# A NUMERICAL PROCESSOR BASED CONTROL FOR A SOLAR MONITORING AND TEST STATION

by

J. Randall Passella

Submitted in Partial Fulfillment of the Requirements
for the Degree of
Master of Science in Engineering
in the
Electrical Engineering
Program

C L K. Al L Jr. 6/8/78
Advisor Date

Jean Rich 7/21/78
Dean of the Graduate School Date

YOUNGSTOWN STATE UNIVERSITY

June 1977

# ABSTRACT

A NUMERICAL PROCESSOR BASED CONTROL FOR
A SOLAR MONITORING AND TEST STATION

J. Randall Passella
Master of Science in Engineering
Youngstown State University, 1977

With the increased interest in the area of solar energy, a definite need exists for a unit which would make the testing of systems easier and less time consuming. This paper presents the basic design of a unit, a numerical processor based control, which will work both monitoring and testing stations.

The recent advances in processor technology makes possible the ability to acquire, process, and analyze great amounts of data rapidly. This permits the design and implementation of data acquisition systems to be realized more economically than was previously possible. Although a minicomputer could easily be connected on-line to perform the same duties, the dedicated processor is a much more economical approach.

The numerical processing unit (npu) is readily available today and it can be used widely to free the engineer of many of the repetitive tasks sometimes associated with data acquisition. The engineer is then able to devote his time to more important aspects such as design and optimization.

The npu provides for an economical solution in control and data acquisition systems because it is easily adapted to many different situations. The main advantage of an npu based system is that the operation of the unit may easily be changed by altering the microprogram. A microprogram is a series of instructions which is stored in memory and controls the operations of the npu. Since many suitable memories are available in integrated circuit form, the operation of the unit may readily be changed by replacing the original memory with another which has different bit patterns stored in its registers.

The circuit presented here is used to control a solar monitoring and test station. The function of the npu is to gather all input data desired, make it available to the user, process it, and compute control signals. The output signals are made available in several forms to allow connection to different display devices. Two common examples are digital panel meters and teletypes. However, if an external digital to analog converter is added, analog devices such as strip chart recorders may also be used.

Since the npu is digital in nature, all input signals must be in digital form before they can be accepted by the processor. In solar experiments, as in many others, few of the variables of interest are in electrical form. To convert information from its natural form to that suitable to the processor, transducers are used. A transducer is simply an element capable of converting one form of energy into another.

Potentiometers are common examples, they convert a mechanical displacement into an electrical signal. This unit will accommodate any transducer whose output is a voltage within a specified range. This provides the user a certain amount of latitude when using this device.

This paper will show an economical yet effective controller, which should find wide application considering the upward trend in solar energy studies.

The paper begins with an introduction as to why this particular topic was chosen. It is shown that a need does exist for this type of device. Chapters II and III deal with the characteristics of the major digital logic families available. Chapter III expands on the information in Chapter II and discusses what design rules must be followed in using the chosen logic family.

Chapter IV goes into great detail in explaining the major subsystems of the unit. These major subsystems are the Input Unit, Memory Unit, Processing Unit and Timing and Control Unit.

Chapter V is a very brief introduction to actual system programming, and Chapter VI details what results were drawn from the study.

The Appendix provides a specific program example, and should answer most questions that may arise in that area.

# ACKNOWLEDGEMENTS

I wish to acknowledge the assistance and guidance of Dr. Charles K. Alexander. His patience and encouragement are greatly appreciated.

I also wish to thank Jodi Cutich and Colleen Deutinger for typing this paper.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER I

## INTRODUCTION

Since solar energy is just now being developed as a major energy source, there are several problems that must be considered if it is to be developed to its fullest. Among these are its intermittent nature and its high initial cost. Therefore if it is to compete with other energy sources it is essential that the designer optimize the system to the best of his ability. Knowing how much solar energy is available at different locations is one way of insuring the maximum return for the time and money spent.

Maps have been published which show daily radiation date,[1] but these are just averages and they must be used with caution. The data is collected at weather stations and used as the data for all of the surrounding area. This is fine for areas which show little change in sunlight received with change in location, but figures are not accurate for all locations. Factors such as large cities or natural obstructions such as mountains may cause the climate to vary greatly with distance.

