.01%.

The reference voltage is divided by a suitable factor, such as ten to provide an input to the controller from zero to 40 millivolts in 0.1 millivolt steps. The digital voltmeter verifies the correct gain of the combined amplifiers with calibration data in Appendix D.
The analog input test program, with the amplifier input selected, is used to compare converted values with the above calibration data.

Since the amplifier gain is fixed the use of the digital voltmeter is an optional double check. Small variations from tabulated values can be compensated for by the analog-digital converted (ADC). Larger variations indicated a failure or a drift in a component value. In this case the particular amplifier at fault should be identified (the first amplifier has a gain of 10 and the second a gain of 25) and repaired.

The ADC has two adjustments, one for zero and one for full scale trimming (see LED 69-901184). These adjustments will affect only a few digits change in the data listed in the calibration table (only eight of the unit's twelve bits are tabulated because this is all that is used in the control program). Adjustments which are greater than this range cannot be made by these potentiometers and indicate a gross malfunction which should be repaired. Any adjustments which are made should utilize

the full twelve data bits to afford the longest term calibration (full scale of 42.38 ms can be assumed equal to 377.740_8 for this purpose).

Oscilloscope Waveforms

The general status digital and program logic can be determined from a few waveforms. Any oscilloscope with a dual trace feature can check this.

The real-time clock will always be running except for one second after a restart. This signal, at pin 26-R, is triggered by a ten Hertz clock on pin 26-N (see Fig. 12). The processor continually samples this signal, and once it is triggered the processor resets it (pin 26-L).

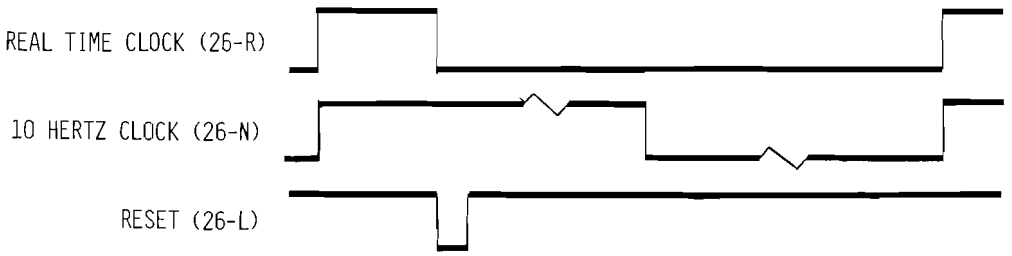


FIGURE 12. Real Time Clock Waveforms.

Much information about what the processor is doing and when is available from the input multiplexer control signals. To synchronize these signals with a particular point in the program the Real Time Clock signal (26-R) is used to trigger the oscilloscope. The signals in Fig. 13 are typical for a cycle after the push of the START button. About seven milliseconds after the resetting of the Real Time Clock the reset signal will pulse three times. This is the triggering of the ADC for each of the three control parameters for use in calculating the output response. Between 20 to 25 milliseconds after the start of the cycle output pulses to the stepping motor will appear (if there is a change in the control output). These pulses may be CW (38-B) or CCW (38-A) during any cycle, but will never be both. The pulses will be separated by 5 millisecond intervals, and can never exceed 175 total (an unlikely number, with a dozen or less being more typical). Once all output steps have been commanded the processor will resume sampling the Real Time Clock with

the status input signal (23-3). Since this signal is absent during the calculations the execution time during any cycle can be observed from it.

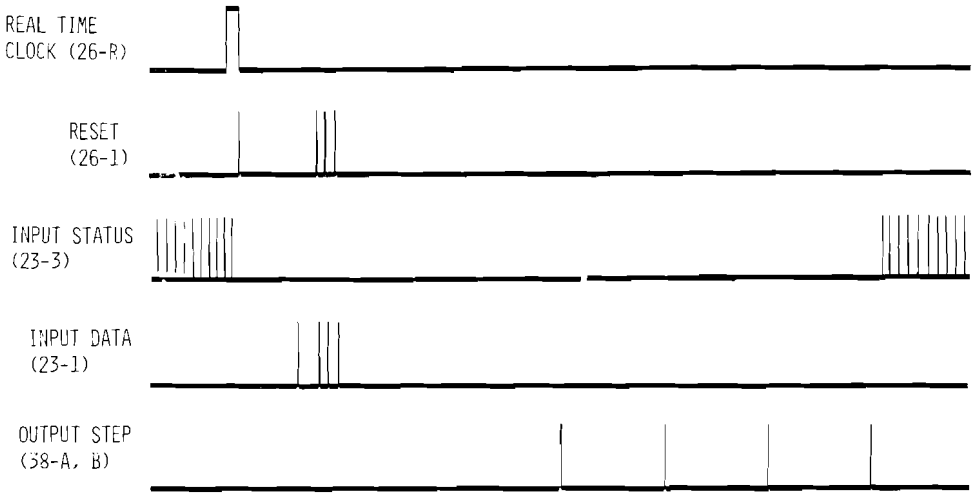


FIGURE 13. Data Multiplexer Waveforms.

Maintaining the Processor

Much of the previous maintenance depends upon the processor being operational so that at least the ODT program can be run. In the event that the processor is not operational procedures are needed for diagnosing and repairing the malfunction.

One simple technique is to replace all cards associated with the processor one at a time until the facility card is isolated. The nine cards in positions 18 through 26 are those which must be exchanged. This technique will only work if the failure is on one of these nine cards. If it is on another card, or if it is in the wiring (it is possible for the corner of a wire wrap pin to cut through the insulation of a wire held against it with excessive tension), this technique will only indicate that the problem is *not* on these cards, which is valuable information in itself.

While the particular problem may dictate the nature of the problem it is wise to ascertain the following items

1. Both power supplies, +5 and -9 volts, are within tolerance.

2. The two phase clocks are running and in tolerance (see MCS-8 manual or LEA 68-9068-94D)

3. The Sync signal (19-W) is running after pushing restart

4. The Ready line (19-19) is low.

Single Stepping - If the following procedures do not locate the problem it will be necessary to examine operation on a step-by-step basis. The processor can be single stepped by pulsing the $\overline{\text{READY}}$ line (19-19) with a low-true pulse. For this purpose the wire to ground was removed and wired to an inverter at 22-V. With the input disconnected the output will be low and the processor will run normally. With a pulse generator connected to the input on the digital simulator which generates a positive 3 to 5 volt 10 microsecond pulse the processor will execute one instruction cycle for each pulse (see Fig. 9 for the setup of this equipment). If the generator is set for single pulses the processor can be easily single stepped through a program. Also the program can be executed at the repitition rate of the generator. The effect of this is best observed by operating the ODT program at a slow speed with an operational processor.

Single stepping and slow speed execution by themselves are of little value without address and data display. LLL has available a card which plugs into one of the memory slots which displays the address and memory output data with light emitting diodes (LED's). But to be useful this information must be properly interpreted, for which a program listing and an MCS-8 manual are useful.

It must be remembered that the address information serves a multiple purpose. Not only is it used for the instruction address, but it also is used for I-O instruction decode and accumulator output data. Further for multiple cycle instructions data and address fetches may be required. The breakdown of each instruction on a sub-cycle basis is tabulated in the MCS-8 User's Manual. An instruction set summary is on the inside rear cover.

ODT - If the ODT program does not respond to input usually either the asynchronous receiver/transmitter is not functioning (card 21) or the input

multiplexer is not properly inputting status (card 23). If data available (21-2) do not go high after pushing a Teletype key check that serial data are actually being sent by examining both 21-W and 21-19 (refer to LEA 68-9068-99C). If the ODT program does not properly recognize the character check to see that the clock on 21-14 is 1760 Hz. If the clock rate is right, one of the data lines, either between the UART and the multiplexer or the multiplexer and processor, has been damaged. ODT may recognize some characters and not others, which will point to a particular data bit (for instance an open line will have the effect of a high signal, which will work for these characters that agree in this bit position).

Memory - If ODT, the processor, and the interface equipment function properly but the temperature control program fails in some manner the memory may be at fault. Memory consists of both random access (RAM) and programmable read-only (PROM) memory, so each must be checked separately.

RAM is checked easiest by replacing it (be sure that the three jumpers of the replacement are identical to the original). If this does not solve the problem, yet the RAM is known to be the problem (as can be determined by a test program which must be entered in *another* RAM), the next step is to check the address bits and the read-write control line. The later must pulse high to write data, but also must stay low at all other times.

The PROM presents a more difficult situation, but has also demonstrated to be more reliable from the admittedly small sample size with which we have worked. The difficulty is that spare PROM's written with the control program (it takes four) may not be available for replacement. Assuming this to be the case, one of two techniques may be used to check the PROM out.

If a computer is available with a synchronous interface of suitable speed (at least 600 baud) a program in the host computer can do a word-for-word compare of the PROM to a program tape using ODT. Such a program has not been developed for this project, but one designed to link the MCS-8 to an SEL 840A could be readily adapted for this purpose.

If this is not practical, the slow speed execution technique may be used to detect hard program failures ('hard' is used to describe those changes in the program which cause a complete stoppage of execution, such as a jump to an unused page of memory). By restarting the program and allowing it to approach the point of failure slowly the approximate address of the failure can be determined. Although it is only necessary to determine the page in which the memory change is located the exact location can be deduced by close examination with ODT.

A 'soft' program failure is more difficult to identify since full operation is not defeated. Further it may be erratic since only certain number sizes may be affected, for example. The best bet here is to start execution and stop it periodically to examine the program variables in RAM with ODT. The program must be manually restarted at 4415_8 to avoid program initialization each time. Only experience and a thorough knowledge of the control program make this feasible, however, as these variables are only the effects, and not the causes, of the failure.

SPECIFICATIONS

Dimensions

Controller	7 x 19 x 15 in
Remote control box cable	10 ft
Stepping motor cable	10 ft
Analog input cable	15 ft

Power

A.C.: 120 v @
D.C.: 15 v @ 6 a

Analog Input

Bandwidth:	1 KHz
Impedance:	300 MΩ
CMRR:	100 db
Full scale Voltage:	42.38 mv
Connector:	Trompeter PL-72

Stepping Motor

Type:	Superior Electric HS-50
Torque:	85 oz-in
Steps/revolution:	200
Control range:	180 degrees
Mounting:	Any position
Thermal:	55° C rise

Control Program

Control modes:	Proportional, integral, and differential
Digitation accuracy	±0.012%
Sampling rate:	10 Hz
Output resolution	
Nominal:	0.6%
Maximum:	0.0015%
Temperature profile	
Number of points:	256
Accuracy:	±0.012%
Advancement Rate:	Selectable, 0.6 to 5.4 sec/point, 0.6 sec increments
Hold delay:	Selectable, 1 to 9 min.

Teleprinter Interface

Type:	Asynchronous, current loop
Level:	8
Code:	ASCII
Rate:	110 baud

SOFTWARE

As mentioned in the Introduction the largest effort in the controller development was software. Obviously the success or failure of a microprocessor application will depend greatly on the continuity and strength of all associated software. The general approach of using the same system to develop software which it will eventually run on, and relying on vendor supplied system software, will not be viable for microprocessors. For even if the system software exists, the microprocessor, by nature a small dedicated purpose machine, will not be adequate for this use. The software development, then, must be done on a larger system with the necessary hardware and software to support the development.

Support is of several different types. First, facilities must be available to assist in the conversion of a symbolically coded microprocessor program to the machine code of the microprocessor. This will consist of means of preparing the symbolic code, such as on punched cards, editing it, and storing it on a mass storage device such as magnetic tape. This code then must be converted to machine code by what the current programming jargon calls a "cross assembler." The designation "cross" refers to the generation of code for one type of machine on another. The assembler generates both the machine code (termed the "object code") and a program listing with both the machine code and the symbolic or "source" code.

An assembler is a mandatory aid in the software development. The only alternative to an assembler is to "hand" assemble the program. This is a lengthy and error prone procedure for a reasonably sized program, and must be repeated each time the program is changed (which, during development, is often).

The second type of system support is a *hardware simulator* program. The simulator must be capable of performing every function of the microprocessor. It executes the identical program code which the microprocessor executes. It also provides the operator with sophisticated debugging aids unavailable on the microprocessor. The use of the simulator speeds program development by a factor several times the rate using the

microprocessor alone. Further the simulator could be used to check out program changes before they are applied to an actual process. This is done by simulating the process as well as the microprocessor.

It should be stressed that the sole value of a simulator is the information it supplies the designer. For this reason the simulator must be highly interactive with the user, providing all pertinent information and full control of execution. Also, because information can be voluminous, the simulator should be capable of condensing it down to a useful form.

The third type of support involves mating the microprocessor to the larger system. With a data link between the two, the larger computer can assist in such tasks as program loading, program debugging, memory checkout, and microprocessor exercising.

Program loading is a typical chore during development. Because re-programming of several ROM's can be time consuming, much time can be lost making simple program changes. If RAM is used during this stage instead of ROM this time can be cut drastically. The key to the method is using the data link to pass the program code from the large computer to the microprocessor.

The data link itself is the RS232C serial asynchronous convention which is relatively commonplace. The controller is equipped with such an interface, but is designed for a Teletype at 110 baud (data elements per second). Each character requires eleven bits, which sets the character rate at ten per second. This is not satisfactory for an inter-computer link, since both devices can operate at a much higher rate, which necessitates modification of the interface for higher speeds (a straight-forward modification).

The host computer can be used for debugging by capturing information during program execution. The host will have facilities for storing, condensing, and outputting this information. Exercising the microprocessor can be written to return control to the host computer after execution. However the data link is used, for program loading or controller checkout,

there must exist a service routine in the controller to support its end of the data link. Such a program has been written for this purpose.

Octal Debug Technique

The octal debug technique (ODT) allows the operator to examine and change memory from a terminal such as a Teletype. Lawrence Livermore Lab's document LER 72-103402 describes the use of ODT for the purpose. The program given in the document has been modified in minor ways. Primarily the program loading command was altered to a more efficient binary format.

The host computer uses ODT by minimizing the operator's interaction, but at a much higher speed. Through ODT the host computer can examine and change memory, and transfer control to some other program. The capabilities of the MCS-8 ODT should not be confused with powerful debug programs written for the larger minicomputers. ODT, for practical reasons, is limited to 256 words (one page) of memory.

The program starts by typing a question mark and waiting for input. As each character is typed it is examined as to its type: command terminator, or digit. Legal octal digits are accumulated to form a number up to 255, anything greater is an error. The number typed is interpreted by ODT when a terminating command is entered, such as G for go. For example, 377G.

The terminator is decoded by successively subtracting the numerical difference between legal terminating characters. For instance, the difference between a line feed and a carriage return is 3. The program first subtracts octal 212 (the line feed code) from the terminator. If the result is not zero the character was not a line feed. It then subtracts 3 (for a total of octal 215, the carriage return code) and again tests for zero. In like manner all other legal terminators are tested.

Memory addressing must be done in two separate steps. The memory page is set by entering the page number in octal followed by S. The ODT page is selected by typing

ØS,

and the RAM page with

1ØS.

It must be remembered that the assembled address, as outputted in the program listing, is a complete octal number which is not separated by page numbers. Consequently page $1Ø_8$, word Ø is equivalent to address $4ØØØ_8$.

Once the page has been set a particular location within that page can be examined by typing the octal word number (0 to 377_8), followed by a slask. ODT will respond by typing the contents of that location. If this location is in RAM the contents can be changed by typing the desired quantity followed by a line feed or carriage return. The contents are unchanged if no number is typed, or the location is a part of ROM.

A line feed automatically displays the next sequential location in the selected page. This command uses an output routine which converts an eight bit octal number to three octal digits preceded by a space. First typed is the page number, followed by the word number, completed with the contents of that address typed as three octal digits. It is up to the operator to interpret this octal number as either instruction code of program data. As with the slash terminator, the contents can be altered if a one to three digit octal number is typed by the operator.

Control can be transferred to another program from ODT by entering the starting address followed by G. The page number, as before, must have been set by the S command. ODT constructs a jump command in the last three words of page 10_8 (375, 376, and 377). For this reason these locations should not be used except as temporaries. It also requires a page of RAM at this page number.

A "bootstrap loader" is executed by the R command. It is not intended to be operated from the Teletype in as all data is in binary format (the host computer takes over at this point, having itself issued the "R" to begin program load). The program is loaded into RAM in contiguous pages, whether or not the program uses all of the pages for code. Load parameters consist of the starting page and the number of pages. The starting page is typed in octal preceding the R command. The first data

following the R are the number of pages to be loaded, beginning at word Ø of the starting page. The number of pages and the program code is in binary.

MCS-8 Instruction Set

The MCS-8 instruction set is divided into five classes:

I. Index register instructions
II. Accumulator group instructions
III. Program counter and stack control instructions
IV. Input-output instructions
V. Halt instruction.

Index register instructions allow for the loading of registers from memory and other registers (a NOP is performed when an index register is loaded from itself). Also any register, except for A, can be incremented or decremented by one.

Accumulator group instructions use the A register as one of the operands for an arithmetic or logical operation. The source of the second operand, which can be an index register, memory, or data in the program code (immediate data), subdivides the accumulator group class three ways. A carry flip-flop provides the mechanism for propogating carries during multiple precision arithmetic operations.

In the third class of instructions are program jumps, and subroutine calls and returns. This class allows use of conditional testing where the programmer can specify which of the status flip-flops must be true or false for the instruction to be executed. While being quite useful it is also the only way in which the status flip-flops can be examined. Because the programmer has no access to the return address during a subroutine call, arguments must be passed via the general registers or predetermined memory locations.

Input-output instructions are the sole source of communication with the external world. Both use the A register for the destination and source of the transferred data, respectively.

The halt instruction is, literally, in a class by itself. Because of the dedicated type of application to which microprocessors will be put,

this instruction is not expected to be normally used in program code. It can catch the gross error of a jump to a memory address with no actual memory (the structure of the memory generates the code of a halt instruction if no valid decode occurs).

A serious deficiency of the MCS-8 instruction set is the lack of provisions for signed arithmetic. For instance there is no direct means to detect a signed overflow. If A and B are the operands and R is the condition for overflow is

$$\overline{(A_S + B)} \cdot (A_S + R_S)$$

Clearly this calculation by software means would be costly, so much so that it is as fast to do double precision arithmetic and not check overflow. This, in fact, is what the control program does. Enough dynamic range is allocated to contain a worst case set of inputs. The output of the program is limited to the maximum range of the controlled variable after the calculations have been done.

Control Program

The control program executes a discrete version of the three mode control algorithm and a first order digital filter. The term three mode refers to the three types of responses available with the controller: proportional, integral, and differential. The time response of this controller is

$$U(t) = k_1 e(t) + k_2 \int_0^t e(t)\, dt + k_3\, de(t)/dt.$$

It also controls analog conversion, checks control limits, and generates a temperature profile. This code occupies four 256 word memory pages, with the profile requiring one of those pages.

Temperature profile data are stored in the table as eight bit unsigned integers, of the format

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

This permits a resolution of 1 part in 256, or about $\pm$0.0125%. The value, or weight, assigned to the least significant bit is not in units of temperature and must be converted by multiplying by a constant. A correlation to temperature requires two conversions; first the temperature--voltage function of the thermocouple must be known, and second the voltage--digital factor of the analog-digital converter must be known. For a cromel-alumel thermocouple a temperature of 810° C produces a voltage of 32.91 mv corrected to an ambient temperature of 20° C. For this voltage the output of the converter is 305.34_8 (see Appendix for an ADC calibration data and a temperature-digital conversion table). This calculates to be approximately 4.09° C least significant bit (LSB) weight.

The control program is not aware of this factor. It merely compares converted input data to profile data, both of which are in the same units and need not be converted any further. The difference between the table entry and the measured temperature is known as the control error, e(t). The control response calculation is based on this error.

Prior to making this calculation, however, the temperature data are operated on by a recursive digital filter. This filter has two purposes: first it reduces any signal noise that may be present, and secondly it minimizes the truncation effect of digitization (quantization). Both of these problems tend to be very disruptive to the derivative action. Even if noise were not a problem, quantization, or truncation error, would be. Quantization error is illustrated by Fig. 14, where continuous signals must be approximated both by sampling and by discrete levels. Although the continuous signal can usually be approximated adequately for most applications by quantizing, the computation of the derivative may not. If only two data are used it is easy enough to show that for any frequency the sampling rate can be increased to the point where the derivative is either one or zero. The addition of noise at this point greater than the quantization level will mean that the signal to noise ratio will be less than one, a disastrous control situation. The use of programmed filters, however, can circumvent this problem.

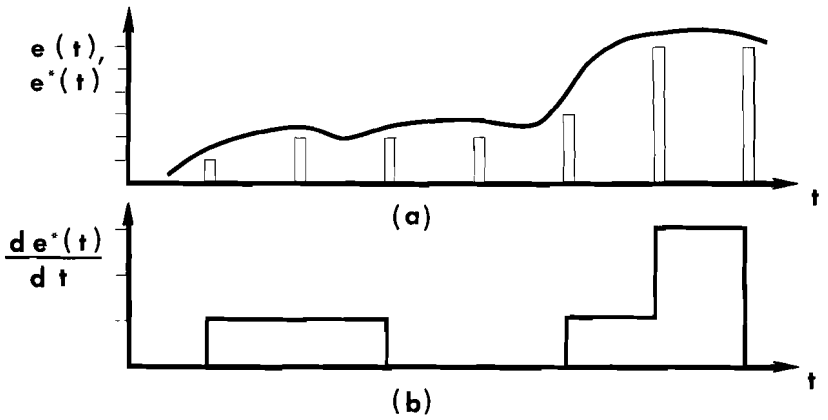


FIGURE 14. Quantization error. (a) Approximation of a continuous signal. (b) Discrete derivative.

Digital Filter

If the input to the filter is X and the output X', the output at step n is

$$X'n = aX_n + bX'_{n-1}.$$

The fact that X' appears on both sides of the equation makes it recursive, that is the result of any successive calculation depends upon all previous calculations. The output, X'_n, can be written as an infinite series

$$X'_n = a[X_n + bX_{n-1} + b^2X_{n-2} + \ldots + b^k X_{n-k}]$$

It is evident from this that X'_n is a function of all previous inputs, X_n. If the difference is taken between two successive outputs, X'_n and X'_{n-1} (as the derivative does), the result will also contain differences between all successive inputs, as can be seen from

$$X'_n - X'_{n-1} = a[(X_n - X_{n-1}) + b(X_{n-1} - X_{n-2}) + \ldots + b^k(X_{n-k} - X_{n-k-1})]$$

Control Algorithm

The error thus calculated is used as the input to the three mode control algorithm. The control response derived by this algorithm is calculated as follows:

$$U_n = P\, e_n + I \sum_{i=0}^{n} e_i + D\,(e_n - e_{n-1})$$

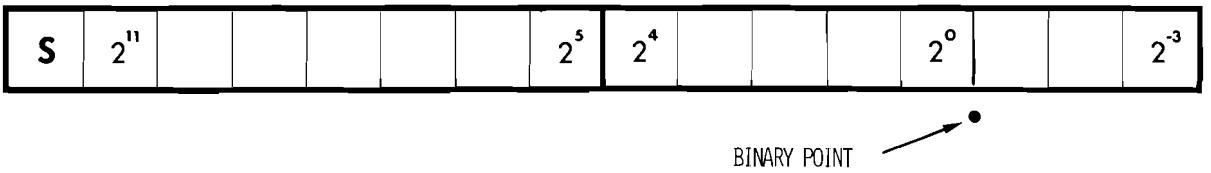
where U_n is the control output at sampling interval n
e_n is the control error at sampling interval n
P is the proportional gain
I is the integral gain
D is the differential gain.

The proportional calculation uses the single precision value of the control error (which is scaled back down to regain the original weight). This is multiplied by the proportional gain, measured from a potentiometer, which is also a single precision number. If no further operation is done on the result choices of proportional gain would be limited to integral values. Since this would be insufficient resolution the proportional output is scaled down by a factor of one-sixteenth. This allows a proportional gain range of 0 to 7 15/16 units in 1/16 unit steps (1 part in 127).

The integral control is calculated in two steps: first the current control error is summed to the error integral, and then the integral is multiplied by the integral gain. If the integral were allowed to accumulate at 10 summations per second, the sampling rate of the controller, the integral would rapidly saturate if a significant error were present (even though the integral is maintained in double precision). Prescaling the error prior to summation is undesirable in as this creates a deadband for errors less than the minimum resolution of the scaled error. For instance, say the control error is caled by four, then errors in the range of one to three units (approximately 4°C to 12°C) would appear as zero after scaling. Since this is the type of error which the integral response is to eliminate this is an unacceptable technique. A second approach is

to integrate at a slower rate, such as one-tenth the sampling frequency. This has the effect of scaling the integral by one-tenth without introducing deadband. It does, however, reduce the bandwidth of integral control and increases phase lag at frequencies near and greater than the summation rate. This is accomplished in the program via a timeout variable which is initialized to ten and is decremented at each calculation loop. Summation occurs only if it is zero, otherwise the control response is calculated from the previous integral without summing the current error. To do this the single precision equivalent is found from the double precision integrand, which is multiplied by the integral gain and then scaled down for the same reasons outlined in the proportional response discussion.

Differential response is found by differencing two successive control errors, and multiplying the difference by the differential gain. To avoid the truncation problem mentioned earlier the scaled double precision error is used in the difference calculation. These data have the two word format



It is mandatory, however, that the multiplication routine be entered with two single precision numbers. Thus if the greater resolution (2^{-3}) is to be maintained some magnitude restrictions on the error *difference* must be made. What this means is that the largest positive error difference recordable is 7 5/8 units and the negative bound is -8 units. This would only happen if the set point changed by an amount greater than this in two successive steps, which it does not. As before, the double precision value generated by multiplication is scaled down.

After all three control responses have been calculated a signed double precision value represents the control variable. This must be translated to some physical quantity capable of controlling the process. This is the purpose of the stepping motor, which, when connected to some suitable device as a Variac, can regulate the energy inputted to the process. Some limitations are applied to the control variable. First

only positive energy is meaningful so all negative control values are limited to zero. Secondly, the Variac has a travel limit which must not be exceeded. Since the motor is directly coupled to the Variac shaft, and the motor makes one revolution with 200 steps, maximum clockwise actuation is limited to 185 steps (310 degrees). The program keeps the motor position equal to the control variable by stepping it in the appropriate direction (clockwise for positive differences and counterclockwise for negative) one step for each unit difference up to the travel limit (0 and 175). Because there is no position feedback from the stepping motor the controller keeps track of it by storing its current position in memory. Further, the motor has certain dynamic restrictions which limit the maximum synchronous stepping rate. A delay variable in the output stepping program loop sets this rate at 200 Hz.