Due to the above restrictions, an inexpensive method of gathering solar data would be an aid to scientists concerned with solar energy systems. Sophisticated data

---

[1]John A. Duffie and William A. Backman, Solar Energy Thermal Processes (New York: John Wiley & Sons, 1976), pp. 34-37.

loggers are available, but their cost is restrictive. The unit presented here can be constructed with a minimum amount of time and money. It not only monitors environmental conditions, but it is easily programmed to compute control signals based on those conditions. This unit works equally well with both static and dynamic variables. The monitored variable should have an output voltage range between zero and ten volts. Some typical transducers have outputs in the microvolt region. The designer should correct for this by using a simple analog amplifier between the output device being monitored and the voltage to frequency converter. The voltage to frequency converter is explained later.

The basic unit handles sixteen input signals, and is capable of controlling sixteen external devices. The frequency with which the control signals depends on the length of the control program, but the programs generally last no more than two minutes. In a typical home situation, the program need be run no more often than twice an hour. But testing via simulation may be greatly sped up by running the program as often as possible.

# CHAPTER II

## CHOICE OF LOGIC FAMILY

There are many families of digital logic available today and naturally, some are better suited to some applications. By far the most common logic devices today are constructed of transistor-transistor logic (TTL) gates. Most circuit types are supplied by several manufacturers and the price has been declining almost since their introduction. A standard TTL gate has a typical propagation delay of ten nanoseconds and a power dissipation of ten milliwatts.

An increase in speed was realized when the semiconductor manufacturers added Schottky diodes to the standard TTL parts. These additional diodes keep the transistors out of saturation and typical Schottky gate has a delay of only three nanoseconds. However, its power dissipation, of nineteen milliwatts, is almost double that of the standard gate. Since this unit should be battery operated for maximum flexibility, power consumption is an important factor.

By reducing the inherent capacitance of the integrated circuit, a low power Schottky device may be fabricated. This results in higher circuit densities and the reduction of power consumption to only two milliwatts per gate.

TTL is the most common and least expensive digital logic to use. However, if the unit is kept small enough,

expense is not a major consideration. One alternative that is only slighly more costly is a family manufactured of Complementary Symmetry-Metal-Oxide-Semiconductors (CMOS). The CMOS family has several characteristics that make it a good choice for this unit. The characteristic of prime importance is its ultra-low quiescent power requirements. With a five volt supply the power requirements range from .005 microwatts to .05 microwatts for small scale integration (SSI), to .1 to .5 microwatts for medium scale integration (MSI).

Another advantage of CMOS is its very high noise immunity. CMOS has a typical noise immunity of 45% of the supply voltage, and a 30% noise immunity is guaranteed for most devices. Standard TTL devices have a guaranteed noise immunity of approximately 20%. This is an important consideration if the controller is to operate reliable in close proximity to electrical machinery.

CMOS logic will operate in a wide temperature range. Even the least costly, that is, those supplied in plastic dual in-line (DIP) packages, have an operating range of from -40°C to +85°C. Although this is adequate for most applications, a high performance version is available with an operating range of from -55°C to +125°C.

Another characteristic which is helpful - but not considered a deciding factor - is the high fan-out capability of CMOS. Fan-out is the ability of one logic gate to drive

another. One CMOS gate is capable of driving fifty other CMOS gates. A typical TTL gate will only drive ten other gates like itself.

Typical CMOS gates are almost an order of magnitude slower than comparable TTL gates, but the CMOS logic is capable of operating at typically 5.0 megahertz. This is much faster than required for this solar unit controller.

# CHAPTER III

## MAJOR CONSIDERATIONS IN THE USE OF CMOS LOGIC

As in all circuit design, there are rules which must be followed to insure correct system operation. The first in logic design is the operating range of the supply voltage. The "B" series devices used in this unit are guaranteed to operate with supply levels between 3 and 18 volts D.C.

Power dissipation must be carefully considered since battery life is totally dependent on this aspect. The quiescent dissipation was mentioned previously and it is due to a combination of leakage effects. This is very important in this application, since under normal conditions, the unit is idle for a much greater time than it is active. The other form of power dissipation is known as dynamic dissipation. It consists of two parts; "through" current which flows when both the N-MOS and P-MOS devices are conducting during switching, and the second part is the supply current needed to charge the output capacitance during switching. The dynamic dissipation of most CMOS circuits is given by

$$P_d = C_o V_{DD}^2 f \quad \text{(watts)}$$

where $C_o$ is the effective output capacitance in farads, $V_{DD}$ is the supply voltage in volts, and f is the frequency in Hertz.

A curve showing dynamic power dissipation as a function of frequency is included with most CMOS data sheets. An example is shown in Figure 3.1.

Although CMOS devices are less sensitive to noise than bipolar logic families, precautions must still be taken. Discrete decoupling capacitors are recommended across the power bus to insure a low dynamic impedance.

Most of the devices in this unit are clocked circuits. That is, they operate synchronously with an input referred to as the clock. Clock rise and fall times are limited to less than 15 microseconds. Longer rise and fall times may cause errors due to false triggering and data ripple through. False triggering may occur if noise is picked up on a clock line during a slow edge. The noise may be of a frequency that during a single edge it may cross the switching threshold several times, thereby clocking in unwanted data. The other hazard mentioned, data ripple through, occurs mainly because all circuits do not switch at the exact same voltage level. The switching level for a CMOS circuit may lie anywhere in the range of 0.3 to 0.7 $V_{DD}$, where $V_{DD}$ is the potential of the positive supply. Ripple through may easily be explained with the aid of Figure 3.2. The circuit illustrated could be used to shift data bits through the flip flops by one position for each clock cycle. If all of the circuits switched on the same value of clock voltage, there would be no problem. However, assume that flip flop 1 has a switching value of $0.7V_{DD}$.

Typical CMOS Power Dissipation

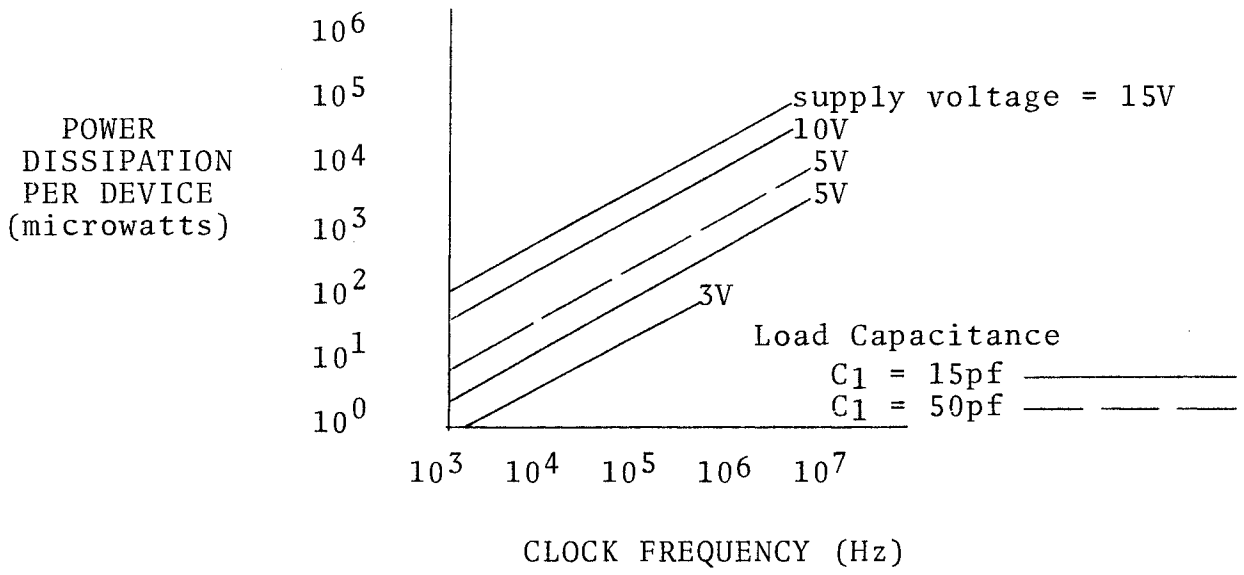



Figure 3.1--Power dissipation versus clock frequency for a CD4042 CMOS Quad Clocked "D" Latch.