Sequential Control

The remainder of the control program is related to sequence control. Facets of the program which come under this category are program initialization, cycle start, temperature ramping, braze melt, and cycle completion. The decision logic of this part of the program is regulated by a table controlled routine known as a *finite state automation* (FSA). Process control is exercised by which of the several states the FSA is in. It advances from one state to the next on the basis of control inputs and the particular state it happens to be in at the time. The FSA for the control program has four states, which have the following functions:

1. Initialization - Reset stepping motor position, input data switches and initialize other control variables.
2. Wait - Wait for the start of a cycle.
3. Cycle - Advance process through a temperature cycle.
4. Melt - Hold process at plateau temperature for specified delay, then return to initialization.

Inputs to the FSA include the two control pushbuttons and an internally generated advance and reset command. The decision of what the next state the process will transfer to is made by a table containing all the possible

combinations of possible states and inputs (4 states x 4 inputs = 16 entries). An example is if the FSA is in state 2 and the Start button is pressed the FSA is advanced to state 3. If the Melt pushbutton is pressed in state 2 the next state is also state 2. In effect the Melt pushbutton is ignored at this time. If it is pushed when the FSA is in state 3, however, the FSA will change to state 4.

The switches are scanned at the sampling rate, 10 times per second, to test for activation. If one is pushed the FSA is executed to determine the next state, otherwise the current state is decoded in order to execute the appropriate routine. The other two inputs are generated by the program itself, as in the initialization routine which is to be executed only once. It causes the FSA to advance to state 2, the Wait state. The other instance is in the Melt state following the hold delay, at which time the FSA is to be triggered into the Initialization state to shut power off and re-initialize.

The Cycle state executes the three mode control algorithm, and advances a pointer through the temperature profile table. The table contains 256 entries, which are sequenced at a rate of 0.6 second times the rise time setting per entry. For instance, if the switch is set at one (for a two minute rise time) the pointer will be incremented once every six program loops. The pointer will reach the two hundredth entry after 120 seconds, or two minutes. The profile stored in the table is set such that the brazing temperature is reached at this entry. The remaining 56 entries also are this value, which corresponds to the temperature plateau. If the cycle is not completed by the end of the table the program automatically halts advancing the pointer at the last entry.

The Melt routine decrements the hold time out, which is the setting of the hold delay switch times 60 seconds. Since the loop is executed at 10 Hz the setting is multiplied by 600. The hold delay switch is read during initialization, so if it is changed *after* initialization the processor must be restarted to reflect that change for the next cycle. All the time the hold delay is being executed the three mode control algorithm is being executed to maintain temperature control right up to the moment when the power is shut off.

THEORY OF OPERATION

The hardware discussion is separated into two sections. The first section deals with the processor itself and compares it with the MCS-4. Explained are both the internal and external organization of the MCS-8. The second section goes into the interface equipment and its theory of operation.

Intel Micro Computer Set

Intel pioneered the advent of large scale integrated (LSI) digital processors with the MCS-4 (which stands for Micro Computer Set) and the MCS-8. The MCS-4 is a four bit parallel organized processor, with a total of 45 instructions. It has a maximum program size of 4096 words. The MCS-8 is an eight bit parallel processor with a maximum memory addressing of 16,384 words. It has 48 instruction set repertoire. The MCS-4 has a 10.8 μsec instruction cycle, compared to 20 μsec for the MCS-8.

MCS-4

The MCS-4 communicates with external circuitry over a four bit data path. This bus is shared for both input and output transfers. Four data transfers consisting of more than four bits the transfer must be done during several subcycles, with a four bit "slice" being transferred during each subcycle. For instance, program addresses are twelve bits and require three subcycles to complete the transfer.

Most computers store their program code in read/write memory. It is becoming increasingly more common to find well used programs stored in read only memory. This assures that the program is not accidently altered or erased. In the situation where the read/write memory is *volatile* (the contents are lost if the power is shut off) the use of ROM is especially useful. The MCS-4 has gone beyond this, requiring all program code to be stored in ROM. The processor distinguishes between the writeable memory (RAM) and the program store (ROM). For programs which are being changed regularly, as they are during development, this can be an awkward arrangement.

The MCS-4 has the simplified internal organization of the block diagram in Fig. 15. A four level, twelve bit stack stores all program addresses, the top of the stack being the address of the current instruction (the program counter). Subroutine calls cause the specified address to be "pushed" onto the stack. Returning from a subroutine reverses this process, "popping" the stack up one level. The four level stack permits subroutine nesting of three levels. The address is incremented as it is gated out onto the data bus four bits at a time. The incremented four bit slices are rewritten into the stack register after each of the first three subcycles.

Internal working registers consist of 16 four bit index registers. One of these is used as the accumulator during arithmetic operations. There is also a carry flip-flop, which can be used to propogate carries during multiple precision arithmetic or shift operations.

A series of special purpose instructions enable the addressing of the read-write storage. This cannot be considered as a general memory array, addressable by a single operation. Each RAM has four sections, each with two types of storage - four "status" words (or four bits each) and sixteen other storage locations. Four *separate* instructions permit the accessing of the status words. The general locations, however, must be selected by first setting up their location in two of the index registers (by a two word instruction), outputting these registers by a second instruction, and finally reading the location with a third instruction. And then the RAM must be in the currently active "bank" (a bank consists of 4 RAM's). The same procedure must be used to write data into the RAM's. Use of the read/write storage is artfully complex.

Constants can be accessed from the ROM memory in a considerably more straight forward manner. Since each ROM consists of 256 words, eight bits are required to address all locations. This is done indirectly by a pair of index registers which are initialized prior to reading. The ROM is selected by virtue of being the one in which the program is currently executing. Of course this means that any ROM cannot be totally filled with program constants, but must have some program instruction code besides.

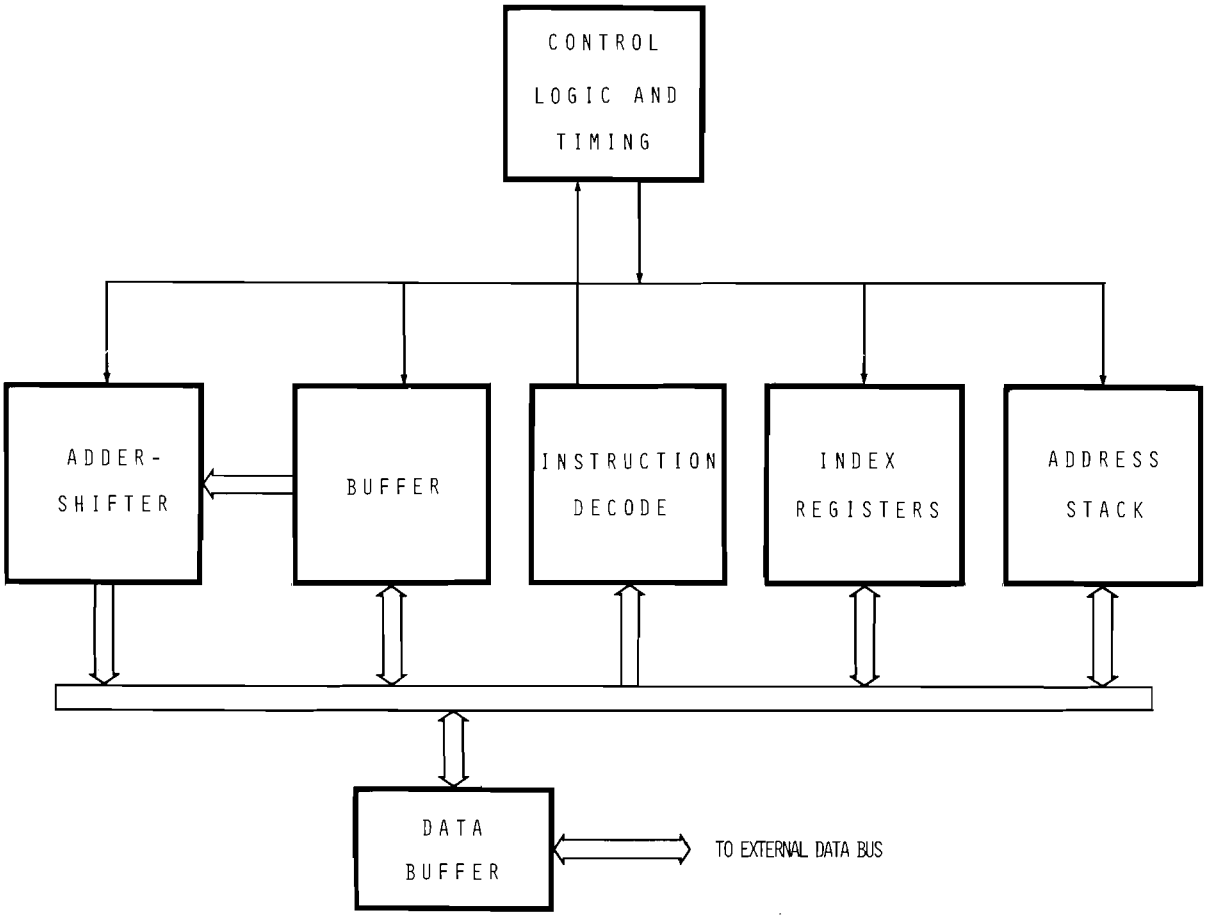


FIGURE 15. 4004 CPU block Diagram

MCS-8

Conservation of chip area and package interconnections dictated the use of a single eight bit data bus (see Fig. 16). The 48 instructions are divided into subcycles to share control of the data bus. But because all data transfers must take place on this bus a considerable number of subcycles are required for each instruction. Take for instance memory addresses which must select one of 16,384 locations. This corresponds to 14 bits, six more than can be transferred at one time. Thus two subcycles are used to transfer just memory addresses.

Addresses are used for referencing two types of memory contents--instruction code and program data. While program data can be fetched or stored, instruction code can only be fetched, and is stored in a register within the instruction decode logic. Program data are stored in one of seven general registers (referred to as A, B, C, D, E, H, and L). One of the registers, A, is used as an accumulator. The A register is used as one operand in all arithmetic and logical instructions, and is also the destination of the result.

Instructions cannot access memory directly. That is, no data addresses, in any form, are combined with any instruction. Instead, memory is referenced indirectly using registers H and L. Two registers are required because 14 address bits must be assembled for each memory access. The L register contains the least significant eight bits, and the six most significant bits are in the H register.

Program addresses are maintained in an eight register stack of 14 bits each. One of these is the program counter, which is incremented after each instruction fetch. As with the MCS-4, subroutine calls and returns are linked by saving the program counter in one of the other seven registers. A three bit counter designates the location of the program counter (the present address). A subroutine call causes this counter to be incremented and the subroutine address to be stored in that address register. This register is now the program counter and execution continues from this point. A subroutine return reverses this process,

decrementing the stack pointer so that it now uses the previous register as the program counter. An eight level stack permits nesting of subroutines to seven levels.

Some computers use a location in the subroutine to save the return linkage address. For microprocessors this technique poses the problem of having to use writeable storage for the subroutine instead of ROM. Address stacking solves this drawback, but as designed the address stack is not accessible to the program.

The Arithmetic/Logic Unit (ALU) executes arithmetic and logical instructions. It is capable of two's complement addition and subtraction, and can perform AND, inclusive OR, and exclusive OR. Four flip flops store certain conditions resulting from an ALU instruction. These conditions are

- A carry out of the most significant bit
- Whether or not the result was zero
- The sign of the result
- The parity of the result.

Two temporary registers a and b store operands for ALU operations. They are also used for data storage during transfers of data both within and without the processor.

The MCS-8 has eight subcycles as does the MCS-4, but all subcycles are not entered during any particular instruction. The MCS-8 has state control logic (see Fig. 17 for the state diagram) implemented by a five stage feedback shift register. This eliminates unnecessary execution states for multiple word instructions such as JMP but allows longer cycle times for multiple word instructions. Also some instructions require different numbers of cycles if a condition tested was true or false.

A register indicates which instruction cycle a particular instruction is in. The output of this register is one of three possible states: C1, C2, or C3. Note that most transitions of the state diagram include a term from the cycle register. All transitions back to state T1 cause the cycle register to be updated. The operation is discussed in more detail later for typical one, two, and three word instructions.

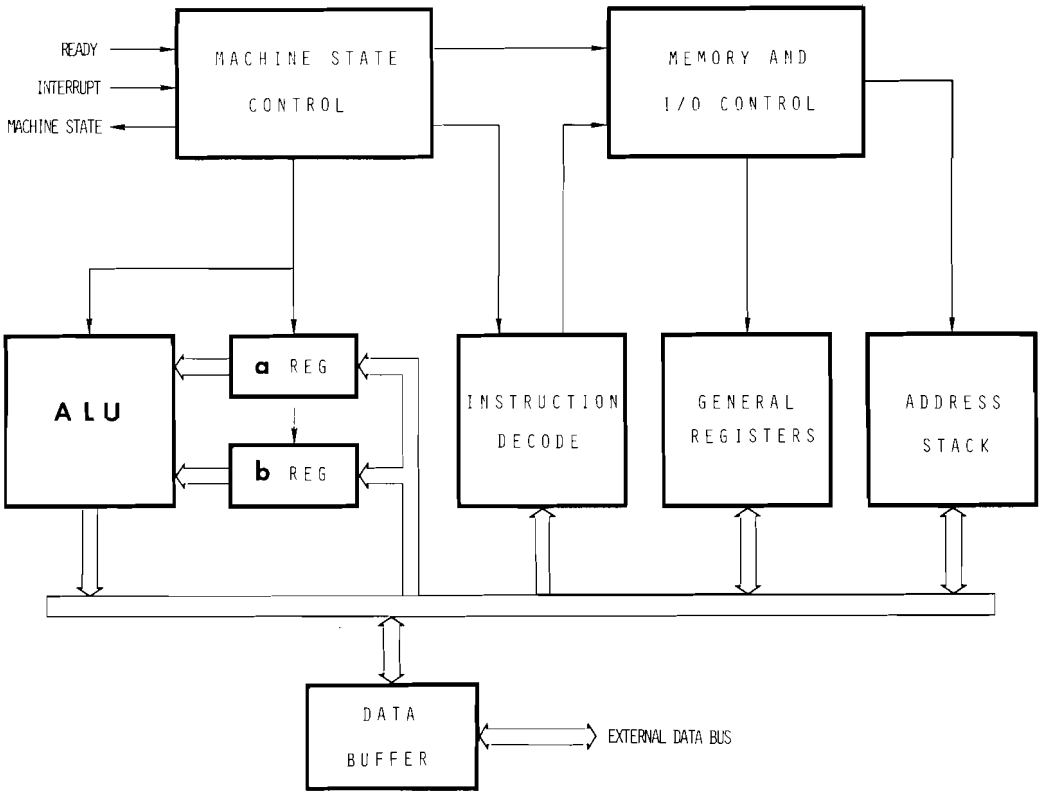


FIGURE 16. 8008 CPU block diagram

The subcycles T1 through T5 are used by the MCS-8 for the following purposes:

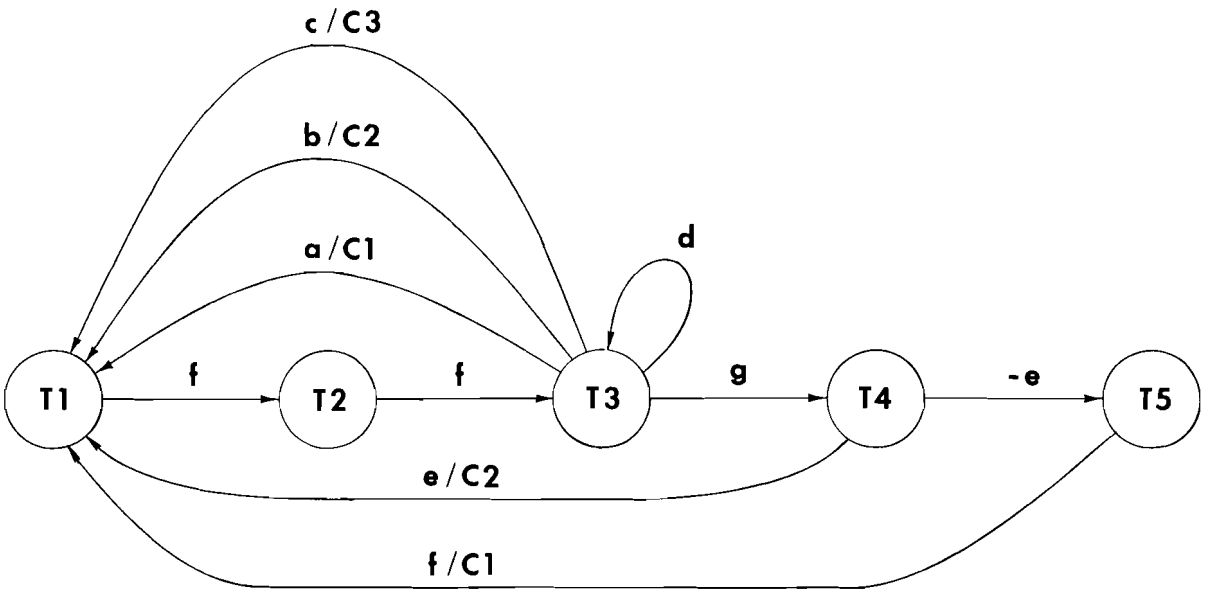
T1 Send lower eight bits of memory address

T2 Send higher six bits of memory address

T3 Memory data fetch

T4 and T5 Instruction execution.

Operation is probably best explained through examples. Consider, first, a single word instruction such as an intra-register data transfer, Lr_1r_2 (load register r_1 with the contents of r_2). Subcycles T1, T2, and T3 are used to send the program counter out and fetch the instruction. At T4 the source register, r_2, is gated onto the internal data bus, and the data bus is strobed into register b (not to be confused with general register B). At T5 register b is transferred, by the internal data bus, to the destination register, r_1. Initially the processor is in state T1 of the



$$a = C1 \cdot (HLT \cdot \overline{INT}^{4} + \overline{RETURN}^{1}) + C2 \cdot (\overline{OUT} + \overline{LMr}) + C3 \cdot (\overline{LMI} + \overline{JUMP}^{1} + \overline{CALL}^{1})$$

$$b = C1 \cdot (\overline{LrM} + \overline{ALUM}^{2} + \overline{ALUI}^{2} + \overline{INP} + \overline{OUT} + \overline{LrI} + \overline{JUMP} + \overline{CALL})$$

$$c = C2 \cdot (\overline{LMI} + \overline{JUMP} + \overline{CALL})$$

$$d = C1 \cdot (HLT \cdot \overline{\overline{INT}^{4}}) + \overline{\overline{RDY}^{3}}$$

$$e = C1 \cdot (\overline{LMr})$$

$$f = \text{automatic transition.}^{5}$$

$$g = -a \cdot -b \cdot -c \cdot -d$$

FIGURE 17. MCS-8 state diagram

---

1 *These conditions are true only if the condition tested failed, or if no condition was specified.*

2 *These instructions include all immediate and memory reference ALU instructions (add, subtract, AND, OR, compare, exclusive OR).*

3 *This is the ready signal originated external to the CPU.*

4 *This is the interrupt signal originated external to the CPU.*

5 *Transition occurs on next clock and is not dependent on any other condition.*

first instruction cycle (C1 is true, and C2 and C3 are false). The lower address bits are sent out the data bus during this subcycle and the state changes to T2. The higher six bits are transferred at this point (along with two additional bits which signal the memory that an instruction fetch is in process). The state advances to T3 during which the memory data are strobed into the instruction register. This state has five possible output transitions to itself, state T1 and state T4. Equations a, b, and c are not true because the particular instruction, Lr_1r_2, does not appear in any, leaving paths d or g. Because a, b, and c are false g will be equal to the inverse of d. Path d allows the processor to wait for the memory if the processor's subcycle was shorter than the access time of the memory. This is indicated by the Ready line being true. In this case the state changes to T4, where the source register is transferred to the temporary register. Because equation e is false state T5 is entered, where the temporary register is transferred to the destination register. Following this control changes to state T1. The cycle register is set to the first instruction cycle, C1, even though it is still in the first cycle. This completes the instruction execution and the processor is ready to execute the next instruction.

A typical two word instruction is add the contents of the accumulator with the data immediately following (add immediate, ADI). Two memory fetches are required, one to get the instruction and the other to get the data. The ADI instruction falls in the ALU immediate class, ALUI, which bypasses the unnecessary execution phase, T4 and T5, of the first instruction cycle. This occurs by transition path b at state T3. During the first cycle C1 is true, and ALUI is true. This transition changes the cycle register to the second cycle, making C2 true and C1 false. The second cycle fetches the word following the instruction, which will be added to the A register. This time path g is followed to state T4 and, hence, to T5. T4 and T5 add and store the result back into the A register. As before, the transition from T5 to T1 resets the cycle to C1, completing instruction execution.

All three word instructions are either program jumps or subroutine calls, which include the full memory address of the operation. Two successive words contain the address, the lower order eight bits are stored in the first word and the higher order six bits in the second. Both instructions have the option of specifying a condition which must be met before the instruction is executed. Otherwise the next sequential instruction is executed. The conditions tested are status flip-flops set according to the result of a previous ALU instruction, and indicate whether the result generated a high order carry, was zero, had even parity, or if the most significant bit was true. Execution begins by fetching the instruction. T4 and T5 are avoided through path b, and the second cycle begins. The lower order address of the jump or call is fetched next, and is stored in register b. Transition path c sets the third cycle of the instruction and initiates fetching of the higher part of the jump or call address. States T4 and T5 are used to set the high and low part of the program counter to the new address, respectively. If a condition was specified and failed, the jump or call is defeated through path a of the third cycle.

A halt instruction, or a low level on the Ready input, causes a continued looping at state T3 by path d. The MCS-8 cannot actually stop since the contents of its dynamic memories would be lost. Instead T3 is used to refresh these memories. The Ready line coming true again will initiate execution, but only an interrupt can bring the MCS-8 out of a halted condition.

The interrupt signal (INT), except in the halted condition, is recognized in state T1 of the first cycle. The presence of the interrupt signal inhibits normal incrementing of the program counter. This allows the substitution of an interrupt instruction in place of the normal memory instruction. This is not done by the 8008 and must be built into the peripheral MCS-8 logic.

The processor signals the external logic that the interrupt has been recognized. At state T3 the external multiplexer selects a hardwired interrupt instruction instead of the memory. The instruction used may be any in the instruction set, but is usually the Restart instruction. A Restart

instruction is used because it is a single word subroutine call in page zero (multiple word instructions can be used, but a penalty is paid by the additional multiplexing required to implement it).

While this forms the skeleton of an interrupt structure the MCS-8 is not capable of a true interrupt. The reason for this deficiency is that memory referencing can only be done through the H and L general registers. It is not possible to save registers after an interrupt without losing the contents of at least one of them. The one alternative is to dedicate one or two registers for interrupt temporary storage so the H and L registers can be saved while the interrupt storage is being set up. If only one page (256 words) is referenced by *all* programs only the L register need be saved. It should hardly be necessary to emphasize the reluctance with which the loss of one or two general registers (out of seven, two of which must be dedicated to memory referencing) is treated. The only other alternative is to build special purpose logic to handle this situation, which is even less desirable than the first alternative. Besides this, status-flip-flops can be tested only through condition jumps or subroutine calls, making the saving and restoring of status cumbersome.

A side-by-side comparison of the MCS-4 and the MCS-8 demonstrates the MCS-8 to be considerably more powerful, as one would expect. Besides the obvious advantage of a larger word size the MCS-8 has a more flexible memory referencing scheme. Constants in the MCS-8 can, for the most part, be accessed through immediate data instructions. The MCS-4 has to fetch immediate data with a separate instruction. The eight level address stack of the MCS-8 has immense advantage over the four level stack of the MCS-4, since this is an absolute limit to subroutine nesting. While the MCS-4 must commit its program code to ROM, the MCS-8 can intermix ROM and RAM for this purpose. This permits checked out code to be stored in a non-volatile ROM, while new programs are being debugged in RAM. The allocation of memory is left solely up to the designer.

It is perhaps simpler to point out the MCS-4's few advantages over the MCS-8 than vice versa. The MCS-8 requires considerably more peripheral logic than does the MCS-4, when the MCS-4 is used with its custom designed

ROM and RAM, the 4001 and 4002, respectively. The 4001 and 4002 require no peripheral circuitry allowing the minimum system package count and, hence, cost. Also it has a few decimal arithmetic instructions that MCS-8 does not, but has none of the logical instructions (such as logical AND and inclusive OR) that the MCS-8 has.

MCS-8 Basic System

Fig. 18 shows the minimum logic required to form an operational MCS-8 system. The bi-directional data bus accommodates instruction and I/O input data as well as memory and I/O output data, but buffering is necessary for TTL compatibility. The 8008 requires a two-phase, non-overlapping dock with the timing diagram of Fig. 19.

The basic processor is contained on three 3 1/2 by 4 inch printed circuit cards, which exclude most of the memory and the data multiplexers. These cards and their Livermore stock numbers are

Central processor	LEA68-9068-93
Input-output control	LEA68-9068-94
Memory address	LEA68-9068-95

The processor card, besides the Intel 8008 CPU, contains the data bus input and output buffers, and the state decode. The state of the 8008 is encoded on three output signals from the 8008, which is decoded to eight separate signals for the input-output control card. This card generates all the necessary control signals for the remainder of the logic, for instance signaling the input multiplexer when to gate memory output data onto the 8008 data lines. This card also contains the two phase clock generator, and its trimming adjustments. The memory address card has a sixteen bit register for storing the current memory address sent during two subcycles. It also contains one 256 word PROM.

Memory for the processor is on two card types, one for semiconductor RAM (LEA68-9068-97) and another for PROM (LEA68-9068-96). The RAM card has storage for 256 words, while the PROM card has a total of 512. Both types of semiconductors internally decode the least significant eight bits of the memory address, with the remaining six bits being decoded by a

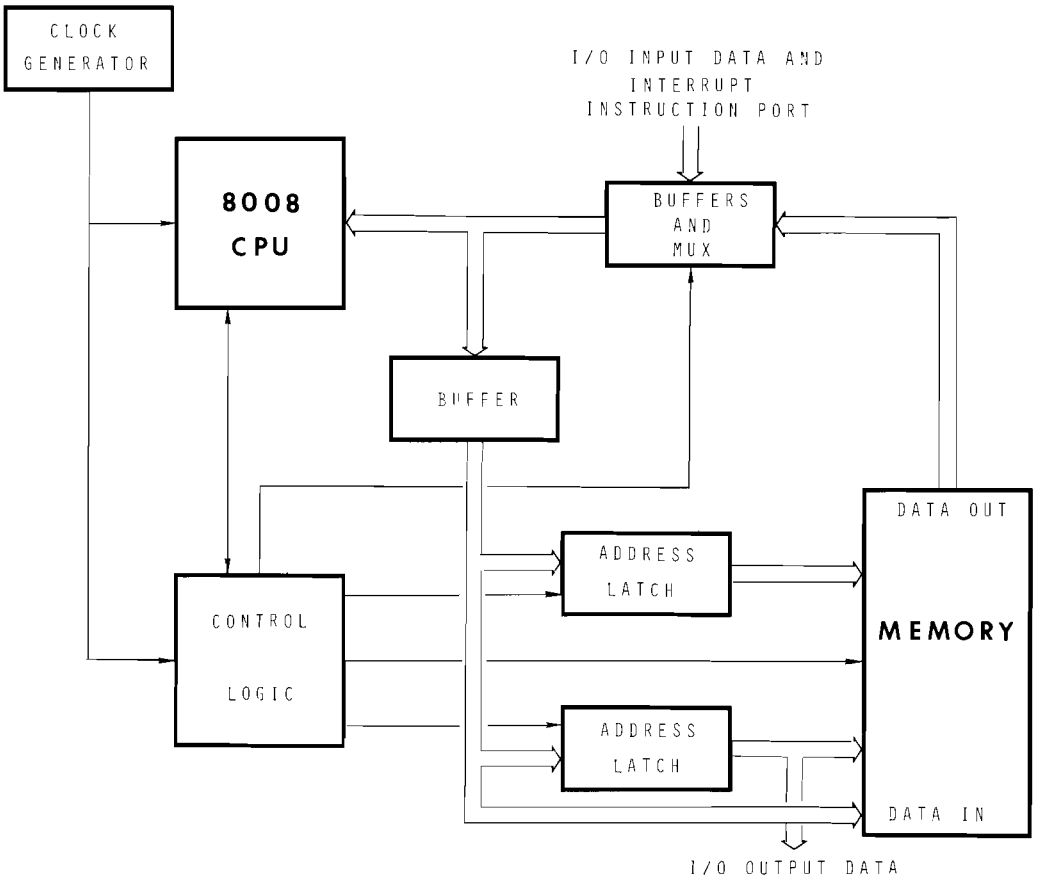


FIGURE 18. MCS-8 system block diagram

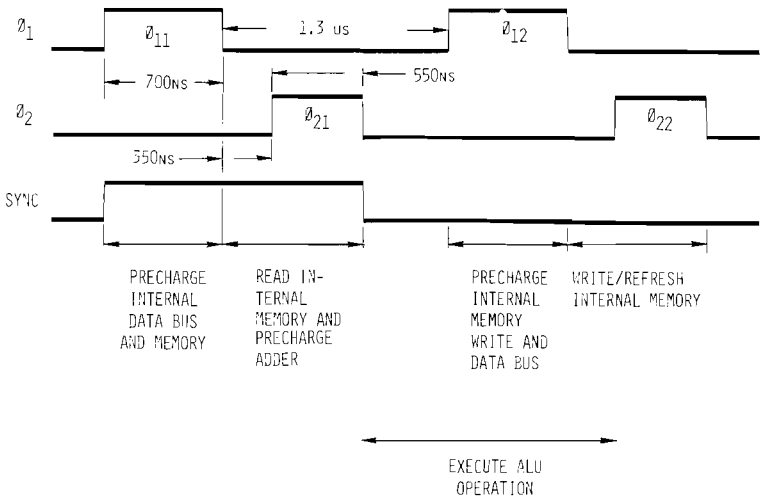


FIGURE 19. Clock timing diagram

combination of two jumpers and a one-of-sixteen decoder IC. Sixteen card positions are dedicated for memory use, with all interconnected by the bussing of fourteen address lines, eight data input lines, eight data output lines, and one control line. Since decoding is done completely on the memory card, the card may be inserted in any of the sixteen slots.

Interface Logic

The controller must be capable of accurately measuring millivolt signals, provided multi-phase current drive for a stepping motor, sense switch and potentiometer settings, and provide the correct asynchronous serial pulses to communicate to a teletypewriter. All of these functions fall under the category of the interface logic, although they may bear little resemblance to each other. This equipment is outlined by the block diagram of Fig. 20. Except for the serial interface this logic is Battelle-designed, utilizing unused card slots in the processor. Schematics of this part of the processor is in Appendix BNWL

The input circuitry must transform a 50 millivolt peak signal to a minimum eight bit digital number, and be as immune as possible to process electrical noise. The input signal is carried to the processor on a shielded twisted pair cable, connected by a triaxial shielded connector (manufactured by Trompeter Electronics). This signal is far too small for the analog-digital converter, which requires a 10 volt full-scale input, necessitating amplification by a factor of 189. The output of the amplifier is connected to a multiplexer (LEA68-9068-64) as are the gain control potentiometers. This multiplexer is controlled by a register loaded under program control, and its output is the input of the analog-digital converter (LEA68-9068-89). This converter is capable of twelve bit accuracy, although only eight are used. The eight most significant bits are wired to one of the ports of the data multiplexer made up of two four-bit multiplexers (LEA68-9068-33). The four least significant bits are transferred to the processor with the use of a multiplexer expander (LEA68-9068-37) since the unused bits of the supplied multiplexer were inadequate.

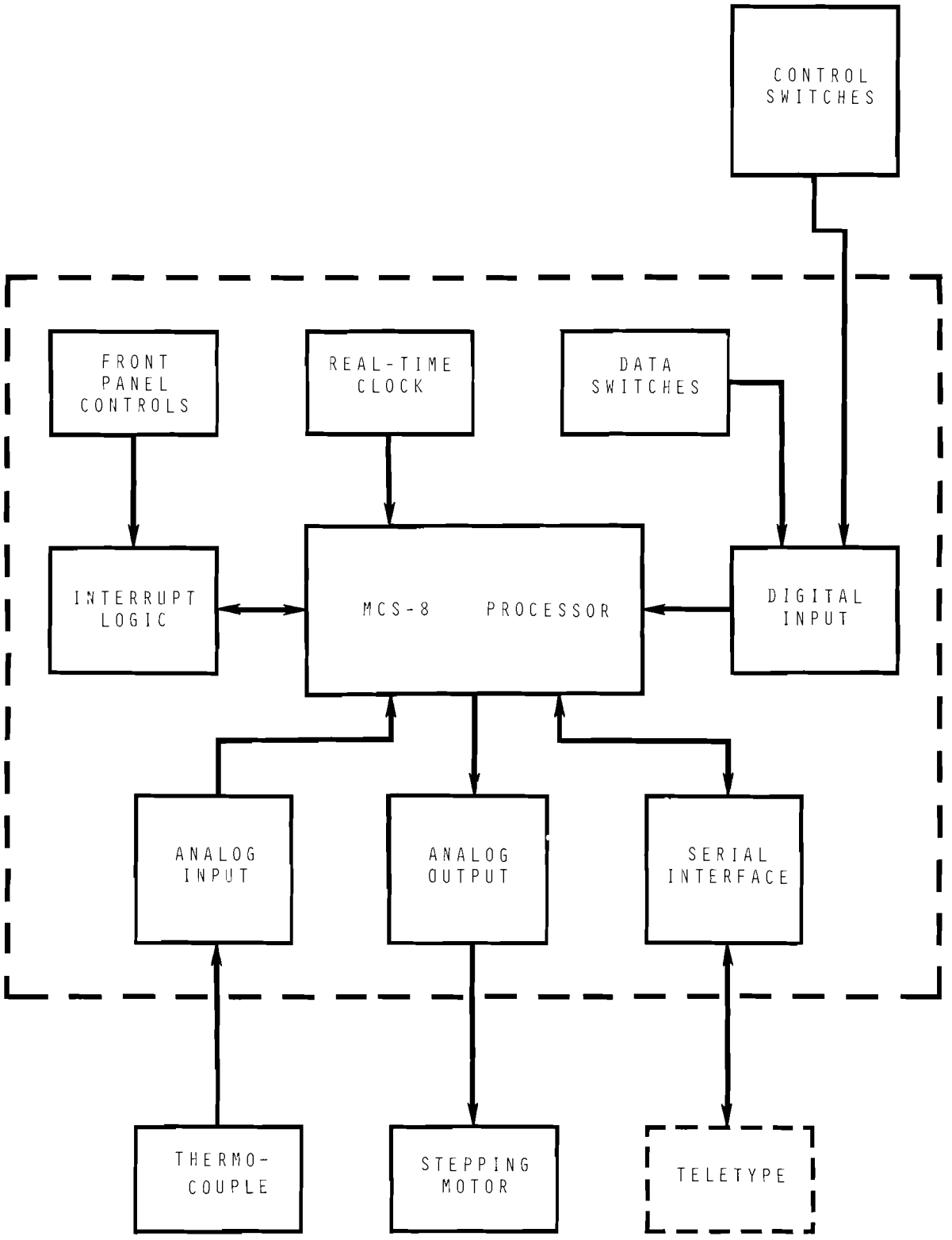


FIGURE 20. Detailed control system block diagram

A stepping motor was chosen as the output device as it offered the greatest number of alternatives for the user. It is a true digital transducer in that a motion command is translated into a discrete motion. Further the error involved in each "step" is nonaccumulative, which means that its long term repeatability is excellent, as is its capability for making accurate long distance movements (which is important if it is connected to a lead screw of a machine for numerical control). The stepping motors driving electronics are also considerably simpler than those for an equivalent dc servo motor. The main disadvantage of the stepping motor, besides its inferior dynamic performance, is that it dissipates as much energy, and, hence, heat, in the standby mode as in the operating mode.

The electronics for the stepping motor, with the exception of the power supply, are entirely self-contained by the controller. This is contained on two p.c. cards (LEA68-9068-85 and LEA68-9068-86) and two power dropping resistors, which are collectively called the stepping motor translator (translate as used in the sense that it "translates" digital stepping commands into rotary motion). The motor is of the variable reluctance type with four windings. Those windings must be activated in a specific sequence to induce motion. Fig. 21 is a simplified version of the translator. The heart of the translator is a two stage counter which generates the correct drive signals for the power drivers. This sequence is specified by the following state transition table. From this table the

Present State	Next State	
A B	Forward	Reverse
0 0	0 1	1 0
0 1	1 1	0 0
1 0	0 0	1 1
1 1	1 0	0 1

sequence for several forward commands can be derived as

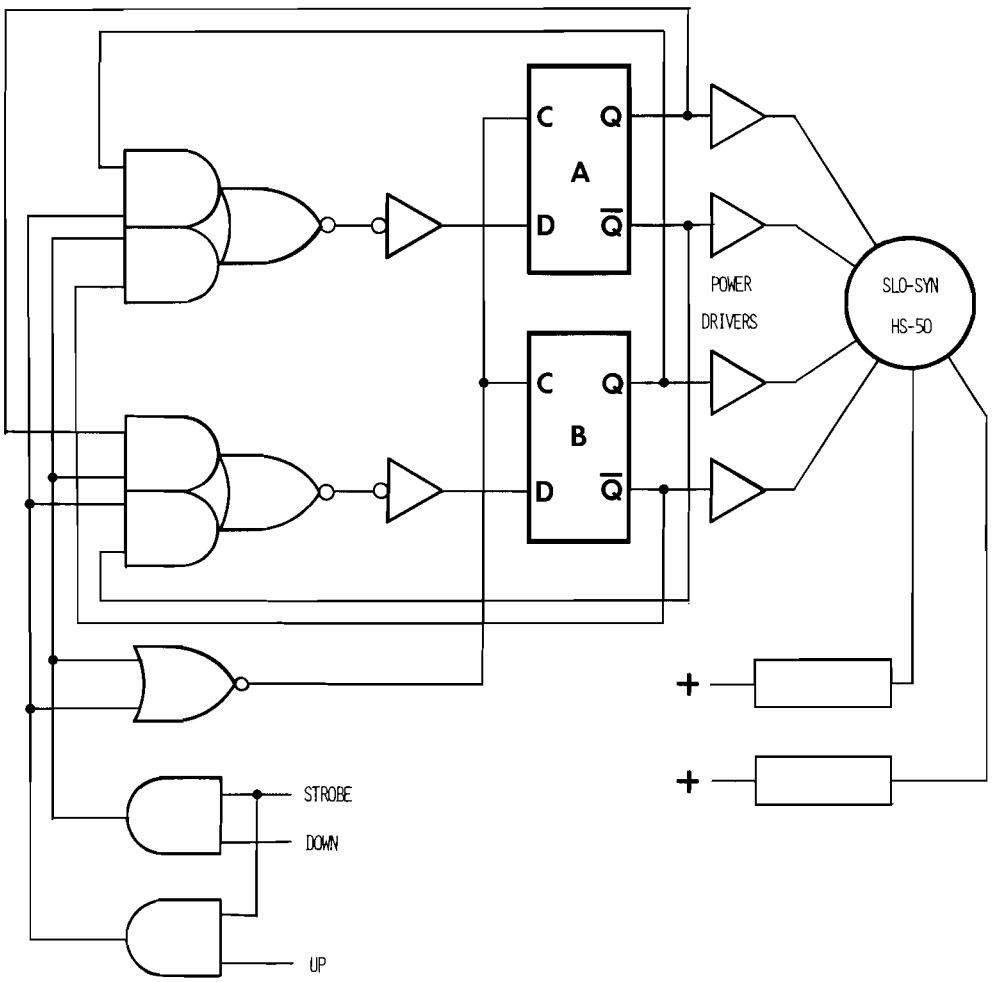


FIGURE 21. Stepping motor translator

Step	A	B
1	0	0
2	0	1
3	1	1
4	1	0
5	0	0

This corresponds to the timing diagram of Fig. 22.

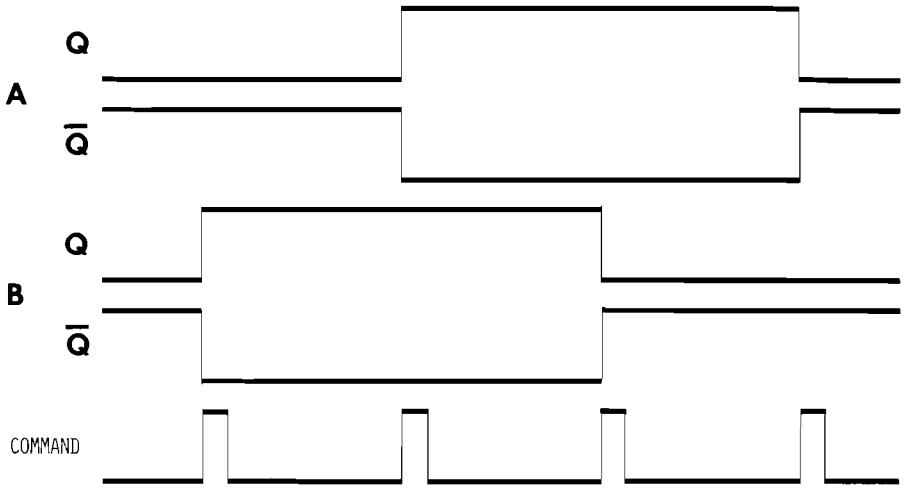


FIGURE 22. Translator output

Note that all four windings are being actuated at a frequency which is one fourth the stepping frequency. It is this lower frequency which is the ultimate limit to the stepping rate of the motor, as the motor will step no faster than the rate at which the windings can be turned on and off. This, in turn, is limited by the inductance/resistance ratio of the windings, which is reduced by the use of external dropping resistors. The tradeoff here is the power which must be dissipated by the dropping resistors.

Turning the windings on and off also generates transient voltage spikes. These are undesirable, and will adversely affect the digital logic. To minimize these spikes clamp diodes were installed at the motor to bypass them back to the power supply. The driver card has its own diodes, but were not used in favor of clamping the spikes as close as possible to the source, and as far as possible from the logic.

Provision was made for the use of a dual power supply for the stepping motor, connected at terminal strip J55, instead of a single, but is not so restricted. Either a 15 volt 6 amp, or dual 15 volt 3 amp, supply is required.

Digital inputs, which include the control switches on the remote control box and the data selector switches inside the cabinet, are connected directly to the data multiplexers used to input data to the MCS-8. The START pushbutton switch serves a dual purpose. Besides being available as data to the processor it is also wired to one of the bits of the interrupt instruction. In this way it can influence what happens when the RESTART pushbutton is pressed, which causes the hardwired interrupt instruction to be executed. The two interrupt instructions which can be executed are restart to location 0 or 10_8, depending on whether the START switch is or is not depressed when the RESTART button is pressed. The former causes ODT to be executed, while the latter causes the control program to be executed. Normally the control program will be started so extra effort is required to start ODT.

The serial interface permits communication with input-output terminals such as Teletypes. The logic of this interface is mostly contained on a single large scale integrated circuit. This circuit converts both parallel data to serial and serial data to parallel, as well as adding "start" and "stop" bits. Because Teletypes provide only contact outputs a modification must be made to generate a 0 to +10 volt signal. A positive supply voltage must also be provided for the printer for use as a current source. The interface transmits to the Teletype by sinking this current, or not sinking it.

Digital Simulator

A digital simulation of the braze process was designed and built to "close the loop" during development, and as a maintenance tool during installation and repair. It allows the processor to be exercised independent of the process. The digital simulator, however, is not meant to be equivalent to the process in response, only representative.

Basically it is a first order digital filter, exhibiting the transfer characteristic

$$d(z) = \frac{a}{b - c\, z^{-1}}$$

This is realized by an integrator (counter) whose output (value) is fed back to the input (see Fig. 23).

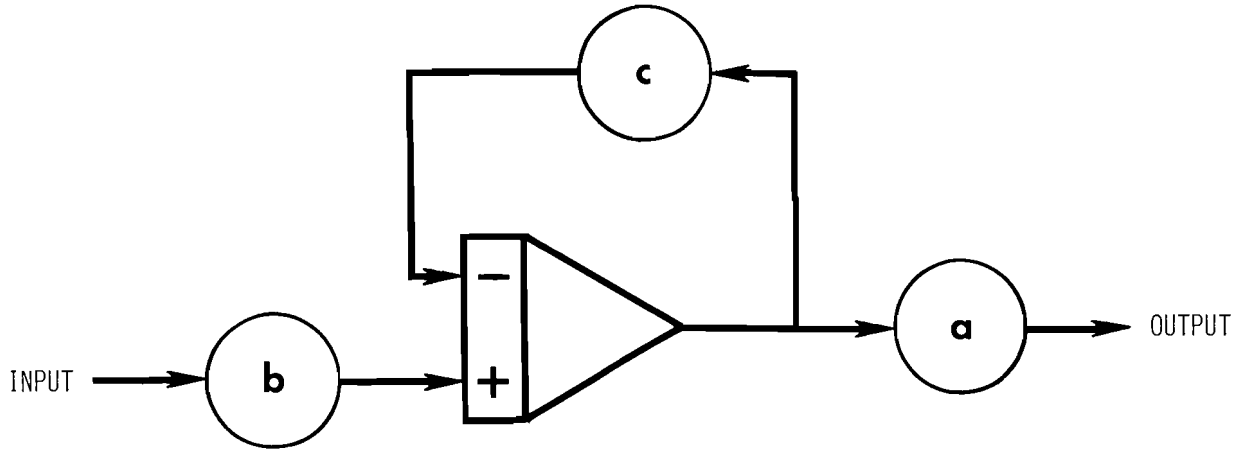


FIGURE 23. Block diagram of the digital simulator.

The multipliers b and c are implemented with digital frequency multipliers (SN7497). The integrator is a cascaded up/down counter. The counter is controlled by logic which prevents it from over or underflowing.

Because the output of the controller is pulsed to a stepping motor the simulator has an up/down counter to accumulate the output pulses and produce a parallel binary value equivalent to the stepping motor position. A frequency multiplier (represented by multiplier b) converts this to the frequency domain. This pulse stream is applied to the up clock of the integrator.

A digital/analog converter changes the process "temperature" to voltage form. The 0 - 10v output is inverted by an amplifier, and scaled by a voltage divider to 0 to +50 mv. This signal is returned to the controller and applied to the input amplifier.

The clock for the simulator comes from the real time clock. This must be running for the simulator to run. Changing the clock rate (say it were connected to a pulse generator instead) would change the time scale of the simulation. Heating and cooling time constants are altered by changing b and c, respectively. At present no provision has been made in the simulator to do this.

ACKNOWLEDGMENT

The author gratefully acknowledges the assistance of George McFarland of Lawrence Livermore Laboratory without whom this project would not have been possible.

BIBLIOGRAPHY

1. Cadzow, J. A., Discrete Time Systems, Prentice-Hall, Englewood Cliffs, NJ, 1973.

2. Gorman, Ken, P. Kaufman, "Low Cost Minicomputer Opens Up Many New System Opportunities", Electronics, June 7, 1973, Vol. 46, pp. 109-115.

3. Kuo, B. C., Analysis and Synthesis of Sampled Data Control Systems, Prentice-Hall, Englewood Cliffs, NJ, 1963.

4. Langill, A. W., "Digital Brain for a Digital Valve," Control Engineering, Dec. 1972, pp. 44-46.

5. Lapidus, A., "MOS/LSI Launches the Low Cost Processor," IEEE Spectrum, Nov. 1972, pp. 33-40.

6. Lewis, D. R., W. R. Siena, "Microprocessor or Random Logic?," Electronic Design, Sept. 1, 1973, Vol. 21 no. 18, pp. 106-110.

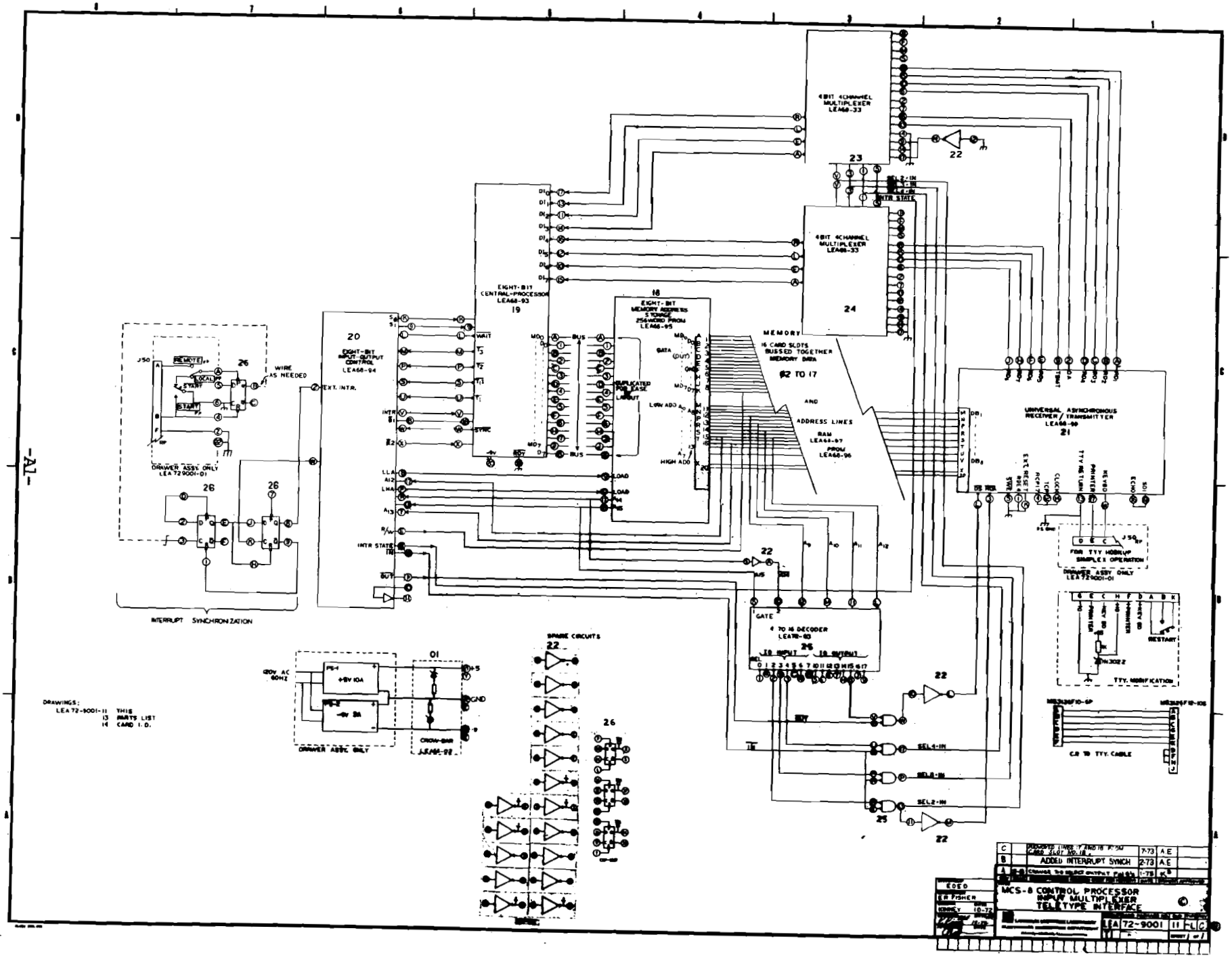
7. Lewis, D. R., and W. R. Siena, "How to Build a Microcomputer," Electronic Design, Vol. 21 no. 19, pp. 60-65.