If flip flop 2 had the same switching value, the state of Q1 just prior to switching would be shifted to Q2 when the clock voltage reached $0.3V_{DD}$. This would be the action desired and a device known as the shift register would have been implemented. However, if flip flop 2 does not switch until the clock reached $0.7V_{DD}$, the input will shift to Q1 and then from Q1 to Q2 during a single rising clock edge. It can be seen that the data has rippled through two stages of logic when only a single shift was desired.

Chapter IV presents a detailed description of the Controller. The Controller is composed of several subsystem parts and each is covered in detail. Actual programming examples are relegated to the Appendix.

Errors Due To Slow Clock Transition

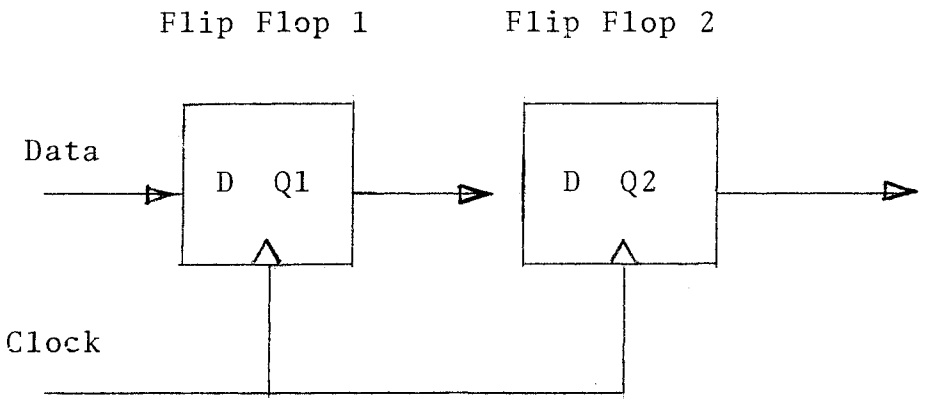



Figure 3.2--Errors caused by slow clock transistions.

# CHAPTER IV

# SYSTEM DESCRIPTION

## Introduction

A block diagram of the complete unit is shown in Figure 4.1. The systems six main sections are: The program storage registers, the input signals conditioning unit, the processor itself, the memory, the output section, and the timing and control section. Briefly, the system operates as follows: an input is entered into the input registers and then the contents of the input registers are held in the program storage unit. Pertinent information is stored in the memory to used in later calculations. Control signals are computed by the processor and sent to the output unit where they are latched into registers and finally, the timing and control unit makes sure everything is done in an orderly fashion. Each of the six sections is described separately and in detail below.

## Input Unit

A detailed diagram of the input unit is shown in Figure 4.2. The input signals are generated by remote sensors and are generally analog in nature. Usually, the variables being monitored are not voltages and currents, but are physical quantities such as temperature, solar radiation, and wind speed. These physical quantities must be converted

to electrical signals by transducers, as mentioned previously. The windmill is a form of transducer, and may be used to convert wind speed into electrical energy. Pyranometers and pyroheliometers are less well known instruments. A pyroheliometer is used to measure beam radiation (direct solar radiation) while a pyranometer can be used for measuring either total radiation or just diffuse radiation.

Even after the inputs have been converted to electrical quantities, they are still not in a form useful to the processor. The analog voltages must be converted to digital representations. There are many ways in which this may be accomplished and the one chosen here is known as voltage to frequency conversion. This type of conversion has advantages in that it requires very little hardware but still has the degree of accuracy necessary in the application.

Figure 4.3 shows a circuit capable of performing the required conversion. The first stage is an operational amplifier connected for differential amplification. With this type of connection the output voltage $V_o$ is the sum of two voltages $V_{o1}$ and $V_{o2}$. Neglecting the input $V_{i2}$, the circuit is simply the common non-inverting connection with output

$$V_{o1} = V_{i1} \left(\frac{R1+R2}{R1}\right)\left(\frac{R4}{R3+R4}\right)$$

If $V_{i2}$ is observed with $V_{i1}$ set to zero, the common inverting connection results with the corresponding output.