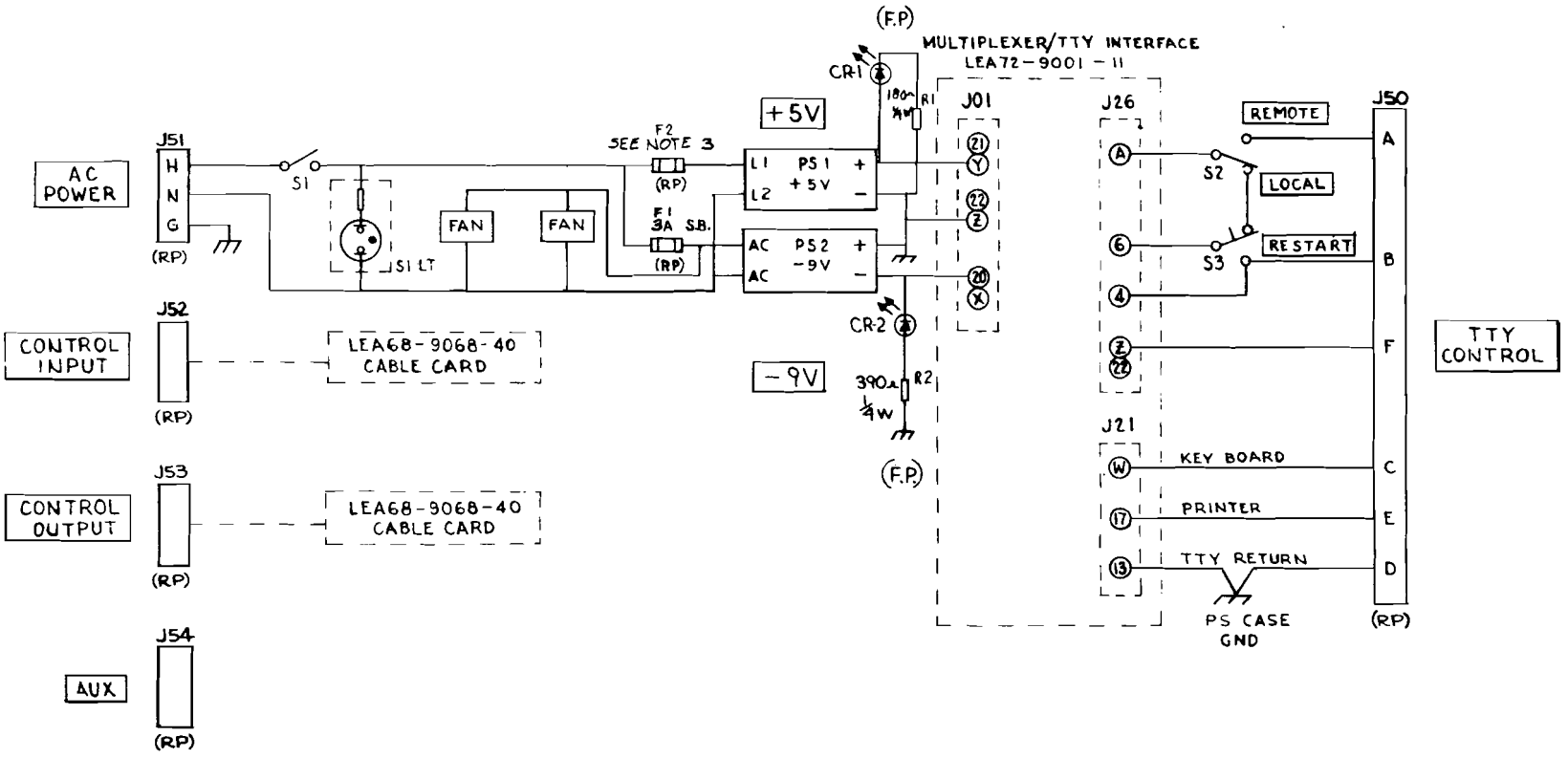
8. Lewis, D. R. and W. R. Siena, "Clear the Hurdles of Microprocessors," Electronic Design, Sept. 27, 1973, Vol. 21, no. 20, pp. 76-80.

9. _____ MCS-8 Users Manual, Revision 3, March 1973.

10. Wickes, William E., "A Compatible MOS/LSI Microprocessor Device Family," Computer Design, July 1973, Vol. 12, pp. 75-81.

APPENDIX A


4 BIT 4 CHANNEL MULTIPLEXER LEA68-33
4 BIT 4 CHANNEL MULTIPLEXER LEA68-33
EIGHT-BIT CENTRAL-PROCESSOR LEA68-93
19
EIGHT-BIT MEMORY REGISTERS 5 TO 256K 256-WORD PROM LEA68-95
18
MEMORY
16 CARD SLOTS BUSSED TOGETHER MEMORY DATA
D2 TO 17
AND
ADDRESS LINES
RAM LEA68-97
PROM LEA68-96
EIGHT-BIT INPUT OUTPUT CONTROL LEA68-94
20
UNIVERSAL ASYNCHRONOUS RECEIVER / TRANSMITTER LEA68-99
21
4 TO 16 DECODER LEA68-103
25
GATE
REMOTE
LOCAL
START
RESTART
DRAWER ASSY ONLY LEA 72-9001-01
WIRE AS NEEDED
INTERRUPT SYNCHRONIZATION
SPARE CIRCUITS
FOR TTY HOOKUP SIMPLEX OPERATION
TTY MODIFICATION
C.R. TO TTY CABLE
DRAWINGS:
LEA 72-9001-11 THIS
13 PARTS LIST
14 CARD I.D.
ADDED INTERRUPT SYNCH
MCS-8 CONTROL PROCESSOR
INPUT MULTIPLEXER
TELETYPE INTERFACE
LEA 72-9001
-A1-



DRAWINGS/ASSEMBLIES REQUIRED:

LEA72-9001-01 THIS DWG.
- -03 PARTS LIST.
- -04 P.S. BRACKET.
- -06 FRONT PANEL TEMPLATE.
- -07 FRONT PANEL SILK SCREEN.
- -08 REAR PANEL TEMPLATE.
- -09 REAR PANEL SILK SCREEN.
- -11 MULTIPLEXER/TTY INTERFACE-ASSEMBLY.
- -13 PARTS LIST FOR -11.
- -14 CARD I.D. FOR -11.
- -15 TERMINAL COVER FOR RO 210
- -16 " " " RO 106

NOTES:

1. FOR TTY OPTION SEE LEA72-9001-11.
2. REMOVE FRAME NEAREST FRONT PANEL.
3. IF PS1 IS LED21559 FUSE WITH 2A; IF PS1 IS LED 21686 FUSE WITH 3A.

REV	ZONE	DESCRIPTION (SEE REVISION SHEET FOR HISTORY)	DATE	DRAWN	APPROVED
C	–	ADDED NOTE 3	8/73	C. H.	
B	–	ADDED R1+R2, CH 53 LABEL	4-19-73	LAL	
A	–	ADDED CR1&CR2	2-8-73	EDWARD	

DIV/GROUP EE&D
ORIGINATOR LINING&R
DRAWN D. GRINDAHL 11-14-72
CHECKED DATE
APPROVED DATE

TITLE: MCS8 CONTROL PROCESSER DRAWER ASSEMBLY

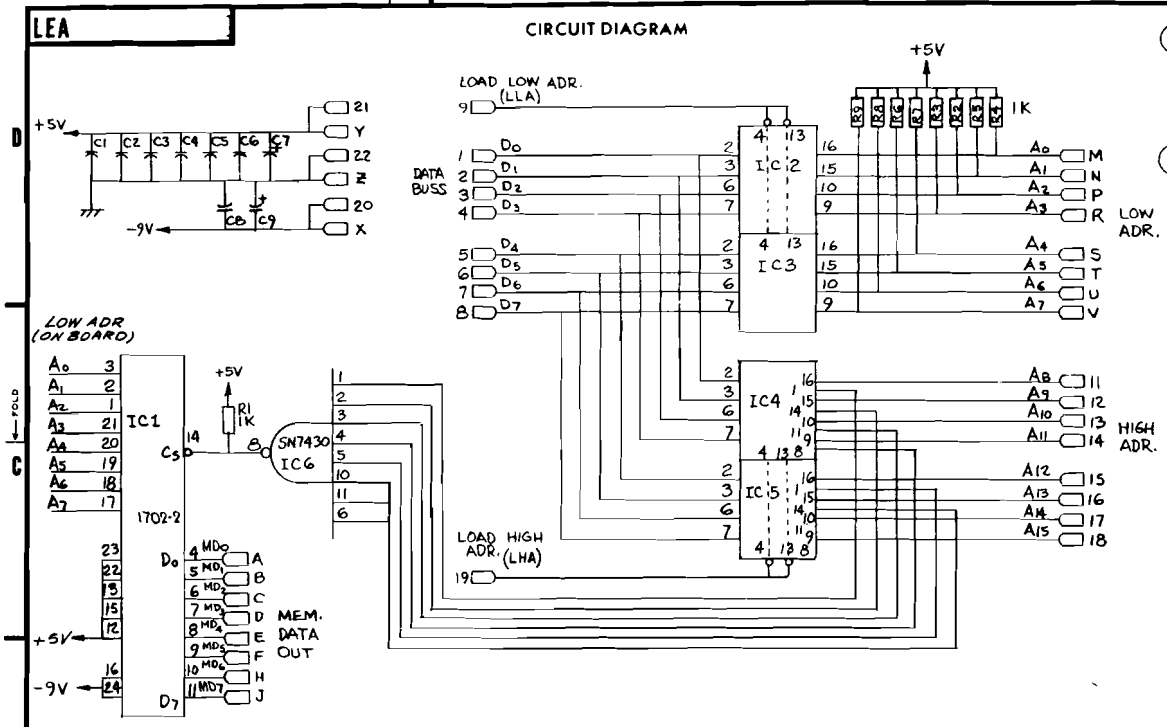
UNIVERSITY OF CALIFORNIA
LAWRENCE RADIATION LABORATORY
ELECTRONICS ENGINEERING DEPARTMENT
LIVERMORE, CALIFORNIA 94550

GROUP	BASIC PACKAGE NO.	SUB	TYPE	REV
LEA	72-9001	01	-S	C

SHEET 1 OF 1

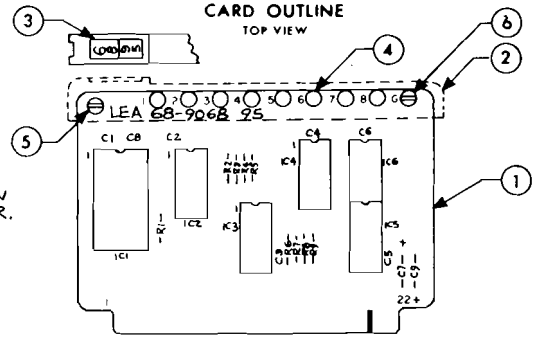
LEA

CIRCUIT DIAGRAM



CARD OUTLINE

TOP VIEW



NOTE

1. SYMBOLS PER LED68-902013
2. DESIGNATIONS PER LED68-902014
3. UNLESS OTHERWIRSE SPECIFIED, ON ALL LEAXX-9070 XX BOARDS
   (a) COMPONENT S TO BE NO GREATER THAN .250 MAX HEIGHT FROM BOARD WHEN LOADED.
   (b) THIS DRAWING NUMBER TO BE STENCILED ON CARD HANDLE, COMPONENT SIDE OF BOARD AS SHOWN ABOVE.
4. SN7475 PIN 12 = GND.
   PIN 5 = +5V
5. 1702-2-IC1 STK. NO. 5961-59498 TO BE LOADED BY USER
6. LOADED CARD LLL STOCK NO. 5975-59536.

SOCKET (FOR IC1)	24	LSC-1DC4-1 AUGAT	N/S
R1,R2,R3,R4,R5,R6,R7,R8,R9	9	1K ¼W ± 5% COMP	5905-16489
IC6	1	SN7430	5961-53223
IC2, IC3, IC4, IC5	4	SN7475	5961-53238
IC1		(SEE NOTE 5)	
C7, C9	2	1 µF ± 10% 50V TANT (PER LED 21418)	5910-52582
C1,C2,C3,C4,C5,C6,C8	7	.01 µF ± 20% 100V CK05 (PER LED 21491)	5910-57925
⑥	1	GOLD GROUND SCREW, CONTROLLOGIC NO 183.050	5975-56592
⑤	1	SELF TAPPING SCREW, 3/8 LX 4-40, BLACK ANODIZE	5305-57101
④	8	TEST POINT PINS, CONTROL LOGIC NO. 183.164	5975-56591
③	1	LABEL, HANDLE, CONTROL LOGIC NO. 183.052	5975-568
②	1	HANDLE, CARD, CONTROL LOGIC NO 183.163	5975-56590
LEA 68-9070 95 ①	1	CARD, PRINTED CIRCUIT (ARTWORK – SHEETS, SIZE C)	
LED11020	–	ELECTRONIC FAB. SPEC.	
LED11344	–	SOLDERING SPEC	
LED11591	–	P.C. BOARD PRODUCTION SPEC	
LED11606	–	P.C. EQUIPMENT SPEC	

COMPONENT DESIGNATION DRAWING OR SPECIFICATION | QTY REQD | DESCRIPTION | STOCK NO FOR INTERNAL USE ONLY

COMMERCIAL PARTS & MATERIALS

* SOURCES SHALL BE THOSE LISTED ON LED 11591 UNLESS OTHERWISE SPECIFIED ON THIS SHEET

PARTS LIST

PIN CONNECTIONS

A	MD0	1	D0
B	MD1	2	D1
C	MD2	3	D2
D	MD3	4	D3
E	MD4	5	D4
F	MD5	6	D5
H	MD6	7	D6
J	MD7	8	D7
K		9	LLA
L		10	
M	A0	11	A8
N	A1	12	A9
P	A2	13	A10
R	A3	14	A11
S	A4	15	A12
T	A5	16	A13
U	A6	17	A14
V	A7	18	A15
W		19	LHA
X	-9V	20	-9V
Y	+5V	21	+5V
Z	GND	22	GND

TEST TERMINALS

1		5	
2		6	
3		7	
4		8	

KEY POSITIONS

20

DIV/GROUP EOED

ORIGINATOR FISHER/M&M ENG.

DRAWN DATE ROSKELLEY 24-AUG'72

CHECKED DATE

APPROVED DATE 10-5-72

REV	ZONE	DESCRIPTION (SEE REVISION SHEET FOR HISTORY)	DATE	DRAWN	APPROVED

TITLE

EIGHT BIT, MEMORY ADDRESS STORAGE & 256 WORD PROM.

UNIVERSITY OF CALIFORNIA
LAWRENCE RADIATION LABORATORY
ELECTRONICS ENGINEERING DEPARTMENT
LIVERMORE, CALIFORNIA 94550

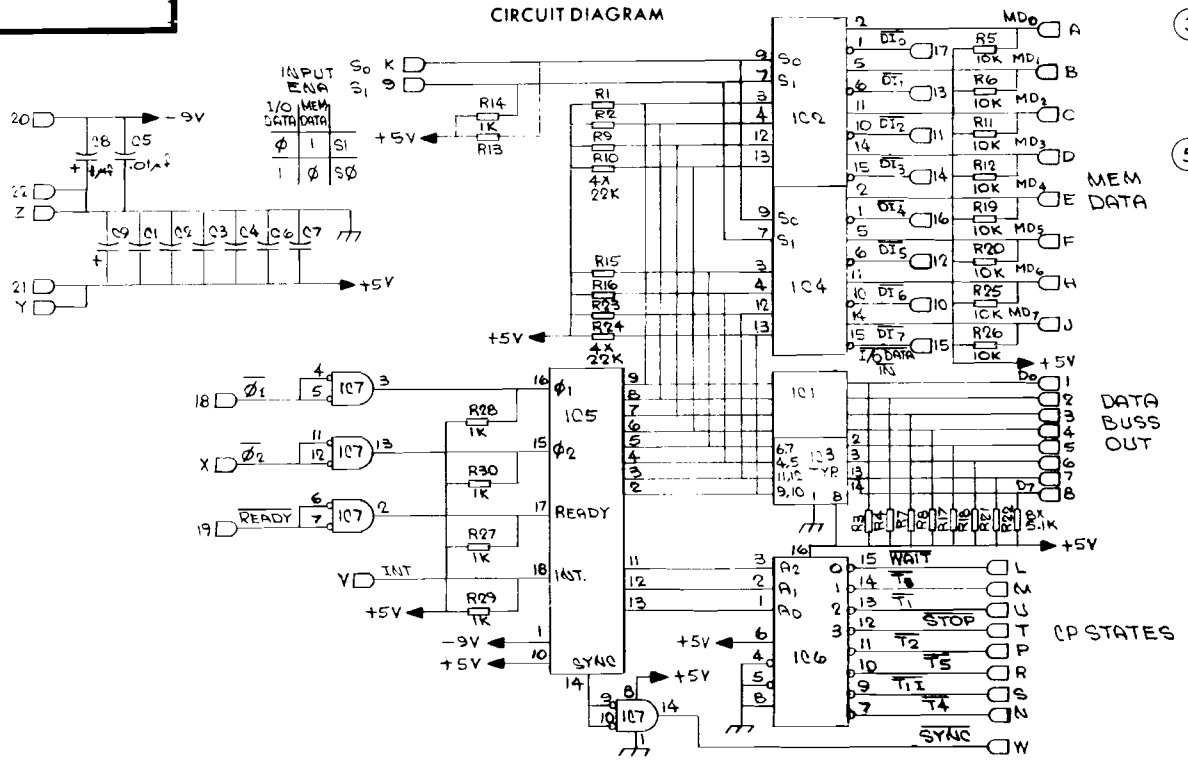
GROUP	BASIC PACKAGE NO	SUB	TYPE	REV
LEA	68-9068	95	-A	0

SIZE C

SHEET 2 OF 2

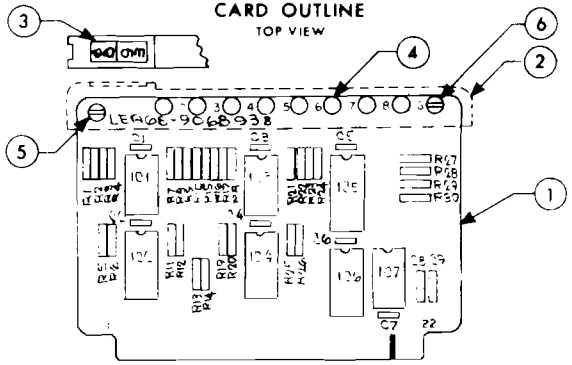
LEA

CIRCUIT DIAGRAM



CARD OUTLINE

TOP VIEW



NOTE

1. SYMBOLS PER LED68-902013
2. DESIGNATIONS PER LED68-902014
3. UNLESS OTHERWIRSE SPECIFIED, ON ALL LEAXX 9070 XX BOARDS
   - (a) COMPONENT S TO BE NO GREATER THAN .250 MAX HEIGHT FROM BOARD WHEN LOADED
   - (b) THIS DRAWING NUMBER TO BE STENCILED ON CARD HANDLE COMPONENT SIDE OF BOARD AS SHOWN ABOVE
4. LOADED CARD LLL STOCK NO. 5975-59534.
5. LLL WILL SUPPLY IC5 (8008).

PARTS LIST

COMMERCIAL PARTS & MATERIALS

COMPONENT DESIGNATION DRAWING OR SPECIFICATION	QTY REQD	DESCRIPTION	STOCK NO
R13, R14, R27, R28, R29, R30	6	1K 1/4W ±5% COMP.	5905-16489
R5, R6, R11, R12, R19, R20, R25, R26	8	10K 1/4W ±5% COMP.	5905-16513
R3, R4, R7, R8, R17, R18, R21, R22	8	5.1K 1/4W ±5% COMP.	5905-16506
R1, R2, R9, R10, R15, R16, R23, R24	8	22K 1/4W ±5% COMP.	5905-16521
IC7	1	SP380	5961-59491
IC6	1	3205 INTEL	5961-59496
IC5	1	8008	5961-59497
IC2, IC4	2	8267	5961-59479
IC1, IC3	2	SP384	5961-59492
C8, C9	2	1 MF ±10% 50V TANT (PER LED 21418)	5910-52582
C1 THRU C7	7	.01 MF ± 20% 100V CK05 (PER LED 21491)	5910-57925
(6)	1	GOLD GROUND SCREW, CONTROL LOGIC NO 183-030	5975-56592
(5)	1	SELF TAPPING SCREW, 3/8 LX 4-40, BLACK ANODIZE	5305-57101
(4)	8	TEST POINT PINS, CONTROL LOGIC NO 183-164	5975-56591
(3)	1	LABEL, HANDLE, CONTROL LOGIC NO 183-052	5975-568
(2)	1	HANDLE, CARD, CONTROL LOGIC NO 183-163	5975-56590
LEA68-9070 93 (1)	1	CARD, PRINTED CIRCUIT (ARTWORK __ SHEETS, SIZE C)	
LED11020	–	ELECTRONIC FAB. SPEC.	
LED11544	–	SOLDERING SPEC	
LED11391	–	P.C. BOARD PRODUCTION SPEC	
LED11606	–	P.C. EQUIPMENT SPEC.	

PIN CONNECTIONS

A	MD0	1	D0
B	MD1	2	D1
C	MD2	3	D2
D	MD3	4	D3
E	MD4	5	D4
F	MD5	6	D5
H	MD6	7	D6
J	MD7	8	D7
K	SC	9	S1
L	WAIT	10	DI6
M	T3	11	DI2
N	T4	12	DI5
P	T2	13	DI1
R	T5	14	DI3
S	T11	15	DI7
T	STOP	16	DI4
U	T1	17	DIC
V	INT	18	DI
W	SYNC	19	READY
X	Ø2	20	-9V
Y	+5V	21	+5V
Z	GND	22	GND

TEST TERMINALS

1		5	
2		6	
3		7	
4		8	

KEY POSITIONS

20

REV	ZONE	DESCRIPTION (SEE REVISION SHEET FOR HISTORY)	DATE	DRAWN	APPROVED
B		CHANGED NOTE 5	6/73	C.H.	
A		PAD ON ARTWORK	1/73	m&m	

DIV/GROUP EOED
ORIGINATOR FISHER/M&M ENG.
DRAWN KERLIN 8-14-72
CHECKED BG DATE 11/4/72
APPROVED DATE 11-5-72

TITLE

EIGHT BIT CENTRAL PROCESSOR

(6) (5) (4)

UNIVERSITY OF CALIFORNIA
LAWRENCE RADIATION LABORATORY
ELECTRONICS ENGINEERING DEPARTMENT
LIVERMORE, CALIFORNIA 94550

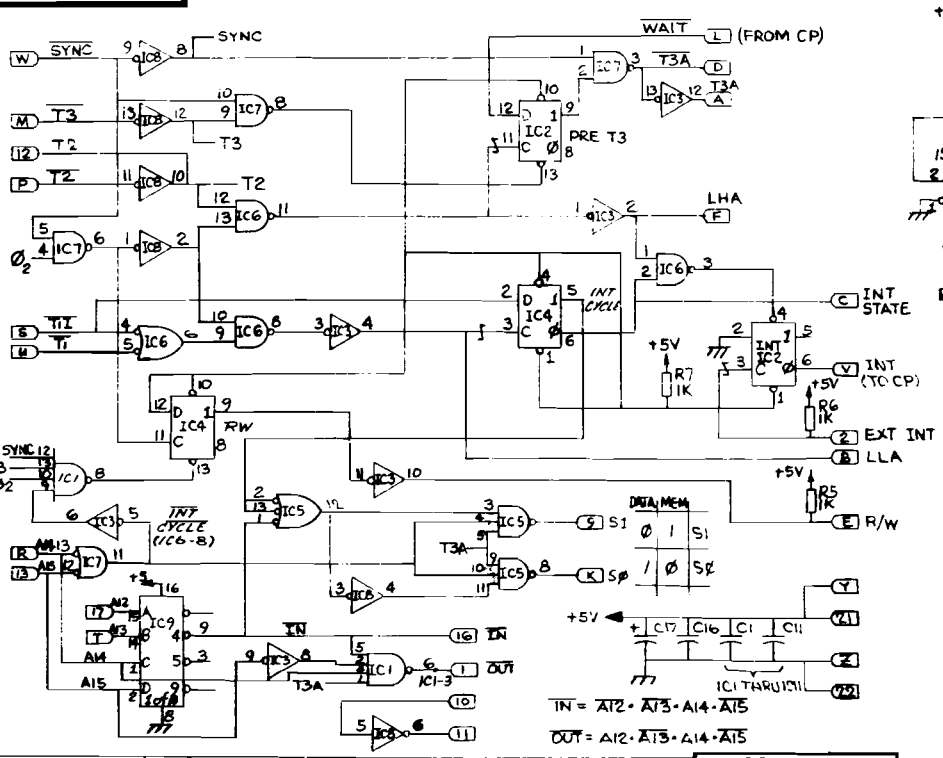
GROUP	BASIC PACKAGE NO.	SUB	TYPE	REV
LEA	68-9068	93	-A	B

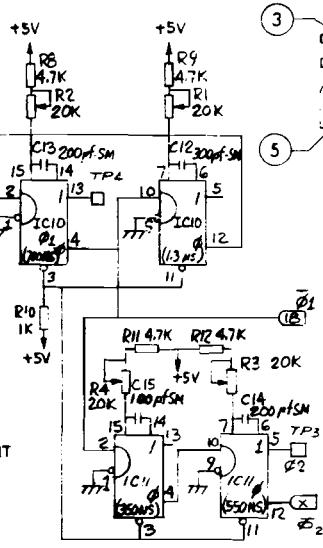
SIZE C | SHEET 2 OF 2

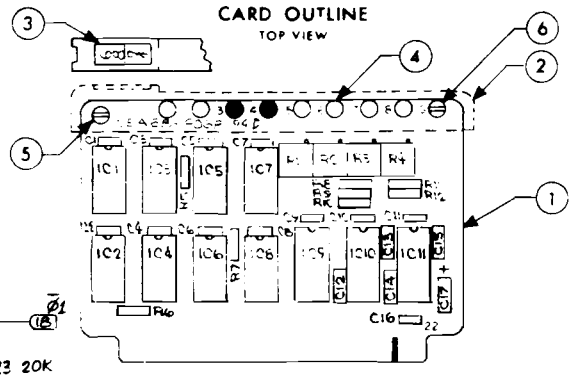
-24-

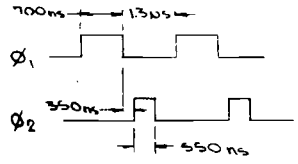
LEA

CIRCUIT DIAGRAM









NOTE

1. SYMBOLS PER LED68.902013
2. DESIGNATIONS PER LED68.902014
3. UNLESS OTHERWISE SPECIFIED, ON ALL LEAXX 9070 XX BOARDS
   - a. COMPONENTS TO BE NO GREATER THAN .250 MAX HEIGHT FROM BOARD WHEN LOADED
   - b. THIS DRAWING NUMBER TO BE STENCILED ON CARD HANDLE COMPONENT SIDE OF BOARD AS SHOWN ABOVE
4. LOADED CARD LLL STOCK NO. 5975-59535.
5. 8 PINS LOADED, DARKENED CIRCLES IN OUTLINE ABOVE, INDICATE CIRCUIT TEST POINTS.
6. PARTS & COMPONENTS WITH X ARE NOT TO BE SUPPLIED BY LLL.

PARTS LIST

COMMERCIAL PARTS & MATERIALS

COMPONENT DESIGNATION DRAWING OR SPECIFICATION	QTY REQD	DESCRIPTION	STOCK NO FOR MATERIAL, NOT ONLY
C15	1	100 pf ± 5 500V SILVER MICA	5910-48423
R8, R9, R11, R12	4	4.7K ± 5% ¼W COMP.	5905-16505
R5, R6, R7, R10	4	1K ± 5% ¼W COMP.	5905-16489
R1 THRU R4	4	20K CERMET TRIMMER (PER LED 21458)	5905-57503
IC10, IC11	2	SN74123	5961-59093
IC9	1	9301	5961-56188
IC6, IC7	2	SN7400	5961-53218
IC5	1	SN7410	5961-53221
IC3, IC8	2	SN7404	5961-57216
IC2, IC4	2	SN7474	5961-53231
IC1	1	SN7420	5961-53222
C17	1	1 Mf ± 10% 50 V TANT (PER LED 21418)	5910-52582
C13, C14	2	200 Mf ± 5% 500V SILVER MICA	5910-48425
C12	1	300 Mf ± 5% 500 V SILVER MICA	5910-48426
C1 THRU C11 & C16	12	.01 Mf ± 20% 100 V CKO 5 (PER LED 21491)	5910-57925
(6)X	1	GOLD GROUND SCREW, CONTROL LOGIC NO 183-050	5975-56592
(5)X	1	SELF TAPPING SCREW, 3/8 L X 4-40, BLACK ANODIZE	5305-57101
(4)X	8	TEST POINT PINS, CONTROL LOGIC NO 183-164	5975-56591
(3)X	1	LABEL, HANDLE, CONTROL LOGIC NO 183-052	5975-568
(2)X	1	HANDLE, CARD, CONTROL LOGIC NO 183-163	5975-56590
LEA68-9070-5M (1)X	1	CARD, PRINTED CIRCUIT (ARTWORK — SHEETS, SIZE C)	
LED11020	-	ELECTRONIC FAB. SPEC.	
LED11544	-	SOLDERING SPEC	
LED11591	-	P.C. BOARD PRODUCTION SPEC	
LED11606	-	P.C. EQUIPMENT SPEC.	

PIN CONNECTIONS

A	T3L	1	OUT		
B	LLA	2	EXT INT		
C	INT STATE	3			
D	T3A	4			
E	R/W	5			
F	LHA	6			
H		7			
J		8			
K	SØ	9	S1		
L	WAIT	10			
M	T3	11			
N		12	T2		
P	T2	13	A15		
R	A14	14			
S	T1	15			
T	A13	16	IN		
U	T1	17	A12		
V	INT (TO CP)	18	Ø1		
W	SYNC	19			
X	Ø2	20			
Y	+5V	21	+5V		
Z	GND	22	GND		

TEST TERMINALS

1		5	
2		6	
3	Ø2	7	
4	Ø1	8	

KEY POSITIONS

20

REV	ZONE	DESCRIPTION (SEE REVISION SHEET FOR HISTORY)	DATE	DRAWN	APPROVED
D	--	CORRECTED DESIG. ON IC6, PIN 5	2/73	C.H.	
C	-	PAD SIZE ON ARTWORK	1/73	mon	
B	-	REWIRED IC1, 3 & 9	1/73	C.H.	
A	-	NOTE 5 & 6 ADDED CHANGE C12 FROM 200 PF TO 100 PF	10-6-72	Kerlin	

DIV/GROUP EOED
ORIGINATOR FISHER/M&M ENG.
DRAWN J. ROSKELLEY DATE 24-AUG-72
CHECKED DATE
APPROVED DATE 10-5-72

TITLE: EIGHT BIT INPUT-OUTPUT CONTROL

UNIVERSITY OF CALIFORNIA
LAWRENCE RADIATION LABORATORY
ELECTRONICS ENGINEERING DEPARTMENT
LIVERMORE CALIFORNIA 94550

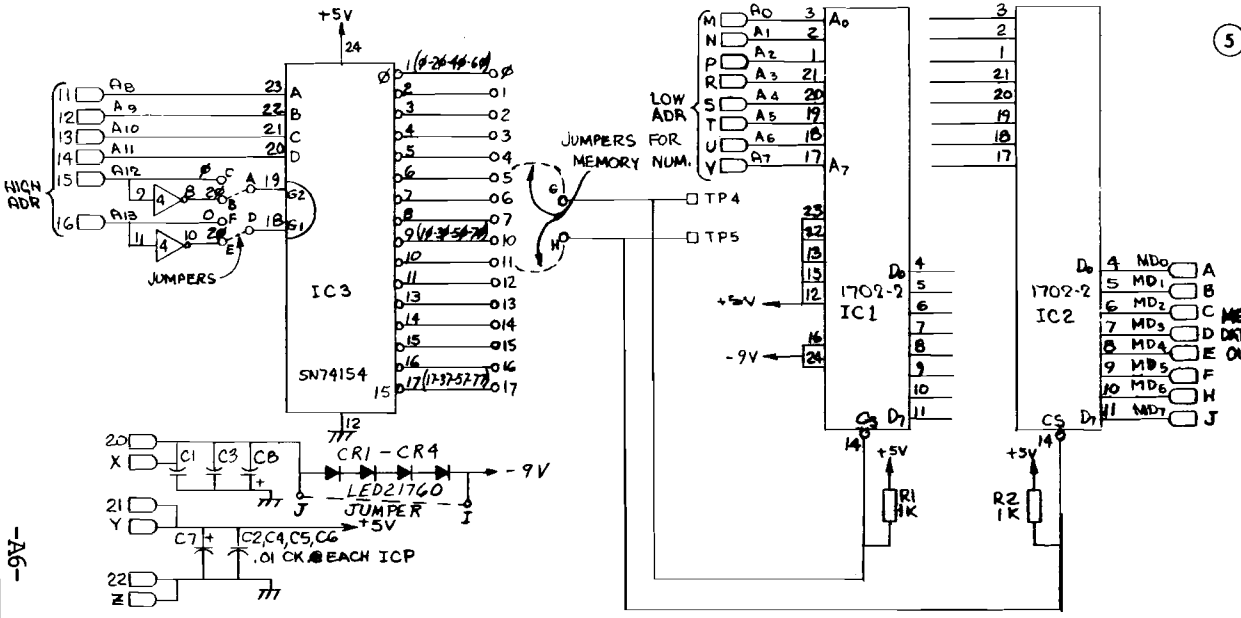
GROUP	BASIC PACKAGE NO	SUB	TYPE	REV
LEA	68-9068	94	-A	D

SIZE C | SHEET 2 OF 2

FOLD → | -A5- | 4 | 3 | 2 | 1

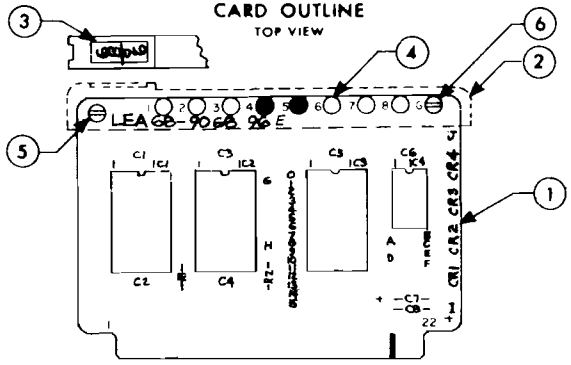
LEA

CIRCUIT DIAGRAM



CARD OUTLINE

TOP VIEW



NOTE

1. SYMBOLS PER LED68-902013
2. DESIGNATIONS PER LED68-902014
3. UNLESS OTHERWIRSE SPECIFIED, ON ALL LEAXX-9070 XX BOARDS
   (a) COMPONENT S TO BE NO GREATER THAN .250 MAX HEIGHT FROM BOARD WHEN LOADED.
   (b) THIS DRAWING NUMBER TO BE STENCILED ON CARD HANDLE, COMPONENT SIDE OF BOARD AS SHOWN ABOVE.
4. 1702-2 = IC1 & IC2 STOCK NO. 5961-59498 TO BE LOADED BY USER.
5. LOADED CARD LLL STOCK NO. 5975-59537.
6. 8 TEST PINS LOADED, DARKENED CIRCLES IN OUTLINE ABOVE INDICATE CIRCUIT TEST POINTS.
7. ALL JUMPERS TO BE INSTALLED BY USER.
8. PARTS # COMPONENTS WITH * ARE NOT TO BE SUPPLIED BY LLL.
9. INSTALL JUMPER WHEN -9V SUPPLY IS USED

PARTS LIST

COMPONENT DESIGNATION DRAWING OR SPECIFICATION	QTY REQD	DESCRIPTION *	STOCK NO FOR INTERNAL USE ONLY
SOCKET (FOR IC1, IC2)	48	LSG-1DC4-1 AUGAT	5940-59656
R1, R2	2	1K ¼W ± 5% COMP	5905-16489
CR1 - CR4	4	DIODE, LED21760	5961-59658
IC4	1	SN 7404	5961-57216
IC3	1	SN 74154	5961-59319
IC1, IC2		(SEE NOTE 4)	
C7, C8	2	1 μf ± 10% 50V TANT (PER LED 21498)	5910-52582
C1, C2, C3, C4, C5, C6	6	.01 μf ± 20% 100V CKO5 (PER LED 21491)	5910-57925
⑥ *	1	GOLD GROUND SCREW, CONTROL LOGIC NO. 183-050	5975-56592
⑤ *	1	SELF TAPPING SCREW, 3/8 L X 4-40, BLACK ANODIZE	5305-57101
④ *	8	TEST POINT PINS, CONTROL LOGIC NO. 183-164	5975-56591
③ *	1	LABEL, HANDLE, CONTROL LOGIC NO. 183-052	5975-568
② *	1	HANDLE, CARD, CONTROL LOGIC NO. 183-163	5975-56590
LEA68-9070 96 ① *	1	CARD, PRINTED CIRCUIT (ARTWORK _ SHEETS, SIZE C)	
LED11020	–	ELECTRONIC FAB. SPEC.	
LED11544	–	SOLDERING SPEC.	
LED11591	–	P.C. BOARD PRODUCTION SPEC.	
LED11606	–	P.C. EQUIPMENT SPEC.	

COMMERCIAL PARTS & MATERIALS

* SOURCES SHALL BE THOSE LISTED ON LED 11606 UNLESS OTHERWISE SPECIFIED ON THIS SHEET

PIN CONNECTIONS

A	MD0	1	
B	MD1	2	
C	MD2	3	
D	MD3	4	
E	MD4	5	
F	MD5	6	
H	MD6	7	
J	MD7	8	
K		9	
L		10	
M	A0	11	A8
N	A1	12	A9
P	A2	13	A10
R	A3	14	A11
S	A4	15	A12
T	A5	16	A13
U	A6	17	
V	A7	18	
W		19	
X	-9V	20	-9V
Y	+5V	21	+5V
Z	GND	22	GND

TEST TERMINALS

1		5	
2		6	
3		7	
4		8	

KEY POSITIONS

20

DIV/GROUP EOED

ORIGINATOR FISHER/M&M ENG.

DRAWN Remelley DATE 24 AUG 72

CHECKED DATE

APPROVED DATE 10-5-72

REV	ZONE	DESCRIPTION (SEE REVISION SHEET FOR HISTORY)	DATE	DRAWN	APPROVED
E	–	REMOVED FORKED TERMINALS	6/73	C.H.	
D	–	ADDED CR1 THRU CR2	4/73	C.H.	
C	–	PAD SIZE ON ARTWORK	1/73	m&m	
B	–	ADDED STK. NO. ON IC SOCKET	1/73	C.H.	
A	–	TP4, TP5 & NOTE 6, 7 & 8 ADDED	10-19-72	O.G.G.	

TITLE

EIGHT BIT, 512 WORD PROM (E) (B) (D) (C) (A)

UNIVERSITY OF CALIFORNIA
LAWRENCE RADIATION LABORATORY
ELECTRONICS ENGINEERING DEPARTMENT
LIVERMORE, CALIFORNIA 94550

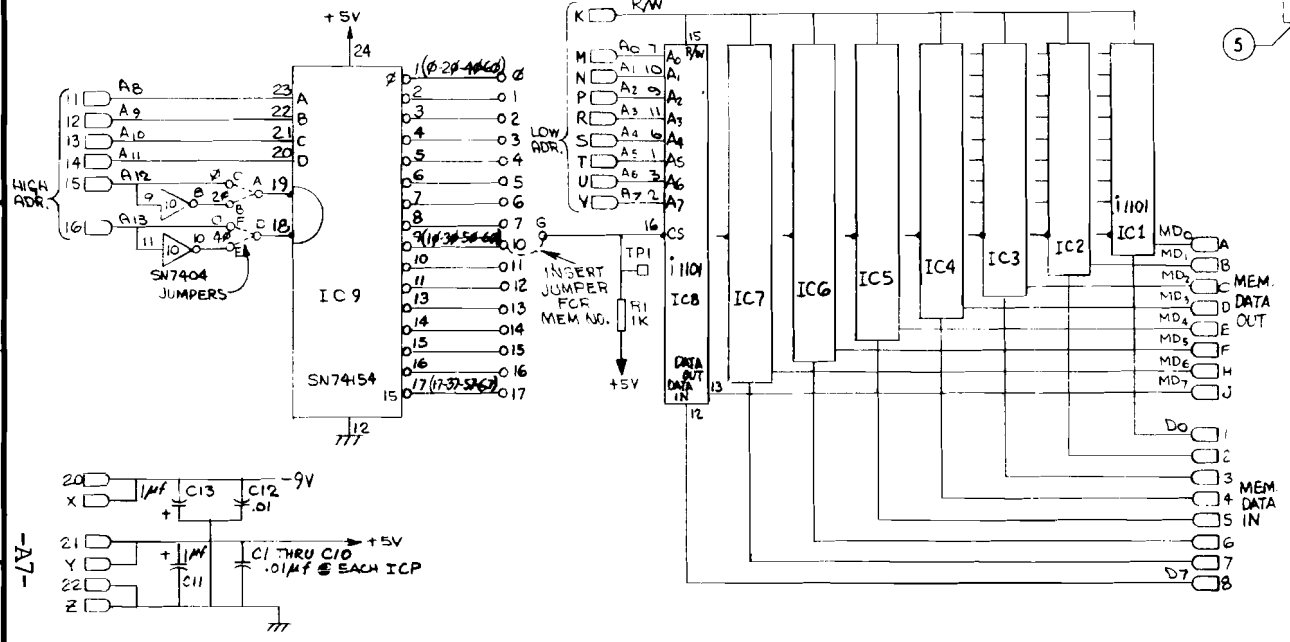
GROUP	BASIC PACKAGE NO.	SUB	TYPE	REV.
LEA	68-9068	96	-A	E

SIZE C

SHEET 2 OF 2

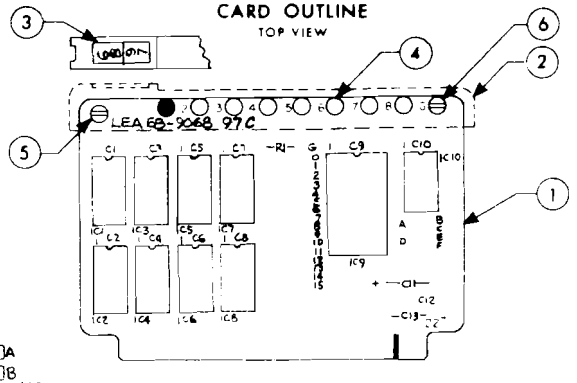
LEA

CIRCUIT DIAGRAM



CARD OUTLINE

TOP VIEW



NOTE

1. SYMBOLS PER LED68 902013
2. DESIGNATIONS PER LED68-902014
3. UNLESS OTHERWIRSE SPECIFIED ON ALL LEAXX-9070 XX BOARDS
   a. COMPONENT S TO BE NO GREATER THAN 250 MAX HEIGHT FROM BOARD WHEN LOADED
   b. THIS DRAWING NUMBER TO BE STENCILED ON CARD HANDLE COMPONENT SIDE OF BOARD AS SHOWN ABOVE
4. LOADED CARD LLL STOCK NO. 5975-59538.
5. 1101 PIN 5 = +5V
   PIN 4,8 = -9V
   PIN 14 = NC
6. 8 TEST PINS LOADED; DARKENED CIRCLE IN OUTLINE ABOVE INDICATES CIRCUIT TEST POINT.
7. ALL JUMPERS TO BE INSTALLED BY USER ONLY.
8. LLL WILL SUPPLY IC1 THRU IC8 (1101A)

COMPONENT DESIGNATION DRAWING OR SPECIFICATION	QTY REQD	DESCRIPTION	STOCK NO
R1	1	1K ¼w ± 5% COMP	5905-16489
IC10	1	SN7404	5961-57216
IC9	1	SN74154	5961-59319
IC1 THRU IC8	8	1101A	5961-59499
C11, C12	2	1 Mf ± 10% 50V TANT (PER LED21418)	5910-52582
C1 THRU C10, C12	11	.01 Mf ± 20% 100V CK05 (PER LED 21491)	5910-57925
6	1	GOLD GROUND SCREW, CONTROL LOGIC NO 183 050	5975-56592
5	1	SELF TAPPING SCREW, 3, BLX4 40, BLACK ANODIZE	5305-57101
4	8	TEST POINT PINS, CONTROL LOGIC NO 183 164	5975-56591
3	1	LABEL, HANDLE, CONTROL LOGIC NO 183-052	5975-568
2	1	HANDLE, CARD, CONTROL LOGIC NO 183 163	5975-56590
LEA68-9070 97 (1)	1	CARD, PRINTED CIRCUIT, ARTWORK, SHEETS, SIZE C	
LED11020	–	ELECTRONIC FAB SPEC	
LED11544	–	SOLDERING SPEC	
LED11591	–	P.C. BOARD PRODUCTION SPEC	
LED11606	–	P.C. EQUIPMENT SPEC	

COMMERCIAL PARTS & MATERIALS

PARTS LIST

PIN CONNECTIONS			
A	MD0	1	D0
B	MD1	2	D1
C	MD2	3	D2
D	MD3	4	D3
E	MD4	5	D4
F	MD5	6	D5
H	MD6	7	D6
J	MD7	8	D7
K	R/W	9	
L		10	
M	A0	11	A8
N	A1	12	A9
P	A2	13	A10
R	A3	14	A11
S	A4	15	A12
T	A5	16	A13
U	A6	17	
V	A7	18	
W		19	
X	-9V	20	-9V
Y	+5V	21	+5V
Z	GND	22	GND

TEST TERMINALS			
1		5	
2		6	
3		7	
4		8	

KEY POSITIONS: 20

DIV/GROUP: EOED
ORIGINATOR: FISHER/M&M ENG.
DRAWN: ROSKELLEY 2A AUG'72
CHECKED: DATE
APPROVED: DATE 10-5-72

REV	ZONE	DESCRIPTION (SEE REVISION SHEET FOR HISTORY)	DATE	DRAWN	APPROVED
C	–	CHANGED NOTE 8	8/73	C.H.	
B	–	PAD SIZE ON ART WORK	1/73	m&m	
A	–	TP1 & NOTE 6,7,8 ADDED	10-19-72	D.G.G	

TITLE: EIGHT BIT, 256 WORD RAM

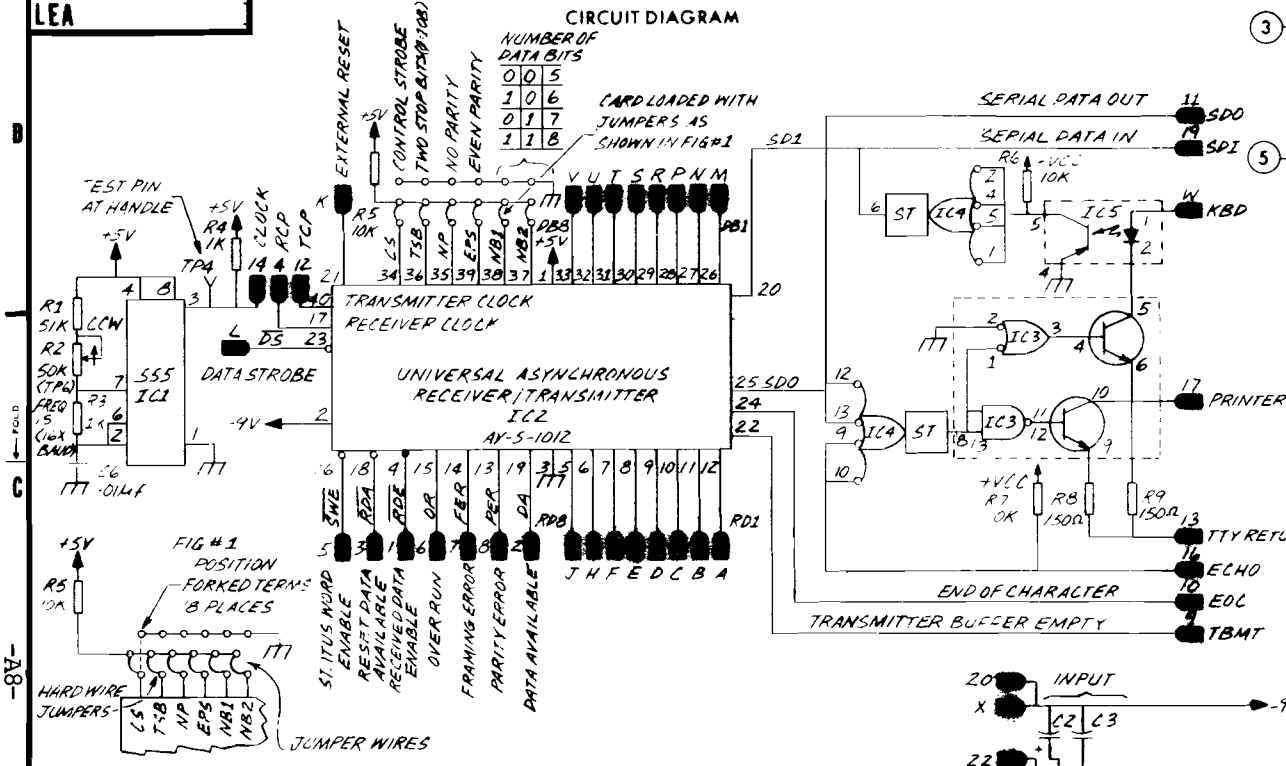
UNIVERSITY OF CALIFORNIA
LAWRENCE RADIATION LABORATORY
ELECTRONICS ENGINEERING DEPARTMENT
LIVERMORE CALIFORNIA 94550

GROUP	BASIC PACKAGE NO.	SUB	TYPE	REV
LEA	68-9068	97	-A	C

SIZE C — SHEET 2 OF 2

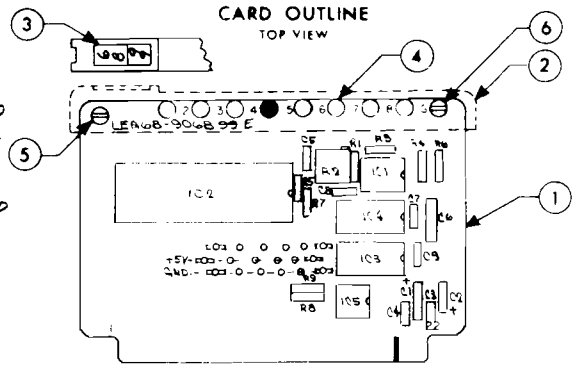
LEA

CIRCUIT DIAGRAM



CARD OUTLINE

TOP VIEW



NOTE

1. SYMBOLS PER LED68-902013
2. DESIGNATIONS PER LED68-902014
3. UNLESS OTHERWISE SPECIFIED, ON ALL LEAXX-9070 XX BOARDS
   (a) COMPONENT S TO BE NO GREATER THAN .250 MAX HEIGHT FROM BOARD WHEN LOADED.
   (b) THIS DRAWING NUMBER TO BE STENCILED ON CARD HANDLE COMPONENT SIDE OF BOARD AS SHOWN ABOVE
4. 7 TEST PINS LOADED, DARKENED CIRCLE IN OUTLINE ABOVE INDICATES CIRCUIT TEST POINT, PIN 6 NOT LOADED.
5. LOADED CARD LLL STOCK NO. 5975-59645
6. ALL JUMPERS TO BE INSTALLED BY USER ONLY.
7. LLL WILL SUPPLY IC2 (LED2174).

IC1	1	SIGNETICS 555-8 PIN DIP	5961-59382
IC2	1	LSD 2174 I -40 PIN	5961-59520
IC3	1	SN75450-14 PIN DIP	5961-58401
IC4	1	SN74 3-14 PIN DIP	5961-58400
IC5	1	FPL A320-6 PIN DIP	5961-59181
C1, C2	2	10μf TANT. 50V-CASE SIZE A	5910-52582
C3,C4,C5,C7,C8,C9	6	.01 μf CKOS	5910-57925
C6	1	.01 μf, 432P WESCO ELECT	5910-59662
R1	1	5IK 1/4 W CARBON COMP. 5%	5905-16530
R2	1	50K 1/2 W TRIM POT 25 TURN	5905-57504
R3 R4	2	1K 1/4W CARBON COMP. 5%	5905-16489
R5, R6, R7	3	10K 1/4 W CARBON COMP. 5%	5905-16513
R8, R9	2	150Ω 1/4 W CARBON COMP. 5%	5905-16469
FORKED TERMINAL	18	CAMBION #180-7338-02-05	5940-59659
⑥	1	GOLD GROUND SCREW, CONTROL LOGIC NO. 183.050	5975-56592
⑤	1	SELF TAPPING SCREW, 3/8 L X 4-40, BLACK ANODIZE	5305-57101
④	7	TEST POINT PINS, CONTROL LOGIC NO. 183.164	5975-56591
③	1	LABEL, HANDLE, CONTROL LOGIC NO. 183.052	5975-568
②	1	HANDLE, CARD, CONTROL LOGIC NO. 183.163	5975-56590
LEA68-9070 99 ①	1	CARD, PRINTED CIRCUIT (ARTWORK 4 SHEETS, SIZE C)	
LED11020	–	ELECTRONIC FAB SPEC.	
LED11544	–	SOLDERING SPEC	
LED11591	–	P.C. BOARD PRODUCTION SPEC	
LED11606	–	P.C. EQUIPMENT SPEC	

COMPONENT DESIGNATION DRAWING OR SPECIFICATION	QTY REQD	DESCRIPTION	STOCK NO FOR INTERNAL USE ONLY

COMMERCIAL PARTS & MATERIALS

ALL PARTS SHALL BE THOSE LISTED ON THIS LIST UNLESS OTHERWISE SPECIFIED ON THIS SHEET

PARTS LIST

PIN CONNECTIONS

A	RD1	1	RDE 4
B	RD2	2	DA 17
C	RD3	3	RDA 18
D	RD4	4	RCP 17
E	RD5	5	SWE
F	RD6	6	OR 15
H	RD7	7	FER 14
J	RDB	8	PER 13
K	EXT. RESET	9	TBMT
L	DS	10	EOC
M	DB1	11	SDO
N	DB2	12	TCP
P	DB3	13	TTY RETURN
R	DB4	14	CLOCK
S	DB5	15	
T	DB6	16	
U	DB7	17	PRINTER
V	DB8	18	
W	KBD	19	
X	-9V	20	-9V
Y	+5V	21	+5V
Z	GND	22	GND

TEST TERMINALS

1		5	
2		6	
3		7	
4		8	

KEY POSITIONS

20

E	–	CHANGED NOTE 7			
D	–	SWITCHED IC3 PINS 5 & 6	6/73	C. H.	
C	–	ADDED C6 STK. NO.	1/73	C. H.	
B	–	ADDED STK. NO.	1/73	C. H.	
A	–	CHANGED LAYOUT OF IC3, IC4 & IC5	10-72	R. FIELDS	
REV	ZONE	DESCRIPTION (SEE REVISION SHEET FOR HISTORY)	DATE	DRAWN	APPROVED

DIV/GROUP EOED
ORIGINATOR EUGENE FISHER
DRAWN DATE R. FIELDS 10-9-72
CHECKED DATE 11/14/72
APPROVED DATE 11-14-72

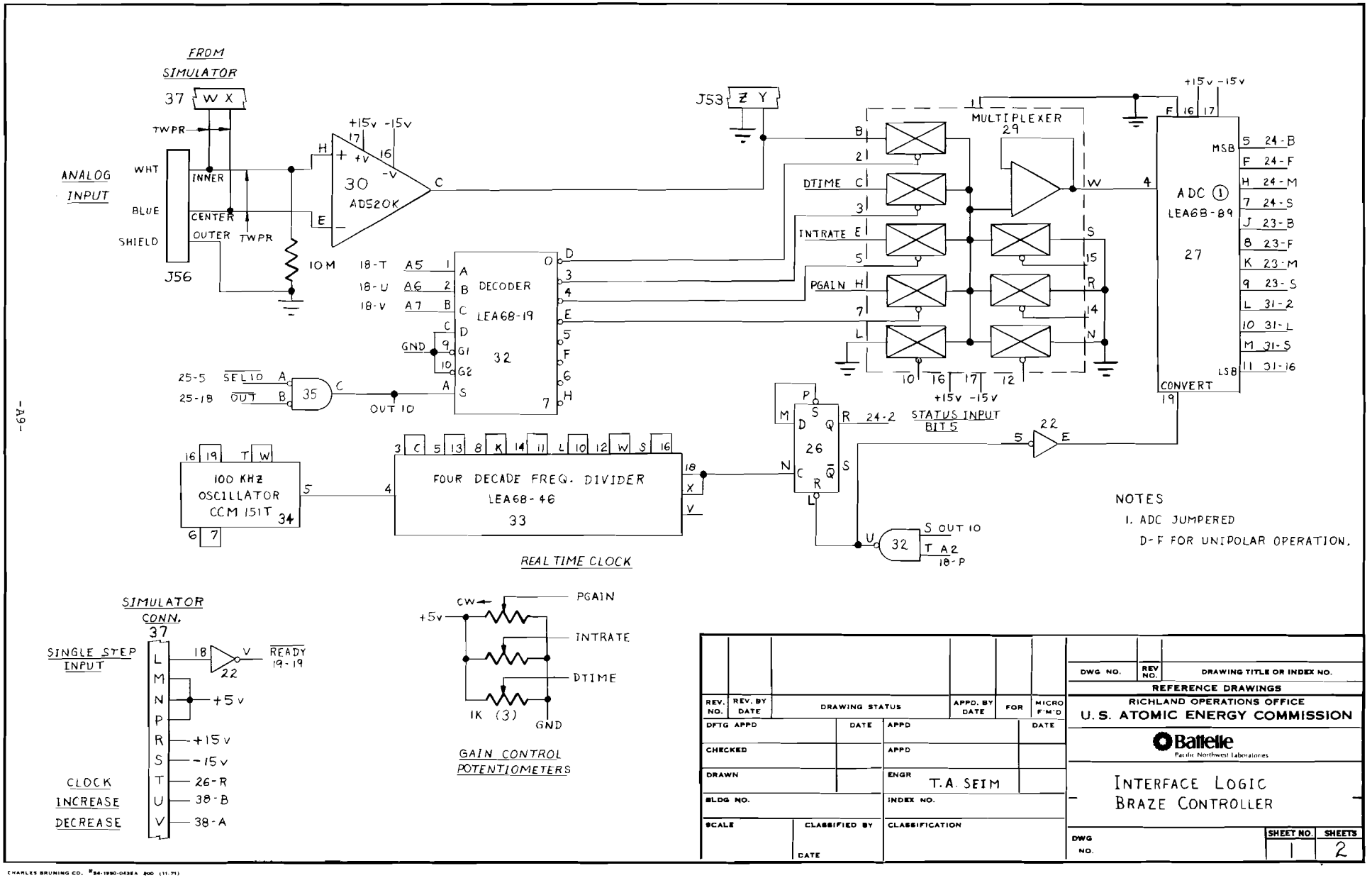
TITLE
UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER

UNIVERSITY OF CALIFORNIA
LAWRENCE RADIATION LABORATORY
ELECTRONICS ENGINEERING DEPARTMENT
LIVERMORE, CALIFORNIA 94550

GROUP	BASIC PACKAGE NO	SUB	TYPE	REV
LEA	68-9068	99	-A	E

SIZE C

SHEET 2 OF 2



NOTES

1. ADC JUMPERED D-F FOR UNIPOLAR OPERATION.

REV. NO.	REV. BY DATE	DRAWING STATUS	APPD. BY DATE	FOR	MICRO F'M'D
DFTG APPD	DATE	APPD	DATE		
CHECKED		APPD			
DRAWN		ENGR T.A. SEIM			
BLDG NO.		INDEX NO.			
SCALE	CLASSIFIED BY DATE	CLASSIFICATION			

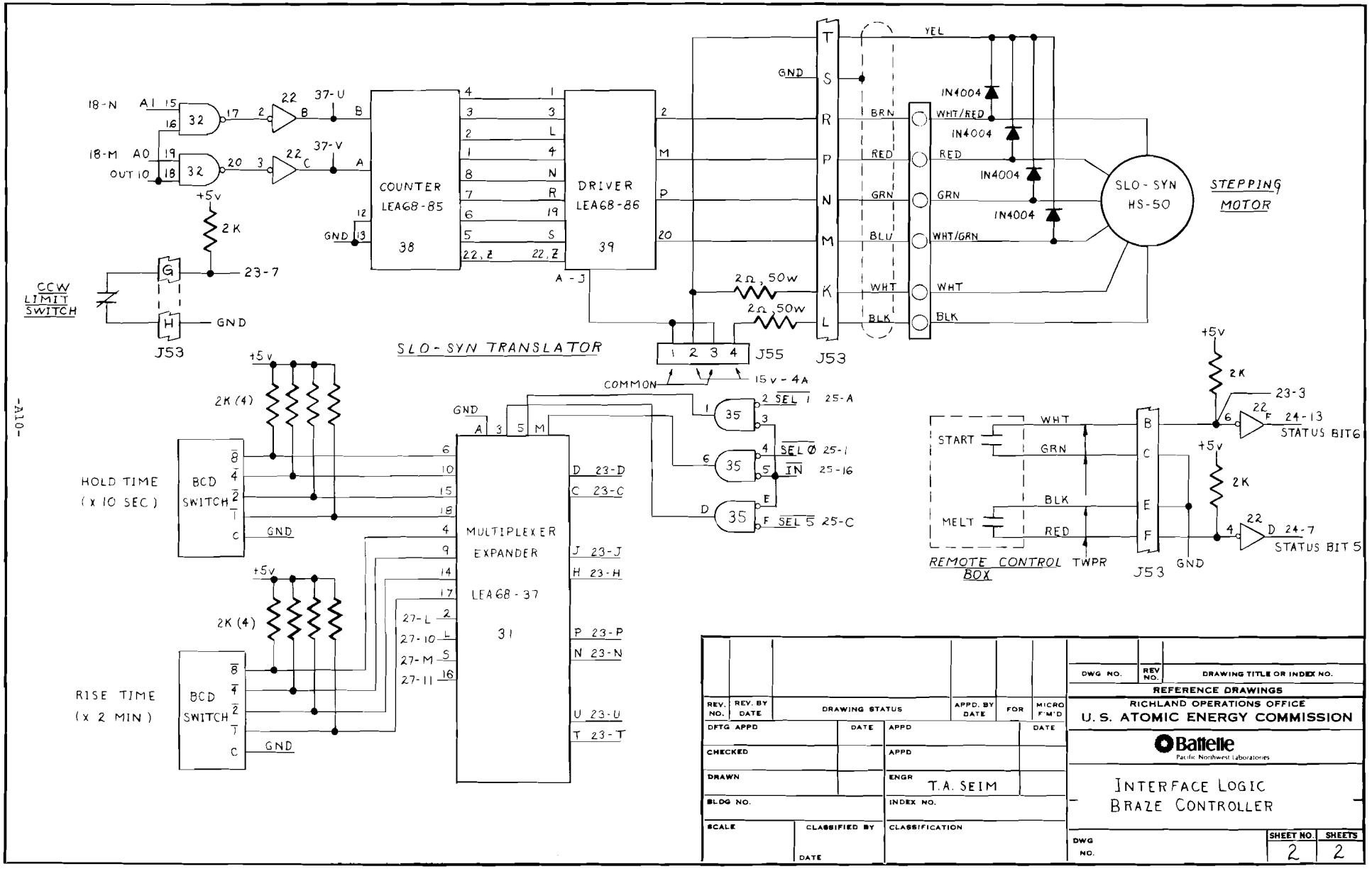
DWG NO.	REV NO.	DRAWING TITLE OR INDEX NO.
		REFERENCE DRAWINGS

RICHLAND OPERATIONS OFFICE
U.S. ATOMIC ENERGY COMMISSION
Battelle Pacific Northwest Laboratories

INTERFACE LOGIC BRAZE CONTROLLER

DWG NO.

SHEET NO.	SHEETS
1	2



REV. NO.	REV. BY DATE	DRAWING STATUS	APPD. BY DATE	FOR	MICRO F'M'D
DFTG APPD	DATE	APPD	DATE		
CHECKED		APPD			
DRAWN		ENGR T.A. SEIM			
BLDG NO.		INDEX NO.			
SCALE	CLASSIFIED BY DATE	CLASSIFICATION			

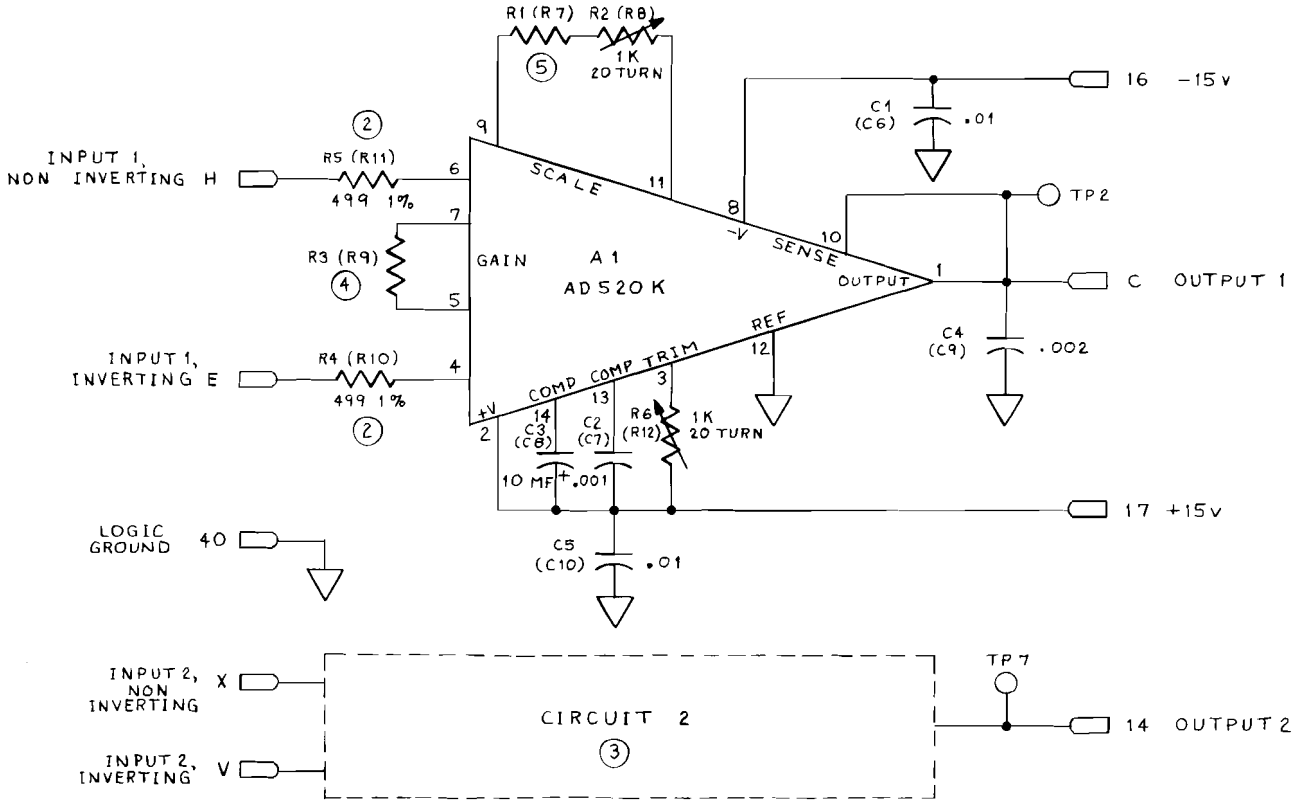
DWG NO.	REV NO.	DRAWING TITLE OR INDEX NO.
REFERENCE DRAWINGS		

RICHLAND OPERATIONS OFFICE
U.S. ATOMIC ENERGY COMMISSION
Battelle Pacific Northwest Laboratories

INTERFACE LOGIC BRAZE CONTROLLER

DWG NO.	SHEET NO.	SHEETS
	2	2

-A10-



NOTES

1. ALL FIXED RESISTORS ARE 1% METAL FILM, 50 PPM TC.
2. INPUT RESISTORS ARE MATCHED PAIRS.
3. PART NO.S IN PARENTHESIS ARE FOR CIRCUIT 2.
4. R3 AND R9 ARE BOURNS SELECTABLE FIXED RESISTOR.
5. NOMINAL GAIN IS DETERMINED BY R1(R7) / R3(R9). RANGE IS SET BY R1(R7), COARSE GAIN BY R3(R9), AND FINE GAIN BY R2(R8).

REV. NO.	REV. BY DATE	DRAWING STATUS	APPD. BY DATE	FOR	MICRO FILM'D
DFTG APPD	DATE	APPD	DATE		
CHECKED		APPD			
DRAWN		ENGR T. A. SEIM			
BLDG NO.		INDEX NO.			
SCALE	CLASSIFIED BY DATE	CLASSIFICATION			

DWG NO.	REV NO.	DRAWING TITLE OR INDEX NO.
		REFERENCE DRAWINGS

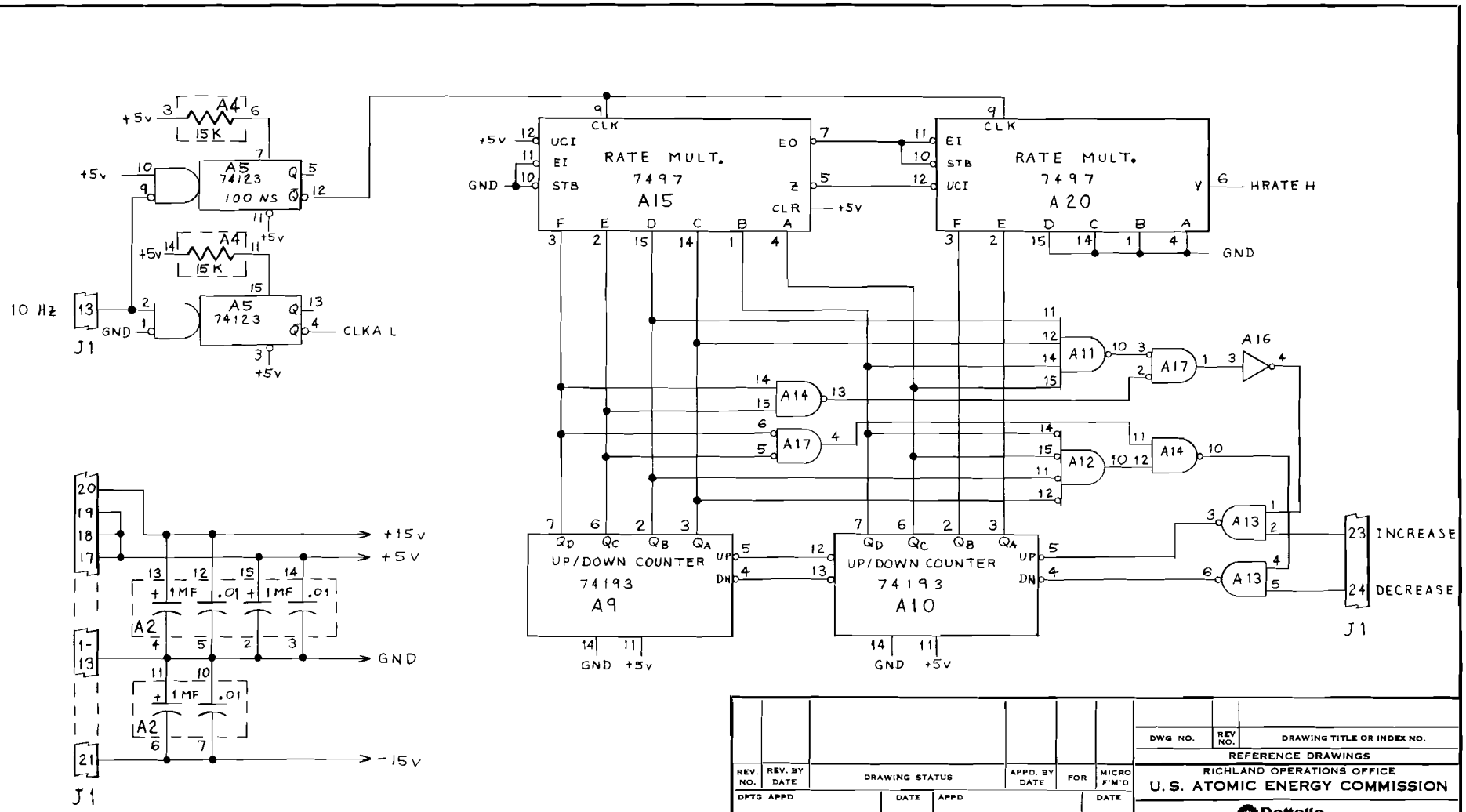
RICHLAND OPERATIONS OFFICE
U.S. ATOMIC ENERGY COMMISSION

Battelle
Pacific Northwest Laboratories

DUAL DIFFERENTIAL AMPLIFIER
4001 BNW

DWG NO.	SHEET NO.	SHEETS
	1	1

-ALL-



-A12-

REV. NO.	REV. BY DATE	DRAWING STATUS	APPD. BY DATE	FOR	MICRO F'M'D
DFTG APPD	DATE	APPD	DATE		
CHECKED		APPD			
DRAWN		ENGR T.A.SEIM			
BLDG NO.		INDEX NO.			
SCALE	CLASSIFIED BY DATE	CLASSIFICATION			

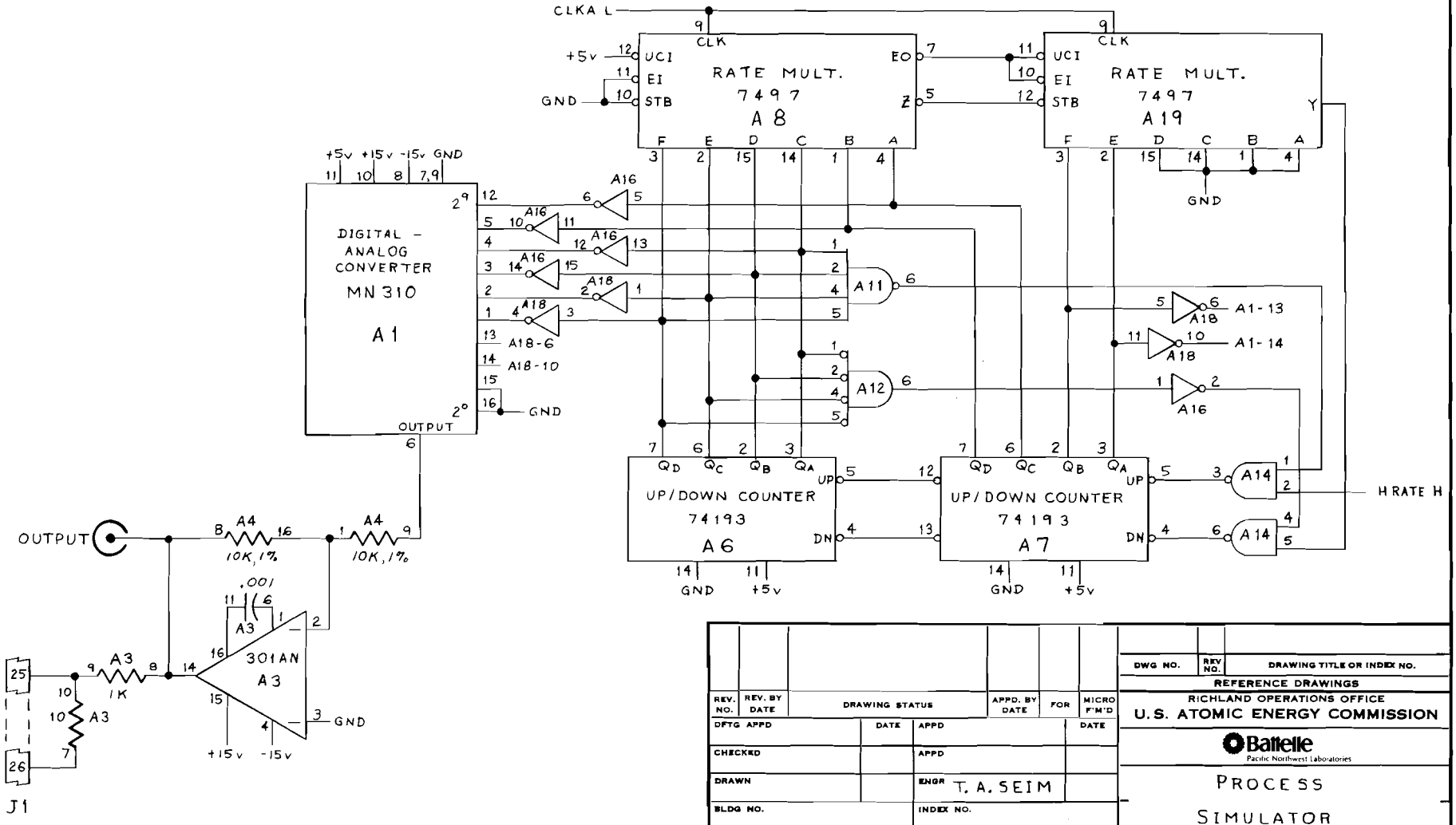
DWG NO.	REV NO.	DRAWING TITLE OR INDEX NO.
		REFERENCE DRAWINGS

RICHLAND OPERATIONS OFFICE
U.S. ATOMIC ENERGY COMMISSION

Battelle
Pacific Northwest Laboratories

PROCESS SIMULATOR

DWG NO.	SHEET NO.	SHEETS
	1	2



-A13-

REV. NO.	REV. BY DATE	DRAWING STATUS		APPD. BY DATE	FOR	MICRO F'M'D	DWG NO.	REV NO.	DRAWING TITLE OR INDEX NO.
							REFERENCE DRAWINGS		
							RICHLAND OPERATIONS OFFICE		
							U.S. ATOMIC ENERGY COMMISSION		
DFTG APPD		DATE	APPD		DATE		Battelle Pacific Northwest Laboratories		
CHECKED			APPD				PROCESS SIMULATOR		
DRAWN			ENGR	T. A. SEIM					
BLDG NO.			INDEX NO.						
SCALE	CLASSIFIED BY DATE		CLASSIFICATION				DWG NO.	SHEET NO. 2	SHEETS 2

APPENDIX B

CARD ASSIGNMENTS

POSITION	FUNCTION	LLL STOCK NUMBER	IDENTIFICATION
1	Crow-Bar Protection	5975-59644	72-02
15	PROM Memory	5975-59537	68-96
16	PROM Memory	5975-59537	68-96
17	RAM Memory	5975-59538	68-97
18	Memory Address Storage	5975-59536	68-95
19	Central Processor	5975-59534	68-93
20	Input-Output Control	5975-59535	68-94
21	Serial Interface	5975-59645	68-99
22	Inverters	5975-56521	IN-OT
23	Digital Multiplexer	5975-58407	68-33
24	Digital Multiplexer	5975-58407	68-33
25	4 to 16 Decoder	5975-59650	72-03
26	'D' Flip-Flops	5975-57276	DF-6T
27	Analog-Digital Converter	5975-59210	68-89
28	-		
29	Analog Multiplexer	5975-59020	68-64
30	Differential Amplifier	4001 BNW[1]	-
31	Multiplexer Expander	5975-58237	68-37
32	2-Input NAND Gates	5975-56517	NG-2T
33	Frequency Divider	5975-58250	68-46
34	Oscillator	5975-56523	CM-IT
35	2-Input NOR Gates	5975-56966	OG-2T
36			
37	Simulator Connector	5975-58215	-
38	Translator	5975-59202	68-85
39	Power Driver	5975-59203	68-86
40	-		

[1] *Designed and built by BNW.*

INPUT-OUTPUT ASSIGNMENTS

INPUT

I-O PORT	INSTRUCTION CODE[1]	DATA FIELD MSB	DATA FIELD LSB	FUNCTION
0	101	3	0	HOLD TIME Switch
2	103	3	0	RISE TIME Switch
4	105	7	0	UART, Receive Data
6	107			STATUS
		0		UART, Data Ready
		1		UART, XMT Ready
		2		CCW Limit Switch
		5		MELT Switch
		6		START Switch
		7		Real Time Clock
10	111	7	0	ADC, most significant bits
12	113	3	0	ADC, least significant bits

OUTPUT

20	121			Control Output
		0		Step, CCW
		1		Step, CW
		2		Reset Clock, Convert
		7	5	Analog Multiplexer

7 6 5	
0 0 0	Input Amplifier
0 0 1	Derivative Gain
0 1 0	Integral Gain
0 1 1	Proportional Gain

32	133	7	0	UART, Transmit Data

CONTROL OUTPUT CONNECTOR WIRING, J53

J53	CONTROLLER	DESTINATION	COLOR
B	22-6, 23-3	Remote Control Box, START p.b. N.O.	WHT
C	GROUND	Remote Control Box, START p.b. common	GRN
E	22-4	Remote Control Box, MELT p.b. N.O.	BLK
F	GROUND	Remote Control Box, MELT p.b. common	RED
G	23-7	Stepping Motor CCW Limit Switch, N.C.	
H	GROUND	Stepping Motor CCW Limit Switch, common	
K	Dropping Resistor	Stepping Motor Common, A	WHT
L	Dropping Resistor	Stepping Motor Common, B	BLK
M	39-20	Stepping Motor Phase 1	BLU
N	39-P	2	GRN
P	39-M	3	RED
R	39-2	4	BRN
S	GROUND	Stepping Motor Cable Shield	
T	J55-2	Clamping Diodes	YEL
Y	30-C, 29-B	Analog Out	
Z	GROUND	Analog Ground	

SIMULATOR CONNECTOR WIRING

SIMULATOR CONNECTOR J1	CONTROLLER CARD CONNECTOR 37	DESTINATION PIN	SIGNAL
1 to 13	40		GROUND
14	J		Unused
15	K		Unused
16	L		Unused
17	M	37 - 21	+5v
18	N	37 - 21	+5v
19	P	37 - 21	+5v
20	R	30 - 16	+15v
21	S	30 - 17	-15v
22	T	26-R	10 Hz CLOCK
23	U	38-B	OUTPUT, INCREASE
24	V	38-A	OUTPUT, DECREASE
25	W	30-H	ANALOG INPUT
26	X	30-E	ANALOG GROUND

APPENDIX C

```
0001                    *       TEST PROGRAMS
0002                    *     ASSEMBLY LISTING STARTS AT LOCATION 100 OF THE RAM
0003                    *     THIS IS PAST THE AREA USED (AND CLEARED) BY THE
0004                    *     CONTROL PROGRAM
0005                    *     THESE PROGRAMS MUST BE ENTERED WITH ODT
0006                    *
0007      04100         *     ORG   '4100
0010                    *
0011                    *     EXERCISES MUX-ADC AND REAL TIME CLOCK
0012                    *
0013                    *     HOLD TIME SWITCH SELECTS MUX CHANNEL
0014                    *           0 -- ANALOG INPUT
0015                    *           1 -- DERIVATIVE GAIN
0016                    *           2 -- INTEGRAL GAIN
0017                    *           3 -- PROPORTIONAL GAIN
0020                    *
0021      00050         SEND  EQU   '50         TELETYPE OUTPUT ROUTINE
0022      00060         READ  EQU   '60         TELETYPE INPUT ROUTINE
0023      00070         CRLF  EQU   '70         TYPES A CARRIAGE RETURN-LINE FEED
0024      00006         FLAG  EQU   6           STATUS DATA
0025      00000         HOLD  EQU   0           HOLD TIME SWITCH
0026      00020         CMAND EQU   '20         CONTROL OUTPUT
0027      00010         DATA  EQU   '10         ADC DATA, MOST SIG
0030      00012         LSDATA EQU  '12         ADC DATA, LEAST SIG
0031      00307         OCTALP EQU  '307        ODT OCTAL OUTPUT ROUTINE ADDRESS
0032                    *
0033  04100  107   MX1   INP   FLAG        TEST IF THE CLOCK IS SET
0034  04101  074         CPI   0           BIT 7 MUST BE ON
      04102  000
0035  04103  120         JFS   MX1
      04104  100
      04105  010
0036  04106  101         INP   HOLD        USE HOLD TIME SWITCH TO
0037  04107  012         RRC               SELECT THE MUX CHANNEL
0040  04110  012         RRC               POSITION BITS 0 - 3 TO 5 - 7
0041  04111  012         RRC
0042  04112  064         ORI   4           SET CONVERT BIT
      04113  004
0043  04114  121         OUT   CMAND       COMMAND CONVERSION
0044  04115  111         INP   DATA        GET CONVERTED DATA
0045  04116  106         CAL   OCTALP      TYPE IT OUT
      04117  307
      04120  000
0046  04121  075         RST   CRLF        CARRIAGE RETURN
0047  04122  104         JMP   MX1
      04123  100
      04124  010
0050                    *
0051                    *     FULL PRECISION ADC OUTPUT ROUTINE
0052                    *
0053                    *     OUTPUT APPEARS AS TWO OCTAL NUMBERS
0054                    *     THE FIRST NUMBER CONTAINS THE FIRST 8 SIG BITS
0055                    *     THE SECOND NUMBER CONTAINS THE 4 LEAST SIG BITS
0056                    *     THE FIRST DIGIT OF THE SECOND NUMBER IS NOT USED
0057                    *     EXAMPLE--FULL SCALE 7777 IS TYPED AS 377 074
0060                    *
0061  04125  006   ADC   LAI   4           CONVERT ANALOG INPUT
      04126  004
0062  04127  121         OUT   CMAND
```

```
0063  04130  111          INP   DATA        GET MOST SIGNIFICANT DATA
0064  04131  106          CAL   OCTALP      OUTPUT
      04132  307
      04133  000
0065  04134  113          INP   LSDATA      GET LEAST SIG DATA
0066  04135  012          RRC
0067  04136  012          RRC
0070  04137  106          CAL   OCTALP      OUTPUT
      04140  307
      04141  000
0071  04142  106          CAL   CRLF
      04143  070
      04144  000
0072  04145  104          JMP   ADC
      04146  125
      04147  010
0073                      *
0074                      *     SINGLE STEP MOTOR TEST
0075                      *
0076                      *     TELETYPE 'A' = CCW STEP
0077                      *              'B' = CW STEP
0100                      *
0101  04150  065    STEP  RST   READ        WAIT FOR INPUT
0102  04151  121          OUT   CMAND       USE CHARACTER AS COMMAND DATA
0103  04152  104          JMP   STEP
      04153  150
      04154  010
0104                      *
0105                      *     DYNAMIC MOTOR TEST
0106                      *
0107                      *     TYPING A CHARACTER CAUSES THE MOTOR TO MOVE 180
0110                      *     DEGREES, THE DIRECTION DEPENDENT UPON THE CHARACTER
0111                      *     TYPED (SEE ABOVE)
0112                      *
0113     00144    DELTA EQU   100         NO, OF STEPS MOVED
0114     00100    RATE  EQU   64          STEPPING RATE (64 USEC PER DIGIT)
0115                      *
0116  04155  026    S1    LCI   DELTA
      04156  144
0117  04157  065          RST   READ        WAIT FOR COMMAND
0120  04160  016    S2    LBI   RATE
      04161  100
0121  04162  011    S3    DCB               DECREMENT RATE UNTIL ZERO
0122  04163  110          JFZ   S3          TEST IF ZERO
      04164  162
      04165  010
0123  04166  121          OUT   CMAND       STEP MOTOR
0124  04167  021          DCC               DECREMENT STEP LENGTH
0125  04170  110          JFZ   S2
      04171  160
      04172  010
0126  04173  104          JMP   S1
      04174  155
      04175  010
```

```
0001                    *       BRAZE CONTROL PROGRAM -- TAS
0002                    *
0003                    *
0004      00010         RAM     EQU     '10         VARIABLE PARAMETER STORAGE
0005      00011         TBPAGE  EQU     '11         STATE TABLE PAGE (STARTING PAGE)
0006      00012         CONST   EQU     '12         MATH SUBROUTINE CONSTANTS
0007      00014         ROM     EQU     '14         PROFILE TABLE PAGE
0010      00000         HTSW    EQU     0           HOLD TIME SWITCH
0011      00002         RTSW    EQU     2           RISE TIME SWITCH
0012      00006         FLAG    EQU     6           STATUS BITS
0013      00010         DATA    EQU     '10         ADC PORT
0014      00020         STEP    EQU     '20         COMMAND INSTRUCTION
0015      00004         SCALE   EQU     4           SCALE FACTOR (POWER OF 2)
0016      00006         DSCALE  EQU     6           DERIVATIVE SCALE FACTOR
0017      00062         DELAY   EQU     50          STEPPING MOTOR RATE FACTOR
0020      04000                 ORG     '4000
0021   04000  000        DTIME   D       0           CURRENT DIFFERENTIAL GAIN
0022   04001  000        IRATE   D       0           CURRENT INTEGRATION RATE
0023   04002  000        PGAIN   D       0           CURRENT PROPORTIONAL GAIN
0024   04003  000        SIGMA   D       0
0025   04004  000                D       0
0026   04005  000        POINT   D       0           POINTER TO TEMP PROFILE
0027   04006  000        U(N)    D       0           CONTROLLER OUTPUT
0030   04007  000                D       0
0031   04010  000        S       D       0           PRESENT STATE
0032   04011  000        MN      D       0
0033   04012  000                D       0
0034   04013  000        MN1     D       0           PREVIOUS MEASURED VALUE
0035   04014  000                D       0
0036   04015  000        MN2     D       0
0037   04016  000                D       0
0040   04017  000        E       D       0           SETPOINT - MEAS. VAR.
0041   04020  000                D       0
0042   04021  000        E1      D       0
0043   04022  000                D       0
0044   04023  000        E2      D       0           E(N-2)
0045   04024  000                D       0
0046   04025  000        SLOPOS  D       0           SLO-SYN POSITION
0047   04026  000                D       0
0050   04027  000        SP1     D       0           PREVIOUS SETPOINT
0051   04030  000        ICLK    D       0
0052   04031  000        ICAND   D       0           MULTIPLICAND REGISTER
0053   04032  000        MT1     D       0           MATH SUBR. TEMPORARIES
0054   04033  000        MT2     D       0
0055   04034  000        HOLD    D       0           TEMPERATURE HOLD TIMEOUT
0056   04035  000                D       0
0057   04036  000        RISE    D       0
0060                    *
0061      04400                 ORG     '4400
0062   04400  056        START   LHI     RAM
       04401  010
0063   04402  066                LLI     S           SET TO PRESENT STATE
       04403  010
0064   04404  250                XRA
0065   04405  370                LMA                 SET PRESENT STATE TO ZERO
0066   04406  066                LLI     ICLK        INTEGRATOR TIMEOUT
       04407  030
0067   04410  076                LMI     10
       04411  012
```

```
0070  04412  104          JMP   INIT        INITIALIZE CONTROLLER
      04413  145
      04414  011
0071  04415  016   CLOCK  LBI   0           INITIALIZE SWITCH STATUS
      04416  000
0072  04417  107   CLK1   INP   FLAG        WAIT FOR REAL TIME CLOCK
0073  04420  261          ORB
0074  04421  310          LBA               SAVE STATUS FOR SWITCH TEST
0075  04422  002          RLC
0076  04423  100          JFC   CLK1
      04424  017
      04425  011
0077  04426  006          LAI   4
      04427  004
0100  04430  121          OUT   STEP        RESET REAL TIME CLOCK
0101  04431  301          LAB               TEST CONTROL SWITCHES
0102  04432  064          ORI   '10
      04433  010
0103  04434  016          LBI   0
      04435  000
0104  04436  002          RLC               BYPASS REAL TIME CLOCK BIT
0105  04437  010   SWITCH INB
0106  04440  002          RLC
0107  04441  100          JFC   SWITCH
      04442  037
      04443  011
0110  04444  301          LAB
0111  04445  106          CAL   STAT        EXECUTE STATE CONTROL
      04446  074
      04447  011
0112  04450  010          INB               PRIME STATE FOR TEST
0113  04451  011          DCB
0114  04452  150          JTZ   INIT
      04453  145
      04454  011
0115  04455  011          DCB
0116  04456  150          JTZ   WAIT        WAIT FOR START
      04457  232
      04460  011
0117  04461  011          DCB
0120  04462  150          JTZ   CYCLE       EXECUTE TEMPERATURE CYCLE
      04463  235
      04464  011
0121  04465  011          DCB
0122  04466  150          JTZ   MELT
      04467  032
      04470  012
0123  04471  104          JMP   CLOCK       WAIT FOR NEXT SAMPLING PERIOD
      04472  015
      04473  011
0124                      *
0125                      *     STATE TABLE EXECUTION ROUTINE
0126                      *     A = INPUT TOKEN (0 - 3)
0127                      *
0130  04474  074   STAT   CPI   4           TEST RANGE OF INPUT TOKEN
      04475  004
0131  04476  320          LCA
0132  04477  056          LHI   RAM
      04500  010
0133  04501  066          LLI   S
```

```
      04502  010
0134  04503  317          LBM                GET PRESENT STATE
0135  04504  301          LAB
0136  04505  003          RFC                IGNORE IF OUT OF RANGE
0137  04506  002          RLC                MULT. BY 4 (4 ENTRIES PER STATE)
0140  04507  002          RLC
0141  04510  202          ADC                ADD TOKEN DISPLACEMENT
0142  04511  004          ADI    TABLE       COMPUTE FULL ADDRESS
      04512  125
0143  04513  056          LHI    TBPAGE
      04514  011
0144  04515  360          LLA
0145  04516  317          LBM                GET NEXT STATE
0146  04517  066          LLI    S
      04520  010
0147  04521  056          LHI    RAM
      04522  010
0150  04523  371          LMB                SET PRESENT STATE TO NEXT STATE
0151  04524  007          RET
0152                 *     TRANSITION TABLE--FOUR ENTRIES PER STATE
0153                 *
0154  04525  001    TABLE  D      1
0155  04526  000          D      0
0156  04527  000          D      0
0157  04530  000          D      0
0160  04531  001          D      1
0161  04532  002          D      2
0162  04533  001          D      1
0163  04534  000          D      0
0164  04535  002          D      2
0165  04536  002          D      2
0166  04537  003          D      3
0167  04540  000          D      0
0170  04541  003          D      3
0171  04542  003          D      3
0172  04543  003          D      3
0173  04544  000          D      0
0174                 *
0175                 *     INITIALIZATION ROUTINE
0176                 *
0177  04545  056    INIT   LHI    RAM
      04546  010
0200  04547  250          XRA                CLEAR A
0201  04550  360          LLA
0202  04551  016          LBI    40
      04552  050
0203  04553  370    NT1    LMA                CLEAR RAM MEMORY
0204  04554  060          INL                ADVANCE ADDRESS
0205  04555  011          DCB
0206  04556  110          JFZ    NT1
      04557  153
      04560  011
0207                 *     FIRST RESET SLO-SYN POSITION
0210  04561  026          LCI    200         MAX. NO. OF STEPS
      04562  310
0211  04563  016    L02    LBI    90
      04564  132
0212  04565  011    L01    DCB
0213  04566  110          JFZ    L01
      04567  165
```

```
      04570  011
0214  04571  107          INP   FLAG        GET STATUS BITS
0215  04572  012          RRC               POSITION LIMIT SWITCH IN CARRY FF
0216  04573  012          RRC
0217  04574  012          RRC
0220  04575  140          JTC   L03         STOP MOTION IF LIMIT REACHED
      04576  207
      04577  011
0221  04600  006          LAI   1
      04601  001
0222  04602  121          OUT   STEP        OUTPUT STEP COMMAND
0223  04603  021          DCC
0224  04604  110          JFZ   L02
      04605  163
      04606  011
0225  04607  106    L03   CAL   TTO
      04610  063
      04611  012
0226                *      INITIALIZE HOLD TIME DELAY
0227  04612  066          LLI   HOLD
      04613  034
0230  04614  076          LMI   100         10 SEC DELAY
      04615  144
0231  04616  101          INP   HTSW        GET HOLD SWITCH SETTING
0232  04617  060          INL
0233  04620  106          CAL   TTO1        MULTIPLY BY SIX
      04621  066
      04622  012
0234  04623  250          XRA
0235  04624  106          CAL   STAT        ADVANCE STATE TO WAIT STATE
      04625  074
      04626  011
0236  04627  104          JMP   CLOCK
      04630  015
      04631  011
0237                *
0240  04632  104    WAIT  JMP   CLOCK       NO OPERATION--WAIT FOR START OF CYCLE
      04633  015
      04634  011
0241                *
0242                *      TEMPERATURE CYCLE CONTROL
0243                *
0244  04635  066    CYCLE LLI   RISE        GET RISE TIME VARIABLE
      04636  036
0245  04637  317          LBM
0246  04640  011          DCB               DECREMENT TIMEOUT (INITIAL VALUE DETERMINES RATE
0247  04641  371          LMB               OF RISE)
0250  04642  110          JFZ   L20
      04643  262
      04644  011
0251  04645  066          LLI   POINT       ADVANCE PROFILE TABLE POINTER
      04646  005
0252  04647  317          LBM
0253  04650  010          INB
0254  04651  371          LMB
0255  04652  110          JFZ   CY1         TEST FOR POINTER OVERFLOW
      04653  257
      04654  011
0256  04655  076          LMI   255         DO NOT ALLOW TO OVERFLOW
      04656  377
```

```
0257  04657  106   CY1   CAL   TTO        RE-INITIALIZE RISE RATE TIMEOUT
      04660  063
      04661  012
0260                 *
0261                 *     CONTROL RESPONSE CALCULATION
0262                 *
0263                 *     CONTROL IS A COMBINED PROPORTIONAL-INTEGRAL (PI) ACTION
0264                 *
0265  04662  106   L20   CAL   FILTER     DIGITAL FILTERING ROUTINE
      04663  265
      04664  012
0266  04665  066         LLI   DTIME
      04666  000
0267  04667  006         LAI   '44        FIND DIFF. GAIN SETTING BY CONVERTING
      04670  044
0270  04671  121         OUT   STEP       POTENTIOMETER SETTING
0271  04672  111         INP   DATA
0272  04673  370         LMA
0273  04674  060         INL
0274  04675  006         LAI   '104       FIND INTEGRATION RATE
      04676  104
0275  04677  121         OUT   STEP
0276  04700  111         INP   DATA
0277  04701  370         LMA
0300  04702  060         INL
0301  04703  006         LAI   '144       FIND PROPORTIONAL GAIN
      04704  144
0302  04705  121         OUT   STEP
0303  04706  111         INP   DATA
0304  04707  370         LMA
0305  04710  106         CAL   PROP       PROPORTIONAL RESPONSE CALCULATION
      04711  074
      04712  012
0306  04713  106         CAL   INT        INTEGRAL RESPONSE CALCULATION
      04714  120
      04715  012
0307  04716  106         CAL   DIFF       DERIVATIVE RESPONSE CALCULATION
      04717  224
      04720  012
0310  04721  066         LLI   U(N)       GET CONTROL RESPONSE
      04722  006
0311  04723  317         LBM
0312  04724  060         INL
0313  04725  327         LCM
0314  04726  066         LLI   MAX
      04727  220
0315  04730  056         LHI   CONST
      04731  012
0316  04732  106         CAL   LIMIT      CHECK BOUNDS ON CONTROL
      04733  275
      04734  013
0317  04735  056         LHI   RAM
      04736  010
0320  04737  066         LLI   U(N)
      04740  006
0321  04741  371         LMB
0322  04742  060         INL
0323  04743  372         LMC
0324  04744  066         LLI   SLOPOS
      04745  025
```

```
0325  04746  317          LBM              GET PRESENT SLO-SYN POSITION
0326  04747  060          INL
0327  04750  327          LCM
0330  04751  066    L22   LLI   U(N)
      04752  006
0331  04753  106          CAL   DCPM       COMPARE WITH COMMANDED POSITION
      04754  315
      04755  013
0332  04756  150          JTZ   L28        EQUAL
      04757  022
      04760  012
0333  04761  160          JTS   L24        LESS THAN, STEP UP
      04762  001
      04763  012
0334  04764  021          DCC              GREATER THAN, STEP DOWN
0335  04765  020          INC              TEST IF C = 0 (DECR, WILL CAUSE A BORROW)
0336  04766  110          JFZ   L23
      04767  372
      04770  011
0337  04771  011          DCB              PASS CARRY INTO MS HALF
0340  04772  021    L23   DCC              DECREMENT LEAST SIG HALF
0341  04773  006          LAI   1          STEP DOWN
      04774  001
0342  04775  121          OUT   STEP       OUTPUT STEP COMMAND
0343  04776  104          JMP   L26
      04777  011
      05000  012
0344  05001  020    L24   INC
0345  05002  110          JFZ   L25
      05003  006
      05004  012
0346  05005  010          INB              PASS OVERFLOW INTO MS HALF
0347  05006  006    L25   LAI   2          STEP UP
      05007  002
0350  05010  121          OUT   STEP       OUTPUT STEP COMMAND
0351  05011  036    L26   LDI   DELAY
      05012  062
0352  05013  031    L27   DCD
0353  05014  110          JFZ   L27
      05015  013
      05016  012
0354  05017  104          JMP   L22
      05020  351
      05021  011
0355  05022  066    L28   LLI   SLOPOS     SET NEW SLO-SYN POSITION
      05023  025
0356  05024  371          LMB
0357  05025  060          INL
0360  05026  372          LMC
0361  05027  104          JMP   CLOCK
      05030  015
      05031  011
0362                *
0363                *      HOLDS AT CURRENT TEMPERATURE FOR SPECIFIED DELAY AFTER MELT OCCURS
0364                *
0365  05032  066    MELT  LLI   HOLD
      05033  034
0366  05034  317          LBM              GET TIME OUT (HOLD)
0367  05035  011          DCB
0370  05036  371          LMB
```

```
0371  05037  110          JFZ   L20         TEST IF TIMED OUT
      05040  262
      05041  011
0372  05042  076          LMI   100         10 SEC DELAY
      05043  144
0373  05044  060          INL
0374  05045  317          LBM               GET HOLD TIMER
0375  05046  011          DCB
0376  05047  371          LMB
0377  05050  110          JFZ   L20
      05051  262
      05052  011
0400  05053  006          LAI   3           TIMED OUT--RESET STATE
      05054  003
0401  05055  106          CAL   STAT
      05056  074
      05057  011
0402  05060  104          JMP   CLOCK
      05061  015
      05062  011
0403                  *
0404                  *    CALCULATES TEMP, TIME OUT
0405  05063  066    TTO   LLI   RISE
      05064  036
0406  05065  103          INP   RTSW        GET RISE TIME SWITCH SETTING
0407  05066  310    TTO1  LBA               ENTRY FOR HOLD TIME CALCULATION
0410  05067  002          RLC
0411  05070  201          ADB
0412  05071  002          RLC               MULT BY SIX
0413  05072  370          LMA
0414  05073  007          RET
0415                  *
0416                  *    PROPORTIONAL CONTROL SUBROUTINE
0417                  *
0420                  *    COMPUTES K(P)*(E(N)-E(N-1))
0421                  *
0422  05074  066    PROP  LLI   E
      05075  017
0423  05076  317          LBM               GET CONTROL ERROR
0424  05077  066          LLI   PGAIN       GET PROPORTIONAL GAIN
      05100  002
0425  05101  327          LCM
0426  05102  106          CAL   MULS
      05103  126
      05104  013
0427  05105  006          LAI   SCALE
      05106  004
0430  05107  106          CAL   DSRA        SCALE PROPORTIONAL OUTPUT
      05110  051
      05111  013
0431  05112  066          LLI   U(N)
      05113  006
0432  05114  371          LMB
0433  05115  060          INL
0434  05116  372          LMC
0435  05117  007          RET
0436                  *
0437                  *    INTEGRAL, OR RESET, CONTROL SUBROUTINE
0440                  *
0441  05120  066    INT   LLI   ICLK        GET TIMEOUT
```

```
      05121  030
0442  05122  317          LBM
0443  05123  011          DCB
0444  05124  371          LMB
0445  05125  150          JTZ    INT1        CONTINUE INTEGRATING
      05126  140
      05127  012
0446  05130  066          LLI    SIGMA
      05131  003
0447  05132  317          LBM
0450  05133  060          INL
0451  05134  327          LCM
0452  05135  104          JMP    INT2        OUTPUT INTEGRAL CONTROL
      05136  176
      05137  012
0453  05140  076    INT1  LMI    10
      05141  012
0454  05142  066          LLI    E
      05143  017
0455  05144  317          LBM                GET CONTROL ERROR
0456  05145  066          LLI    IRATE       GET INTEGRATION RATE
      05146  001
0457  05147  327          LCM
0460  05150  106          CAL    MULS
      05151  126
      05152  013
0461  05153  066          LLI    SIGMA
      05154  003
0462  05155  106          CAL    DADD        ADD TO PREVIOUS INTEGRAL
      05156  104
      05157  013
0463  05160  066          LLI    IMAX        TEST FOR SATURATION
      05161  214
0464  05162  056          LHI    CONST
      05163  012
0465  05164  106          CAL    LIMIT       OUTPUT COMPARED WITH 2 SUCCESSIVE LIMITS
      05165  275
      05166  013
0466  05167  056          LHI    RAM
      05170  010
0467  05171  066          LLI    SIGMA
      05172  003
0470  05173  371          LMB                SAVE RESULT
0471  05174  060          INL
0472  05175  372          LMC
0473  05176  006    INT2  LAI    8
      05177  010
0474  05200  106          CAL    DSRA
      05201  051
      05202  013
0475  05203  066          LLI    U(N)
      05204  006
0476  05205  106          CAL    DADD        COMBINE CONTROL ACTIONS
      05206  104
      05207  013
0477  05210  371          LMB
0500  05211  060          INL
0501  05212  372          LMC
0502  05213  007          RET
0503  05214  077    IMAX  D      177         UPPER INTEGRATOR LIMIT
```

```
0504  05215  377          D     '777
0505  05216  000   IMIN   D     0          LOWER INTEGRATOR LIMIT
0506  05217  000          D     0
0507  05220  000   MAX    D     0          CONTROLLER OUTPUT LIMIT
0510  05221  144          D     100        180 DEGREES (100 STEPS)
0511  05222  000   MIN    D     0          MIN = 0
0512  05223  000          D     0
0513                *
0514                *      DERIVATIVE RESPONSE ROUTINE
0515                *
0516  05224  066   DIFF   LLI   E1         GET CURRENT ERROR
      05225  021
0517  05226  317          LBM
0520  05227  060          INL
0521  05230  327          LCM
0522  05231  060          INL              ADDRESS NOW AT PREVIOUS ERROR
0523  05232  106          CAL   DSUB       (E(N) - E(N-1))*8
      05233  115
      05234  013
0524  05235  106          CAL   CDSI
      05236  220
      05237  013
0525  05240  310          LBA
0526  05241  066          LLI   DTIME      GET DIFFERENTIAL GAIN
      05242  000
0527  05243  327          LCM
0530  05244  106          CAL   MULS       K(D)*(E(N) - (N-1))*8
      05245  126
      05246  013
0531  05247  006          LAI   DSCALE
      05250  006
0532  05251  106          CAL   DSRA       SCALE RESULT
      05252  051
      05253  013
0533  05254  066          LLI   U(N)
      05255  006
0534  05256  106          CAL   DADD       COMBINE WITH OTHER CONTROL ACTIONS
      05257  104
      05260  013
0535  05261  371          LMB
0536  05262  060          INL
0537  05263  372          LMC
0540  05264  007          RET
0541                *
0542                *      DIGITAL FILTERING ROUTINE
0543                *      M(N) = K1*M(N) + K2*M(N-1)
0544                *      K1 = 1/8    K2 = 7/8
0545                *      CUTOFF FREQ, = .199 HZ
0546                *
0547  05265  066   FILTER LLI  E1
      05266  021
0550  05267  317          LBM
0551  05270  060          INL
0552  05271  327          LCM
0553  05272  060          INL
0554  05273  371          LMB              REPLACE E(N-2) WITH E(N-1)
0555  05274  060          INL
0556  05275  372          LMC
0557  05276  066          LLI   MN
      05277  011
```

```
0560  05300  317          LBM
0561  05301  060          INL
0562  05302  327          LCM
0563  05303  060          INL
0564  05304  371          LMB              SET M(N-1) = M(N)
0565  05305  060          INL
0566  05306  372          LMC
0567  05307  061          DCL
0570  05310  006          LAI   3
      05311  003
0571  05312  106          CAL   DSRA       MULT BY 1/8
      05313  051
      05314  013
0572                  *    M(N) IS SCALED BY 8
0573  05315  106          CAL   DSUB       -K2*M(N-1)
      05316  115
      05317  013
0574  05320  106          CAL   DCMP
      05321  006
      05322  013
0575  05323  250          XRA
0576  05324  066          LLI   MN
      05325  011
0577  05326  370          LMA
0600  05327  111          INP   DATA       SET M(N) = TEMP IN DOUBLE PRECISION FORMAT
0601  05330  060          INL
0602  05331  370          LMA
0603  05332  061          DCL
0604  05333  106          CAL   DADD       M(N) = K1*M(N) + K2*M(N-1)
      05334  104
      05335  013
0605                  *    K1*M(N) IS AUTO, DONE BECAUSE OF SCALING ON M(N-1)
0606  05336  371          LMB
0607  05337  060          INL
0610  05340  372          LMC
0611  05341  066          LLI   POINT      GET CURRENT SETPOINT
      05342  005
0612  05343  367          LLM
0613  05344  056          LHI   ROM
      05345  014
0614  05346  327          LCM
0615  05347  016          LBI   0          SET UP AS DOUBLE PRECISION VALUE
      05350  000
0616  05351  006          LAI   3
      05352  003
0617  05353  106          CAL   DSLL       SCALE BY 8
      05354  255
      05355  013
0620  05356  056          LHI   RAM
      05357  010
0621  05360  066          LLI   MN
      05361  011
0622  05362  106          CAL   DSUB       ERROR = SETPOINT - M(N)
      05363  115
      05364  013
0623  05365  066          LLI   E1         SAVE SCALED ERROR
      05366  021
0624  05367  371          LMB
0625  05370  060          INL
0626  05371  372          LMC
```

```
0627  05372  006          LAI   3
      05373  003
0630  05374  106          CAL   DSRA        RESTORE NO, WEIGHT OF 1
      05375  051
      05376  013
0631  05377  106          CAL   CDSI        CONVERT TO SINGLE PRECISION
      05400  220
      05401  013
0632  05402  066          LLI   E
      05403  017
0633  05404  370          LMA
0634  05405  007          RET
0635                  *
0636                  *    COMPLEMENTS DOUBLE PRECISION INTEGER
0637                  *
0640  05406  250    DCMP  XRA
0641  05407  222          SUC
0642  05410  320          LCA
0643  05411  006          LAI   0
      05412  000
0644  05413  231          SBB
0645  05414  310          LBA
0646  05415  007          RET
0647                  *
0650                  *    DOUBLE LENGTH SHIFT RIGHT
0651                  *    LEFT HALF IN B
0652                  *    RIGHT HALF IN C
0653                  *    NO OF SHIFTS IN A
0654                  *
0655  05416  074    DLSR  CPI   16          COMPARE FOR MAX NO OF SHIFTS
      05417  020
0656  05420  140          JTC   DLS1
      05421  027
      05422  013
0657  05423  016          LBI   0           CLEAR BOTH
      05424  000
0660  05425  321          LCB
0661  05426  007          RET
0662  05427  056    DLS1  LHI   RAM
      05430  010
0663  05431  074          CPI   0
      05432  000
0664  05433  053          RTZ
0665  05434  330          LDA
0666  05435  250    DLS2  XRA               CLEAR CARRY FF
0667  05436  301          LAB
0670  05437  032          RAR               SHIFT LEFT HALF RIGHT
0671  05440  310          LBA
0672  05441  302          LAC
0673  05442  032          RAR               SHIFT RIGHT HALF
0674  05443  320          LCA
0675  05444  031          DCD               EXECUTE INDEX
0676  05445  110          JFZ   DLS2        TEST INDEX
      05446  035
      05447  013
0677  05450  007          RET
0700                  *
0701                  *    ARITHMETIC DOUBLE SHIFT RIGHT
0702                  *    ARGUMENT IS IN B AND C
0703                  *    NO OF SHIFTS IN A
```

```
0704                    *
0705  05451  330   DSRA  LDA                 NO OF SHIFTS
0706  05452  074         CPI   0
      05453  000
0707  05454  053         RTZ
0710  05455  301         LAB                 INVERT SIGN FOR ARITH, SHIFT PURPOSES
0711  05456  054         XRI   '200
      05457  200
0712  05460  310         LBA
0713  05461  301   DSR1  LAB
0714  05462  054         XRI   '200          CORRECT SIGN FROM PREVIOUS SHIFT
      05463  200
0715  05464  074         CPI   '200          SET CARRY TO INVERSE OF SIGN
      05465  200
0716  05466  032         RAR                 SHIFT MOST SIG HALF
0717  05467  310         LBA
0720  05470  302         LAC                 GET RIGHT HALF AND SHIFT
0721  05471  032         RAR
0722  05472  320         LCA
0723  05473  031         DCD                 ADVANCE INDEX
0724  05474  110         JFZ   DSR1          TEST FOR COMPLETION
      05475  061
      05476  013
0725  05477  301         LAB
0726  05500  054         XRI   '200          MAKE FINAL CORRECTION
      05501  200
0727  05502  310         LBA
0730  05503  007         RET
0731                    *
0732                    *     DOUBLE PRECISION ADDITION SUBR
0733                    *           B = MOST SIG HALF OF ADDEND
0734                    *           C = LEST SIG HALF
0735                    *           M = ADDER
0736                    *     SUM IS LEFT IN B AND C
0737                    *
0740  05504  060   DADD  INL
0741  05505  302         LAC                 GET LOWER HALF
0742  05506  207         ADM                 COMPUTE LEAST SIG SUM
0743  05507  320         LCA                 SAVE RESULT
0744  05510  061         DCL
0745  05511  301         LAB
0746  05512  217         ACM                 COMPUTE MOST SIGNIFICANT SUM
0747  05513  310         LBA
0750  05514  007         RET
0751                    *
0752                    *     DOUBLE PRECISION SUBTRACTION
0753                    *     CONTENTS OF MEMORY ARE SUBTRACTED FROM DP INTEGER STORED
0754                    *     IN B AND C
0755                    *
0756  05515  060   DSUB  INL
0757  05516  302         LAC                 GET RIGHT HALF
0760  05517  227         SUM
0761  05520  320         LCA
0762  05521  061         DCL
0763  05522  301         LAB
0764  05523  237         SBM                 SUBTRACT WITH BORROW
0765  05524  310         LBA
0766  05525  007         RET
0767                    *
0770                    *     SINGLE PRECISION SIGNED MULTIPLY ROUTINE
```

```
0771                    *
0772                    *       TIME-- 1074 TO 1490 MICROSECONDS
0773   05526   250      MULS    XRA
0774   05527   331              LDB
0775   05530   340              LEA
0776   05531   222              SUC                     CONVERT ARGUMENTS TO ABSOLUTE VALUE
0777   05532   160              JTS     ML0
       05533   142
       05534   013
1000   05535   150              JTZ     ML0
       05536   142
       05537   013
1001   05540   320              LCA
1002   05541   040              INE
1003   05542   250      ML0     XRA
1004   05543   223              SUD
1005   05544   160              JTS     ML1
       05545   154
       05546   013
1006   05547   150              JTZ     ML1
       05550   154
       05551   013
1007   05552   330              LDA
1010   05553   040              INE
1011   05554   304      ML1     LAE                     SET CARRY EQUAL TO MOD 2 SUM OF SIGN BITS
1012   05555   032              RAR
1013   05556   106              CAL     UMUL            MULTIPLY POSITIVE ARG'S
       05557   165
       05560   013
1014   05561   142              CTC     DCMP            COMPLEMENT IF RESULT IS NEGATIVE
       05562   006
       05563   013
1015   05564   007              RET
1016                    *
1017                    *       UNSIGNED MULTIPLY
1020                    *       ARG'S IN C AND D
1021                    *       RESULT IN B AND C
1022                    *       REG'S--A,B,L, AND FLAGS EXCEPT CARRY
1023                    *       UMLS--MULTI-PRECISION ENTRY
1024                    *       (B,C) = C * D + B
1025   05565   016      UMUL    LBI     0
       05566   000
1026   05567   046      UMLS    LEI     9
       05570   011
1027   05571   302      UML0    LAC
1030   05572   032              RAR                     ROTATE CARRY INTO PRODUCT - MULTIPLIER
1031   05573   320              LCA                     PARTIAL PRODUCT, LSB MULTIPLIER BIT
1032   05574   041              DCE                     FORCED INTO CARRY
1033   05575   053              RTZ                     RETURN AFTER 8 LOOPS
1034   05576   301              LAB
1035   05577   100              JFC     UML1            IF CARRY ADD MULTIPLICAND TO PRODUCT
       05600   203
       05601   013
1036   05602   203              ADD
1037   05603   032      UML1    RAR                     ROTATE LSB INTO CARRY
1040   05604   310              LBA
1041   05605   104              JMP     UML0
       05606   171
       05607   013
1042                    *
```

```
1043                    *       CONVERTS SINGLE PRECISION INTEGERS TO DOUBLE PRECISION
1044                    *       RESULT IN B AND C
1045                    *       ARGUMENT IS IN A
1046                    *
1047  05610  320       CSDI    LCA
1050  05611  016               LBI     0               INITIALIZE HI OREDER HALF
      05612  000
1051  05613  271               CPB                     TEST SIGN OF ARGUMENT
1052  05614  023               RFS                     RETURN IF POSITIVE
1053  05615  016               LBI     '377            MAKE MOST SIG HALF NEG
      05616  377
1054  05617  007               RET
1055                    *
1056                    *       CONVERTS DOUBLE PRECISION INTEGERS TO SINGLE PRECISION
1057                    *       ARGUMENT IS IN B AND C
1060                    *       RESULT IS LEFT IN A
1061                    *
1062  05620  301       CDSI    LAB
1063  05621  074               CPI     0               TEST SIGN
      05622  000
1064  05623  120               JFS     CDS1
      05624  243
      05625  013
1065  05626  054               XRI     '377            INVERT MOST SIG BITS
      05627  377
1066  05630  006               LAI     -127            INITIALIZE RESULT TO LOWER BOUND
      05631  201
1067  05632  013               RFZ                     RETURN IF LESS THAN -256
1070  05633  302               LAC                     TEST RANGE OF LOWER HALF
1071  05634  074               CPI     -127            LOWER BOUND IS -127
      05635  201
1072  05636  006               LAI     -127            INITIALIZE RESULT TO LOWER BOUND
      05637  201
1073  05640  043               RTC                     RETURN IF INPUT IS ALGEBRAICALLY LESS THAN -127
1074  05641  302               LAC                     WITHIN ALLOWABLE RANGE--USE INPUT
1075  05642  007               RET
1076  05643  006       CDS1    LAI     '177            INITIALIZE RESULT
      05644  177
1077  05645  013               RFZ                     RETURN IF GREATER THAN 255 (RESULT = 255)
1100  05646  302               LAC                     GET LOWER HALF
1101  05647  271                CPB                    COMPARE WITH ZERO
1102  05650  006                LAI    '177
      05651  177
1103  05652  063               RTS                     RETURN WITH 127 IF GREATER
1104  05653  302               LAC                     IF LESS THAN, USE IT
1105  05654  007               RET
1106                    *
1107                    *       DOUBLE LENGTH SHIFT LEFT
1110                    *       ARGUMENT IS IN B AND C
1111                    *       SHIFT LENGTH IS AN A
1112                    *
1113  05655  330       DSLL    LDA                     USE D REG FOR INDEX
1114  05656  074               CPI     0
      05657  000
1115  05660  053               RTZ
1116  05661  250       DSL1    XRA                     CLEAR CARRY
1117  05662  302               LAC
1120  05663  022               RAL                     SHIFT THRU CARRY
1121  05664  320               LCA
1122  05665  301               LAB
```

```
1123  05666  022          RAL                SHIFT LEFT HALF WITH CARRY
1124  05667  310          LBA
1125  05670  031          DCD                DECREMENT INDEX AND TEST
1126  05671  110          JFZ    DSL1
      05672  261
      05673  013
1127  05674  007          RET
1130                      *
1131                      *     MAINTAINS CONTROL OUTPUT WITHIN SPECIFIED MAXIMUM AND
1132                      *     MINIMUM LIMITS
1133                      *     H AND L POINT TO 2 CONSECUTIVE LIMIT VALUES, MAX FIRST (DP FORMAT)
1134                      *
1135  05675  106    LIMIT CAL    DCPM         COMPARE WITH UPPER LIMIT
      05676  315
      05677  013
1136  05700  120          JFS    LMT2         GREATER THAN= USE MAX VALUE
      05701  311
      05702  013
1137  05703  060    LMT1  INL                 CHANGE ADDRESS TO MIN
1140  05704  060          INL
1141  05705  106          CAL    DCPM         COMPARE WITH LOWER LIMIT
      05706  315
      05707  013
1142  05710  023          RFS                RETURN IF WITHIN LIMITS
1143  05711  317    LMT2  LBM                USE LIMIT VALUE
1144  05712  060          INL
1145  05713  327          LCM
1146  05714  007          RET
1147                      *
1150                      *     DOUBLE PRECISION COMPARE
1151                      *     CONDITION FLAGS S AND Z CONTAIN RESULT
1152                      *                          S     Z
1153                      *        GREATER THAN      0     0
1154                      *        EQUAL             0     1
1155                      *        LESS THAN         1     0
1156                      *
1157  05715  301    DCPM  LAB
1160  05716  257          XRM                COMPARE SIGNS
1161  05717  160          JTS    DCP2         DIFFERENT=-TEST SIGN OF ARGUMENT
      05720  347
      05721  013
1162  05722  331          LDB                SAVE ARGUMENT
1163  05723  342          LEC
1164  05724  106          CAL    DSUB         CHECK DIFFERENCE
      05725  115
      05726  013
1165  05727  110          JFZ    DCP1         TEST IF EQUAL
      05730  344
      05731  013
1166  05732  302          LAC                GET LS HALF
1167  05733  074          CPI    0            TEST FOR EQUALITY
      05734  000
1170  05735  150          JTZ    DCP1         EQUAL=-EXIT
      05736  344
      05737  013
1171  05740  044          NDI    '177         SET SIGN = 0
      05741  177
1172  05742  064          ORI    1            SET Z = 0 (FOR GREATER THAN)
      05743  001
1173  05744  313    DCP1  LBD                RESTORE ARGUMENT
```

```
1174  05745  324          LCE
1175  05746  007          RET
1176  05747  301    DCP2  LAR
1177  05750  064          ORI   1
      05751  001
1200  05752  007          RET
1201     06000             ORG   '6000
1202  06000  001          D     1
1203  06001  002          D     2
1204  06002  002          D     2
1205  06003  003          D     3
1206  06004  004          D     4
1207  06005  005          D     5
1210  06006  006          D     6
1211  06007  006          D     6
1212  06010  007          D     7
1213  06011  010          D     8
1214  06012  011          D     9
1215  06013  011          D     9
1216  06014  012          D     10
1217  06015  013          D     11
1220  06016  014          D     12
1221  06017  015          D     13
1222  06020  015          D     13
1223  06021  016          D     14
1224  06022  017          D     15
1225  06023  020          D     16
1226  06024  021          D     17
1227  06025  021          D     17
1230  06026  022          D     18
1231  06027  023          D     19
1232  06030  024          D     20
1233  06031  025          D     21
1234  06032  025          D     21
1235  06033  026          D     22
1236  06034  027          D     23
1237  06035  030          D     24
1240  06036  030          D     24
1241  06037  031          D     25
1242  06040  032          D     26
1243  06041  033          D     27
1244  06042  034          D     28
1245  06043  034          D     28
1246  06044  035          D     29
1247  06045  036          D     30
1250  06046  037          D     31
1251  06047  040          D     32
1252  06050  040          D     32
1253  06051  041          D     33
1254  06052  042          D     34
1255  06053  043          D     35
1256  06054  044          D     36
1257  06055  044          D     36
1260  06056  045          D     37
1261  06057  046          D     38
1262  06060  047          D     39
1263  06061  047          D     39
1264  06062  050          D     40
1265  06063  051          D     41
1266  06064  052          D     42
```

RETURN WITH COMPARE RESULT UNDISTURBED
RESULT IS EITHER GREATER OR LESS THAN, BUT
EQUAL TO, THUS SIGN IS TESTED WITHOUT SETTING Z

```
1267  06065  053          D     43
1270  06066  053          D     43
1271  06067  054          D     44
1272  06070  055          D     45
1273  06071  056          D     46
1274  06072  057          D     47
1275  06073  057          D     47
1276  06074  060          D     48
1277  06075  061          D     49
1300  06076  062          D     50
1301  06077  063          D     51
1302  06100  063          D     51
1303  06101  064          D     52
1304  06102  065          D     53
1305  06103  066          D     54
1306  06104  066          D     54
1307  06105  067          D     55
1310  06106  070          D     56
1311  06107  071          D     57
1312  06110  072          D     58
1313  06111  072          D     58
1314  06112  073          D     59
1315  06113  074          D     60
1316  06114  075          D     61
1317  06115  076          D     62
1320  06116  076          D     62
1321  06117  077          D     63
1322  06120  100          D     64
1323  06121  101          D     65
1324  06122  102          D     66
1325  06123  102          D     66
1326  06124  103          D     67
1327  06125  104          D     68
1330  06126  105          D     69
1331  06127  105          D     69
1332  06130  106          D     70
1333  06131  107          D     71
1334  06132  110          D     72
1335  06133  111          D     73
1336  06134  111          D     73
1337  06135  112          D     74
1340  06136  113          D     75
1341  06137  114          D     76
1342  06140  115          D     77
1343  06141  115          D     77
1344  06142  116          D     78
1345  06143  117          D     79
1346  06144  120          D     80
1347  06145  121          D     81
1350  06146  121          D     81
1351  06147  122          D     82
1352  06150  123          D     83
1353  06151  124          D     84
1354  06152  124          D     84
1355  06153  125          D     85
1356  06154  126          D     86
1357  06155  127          D     87
1360  06156  130          D     88
1361  06157  130          D     88
1362  06160  131          D     89
```

1363	06161	132	D	90
1364	06162	133	D	91
1365	06163	134	D	92
1366	06164	134	D	92
1367	06165	135	D	93
1370	06166	136	D	94
1371	06167	137	D	95
1372	06170	140	D	96
1373	06171	140	D	96
1374	06172	141	D	97
1375	06173	142	D	98
1376	06174	143	D	99
1377	06175	143	D	99
1400	06176	144	D	100
1401	06177	145	D	101
1402	06200	146	D	102
1403	06201	147	D	103
1404	06202	147	D	103
1405	06203	150	D	104
1406	06204	151	D	105
1407	06205	152	D	106
1410	06206	153	D	107
1411	06207	153	D	107
1412	06210	154	D	108
1413	06211	155	D	109
1414	06212	156	D	110
1415	06213	157	D	111
1416	06214	157	D	111
1417	06215	160	D	112
1420	06216	161	D	113
1421	06217	162	D	114
1422	06220	162	D	114
1423	06221	163	D	115
1424	06222	164	D	116
1425	06223	165	D	117
1426	06224	166	D	118
1427	06225	166	D	118
1430	06226	167	D	119
1431	06227	170	D	120
1432	06230	171	D	121
1433	06231	172	D	122
1434	06232	172	D	122
1435	06233	173	D	123
1436	06234	174	D	124
1437	06235	175	D	125
1440	06236	176	D	126
1441	06237	176	D	126
1442	06240	177	D	127
1443	06241	200	D	128
1444	06242	201	D	129
1445	06243	201	D	129
1446	06244	202	D	130
1447	06245	203	D	131
1450	06246	204	D	132
1451	06247	205	D	133
1452	06250	205	D	133
1453	06251	206	D	134
1454	06252	207	D	135
1455	06253	210	D	136
1456	06254	211	D	137
1457	06255	211	D	137
1460	06256	212	D	138
1461	06257	213	D	139
1462	06260	214	D	140
1463	06261	215	D	141
1464	06262	215	D	141
1465	06263	216	D	142
1466	06264	217	D	143
1467	06265	220	D	144
1470	06266	220	D	144
1471	06267	221	D	145
1472	06270	222	D	146
1473	06271	223	D	147
1474	06272	224	D	148
1475	06273	224	D	148
1476	06274	225	D	149
1477	06275	226	D	150
1500	06276	227	D	151
1501	06277	230	D	152
1502	06300	230	D	152
1503	06301	231	D	153
1504	06302	232	D	154
1505	06303	233	D	155
1506	06304	234	D	156
1507	06305	234	D	156
1510	06306	235	D	157
1511	06307	236	D	158
1512	06310	236	D	158
1513	06311	236	D	158
1514	06312	236	D	158
1515	06313	236	D	158
1516	06314	236	D	158
1517	06315	236	D	158
1520	06316	236	D	158
1521	06317	236	D	158
1522	06320	236	D	158
1523	06321	236	D	158
1524	06322	236	D	158
1525	06323	236	D	158
1526	06324	236	D	158
1527	06325	236	D	158
1530	06326	236	D	158
1531	06327	236	D	158
1532	06330	236	D	158
1533	06331	236	D	158
1534	06332	236	D	158
1535	06333	236	D	158
1536	06334	236	D	158
1537	06335	236	D	158
1540	06336	236	D	158
1541	06337	236	D	158
1542	06340	236	D	158
1543	06341	236	D	158
1544	06342	236	D	158
1545	06343	236	D	158
1546	06344	236	D	158
1547	06345	236	D	158
1550	06346	236	D	158
1551	06347	236	D	158
1552	06350	236	D	158
1553	06351	236	D	158
1554	06352	236	D	158
1555	06353	236	D	158
1556	06354	236	D	158
1557	06355	236	D	158
1560	06356	236	D	158
1561	06357	236	D	158
1562	06360	236	D	158
1563	06361	236	D	158
1564	06362	236	D	158
1565	06363	236	D	158
1566	06364	236	D	158
1567	06365	236	D	158
1570	06366	236	D	158
1571	06367	236	D	158
1572	06370	236	D	158
1573	06371	236	D	158
1574	06372	236	D	158
1575	06373	236	D	158
1576	06374	236	D	158
1577	06375	236	D	158
1600	06376	236	D	158
1601	06377	236	D	158

APPENDIX D

APPENDIX D

CALIBRATION TABLE

Temperature (°)	Thermocouple, Cromel-Alumel (mv)	Amplifier Output (v)	Octal Reading
100	3.3	0.624	16 64
200	7.3	1.380	47 54
300	11.4	2.155	100 64
400	15.6	2.948	132 50
450	17.7	3.345	147 60
500	19.8	3.742	164 70
550	22.0	4.158	202 00
600	24.1	4.555	217 14
650	26.2	4.952	234 20
700	28.3	5.349	251 20
750	30.4	5.746	266 10
800	32.5	6.142	302 70
810	32.9	6.218	305 34
820	33.3	6.294	307 74
830	33.7	6.369	312 40
900	36.6	6.917	333 74

Special Distribution
UC-32

DISTRIBUTION

No. of Copies

Offsite

1 AEC Chicago Patent Group
- A. A. Churm

3 AEC Technical Information Center

16 Dow Chemical Corporation/Rocky Flats
- D. Livesay (12)
- P. H. Healy (2)
- D. Henry
- J. Skufka

13 Lawrence Livermore Laboratory
- D. F. Cruff
- R. Eckard
- R. A. Goodman
- G. McFarland (10)

Onsite

1 AEC/RL Patent Attorney
- R. M. Poteat

2 AEC/RL - RDT Sr. Site Representative, PNP
- F. R. Standerfer

29 Battelle-Pacific Northwest Laboratory
- K. M. Busness
- M. R. Compton
- V. L. Crow
- P. J. Dionne
- G. E. Driver
- M. D. Erickson (10)
- R. E. Falkoski
- J. C. Fox
- P. J. Hof
- R. R. King
- R. E. Mahan
- T. A. Seim (5)
- R. W. Stewart
- Technical Information Files (3)