General Layout Of Controller

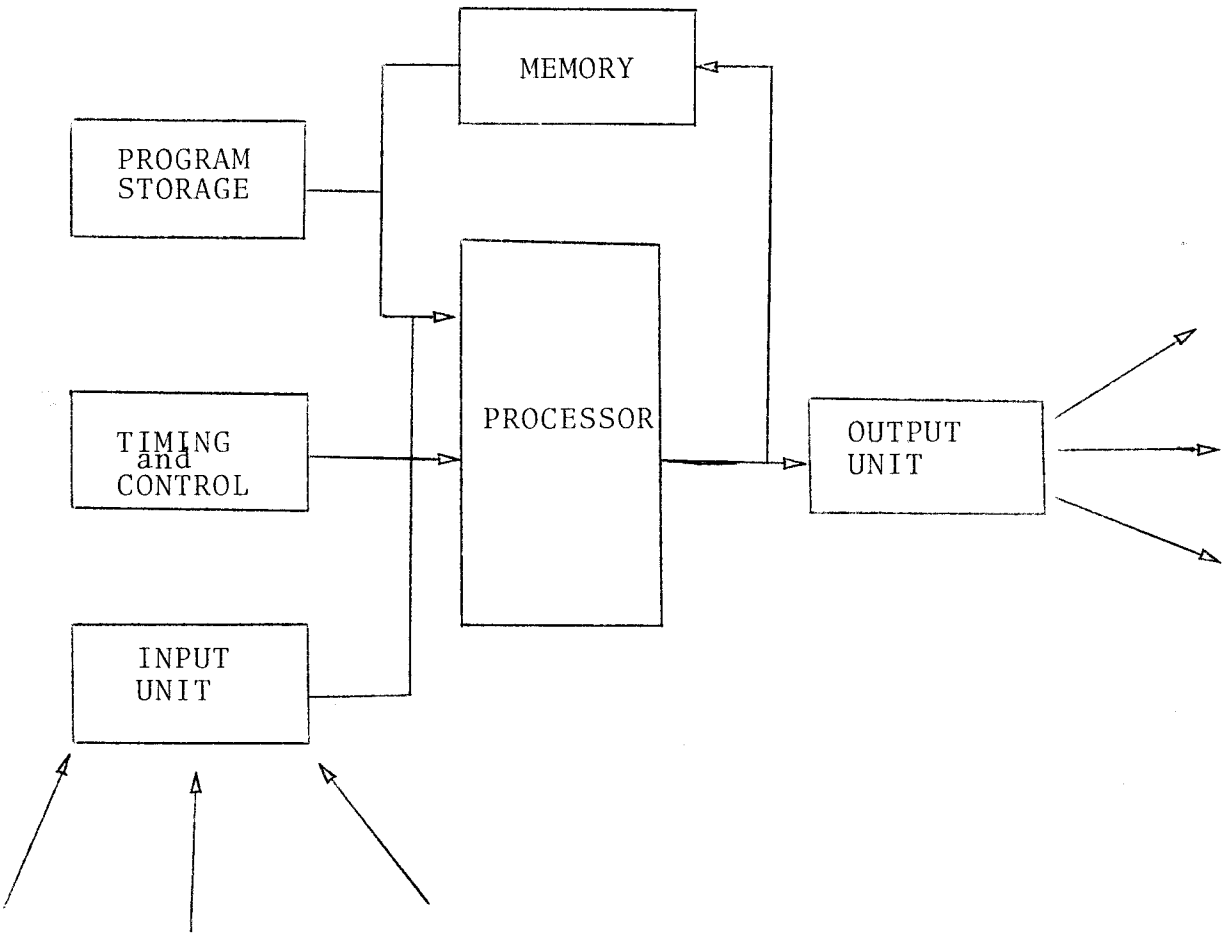



Figure 4.1--General Layout of Controller

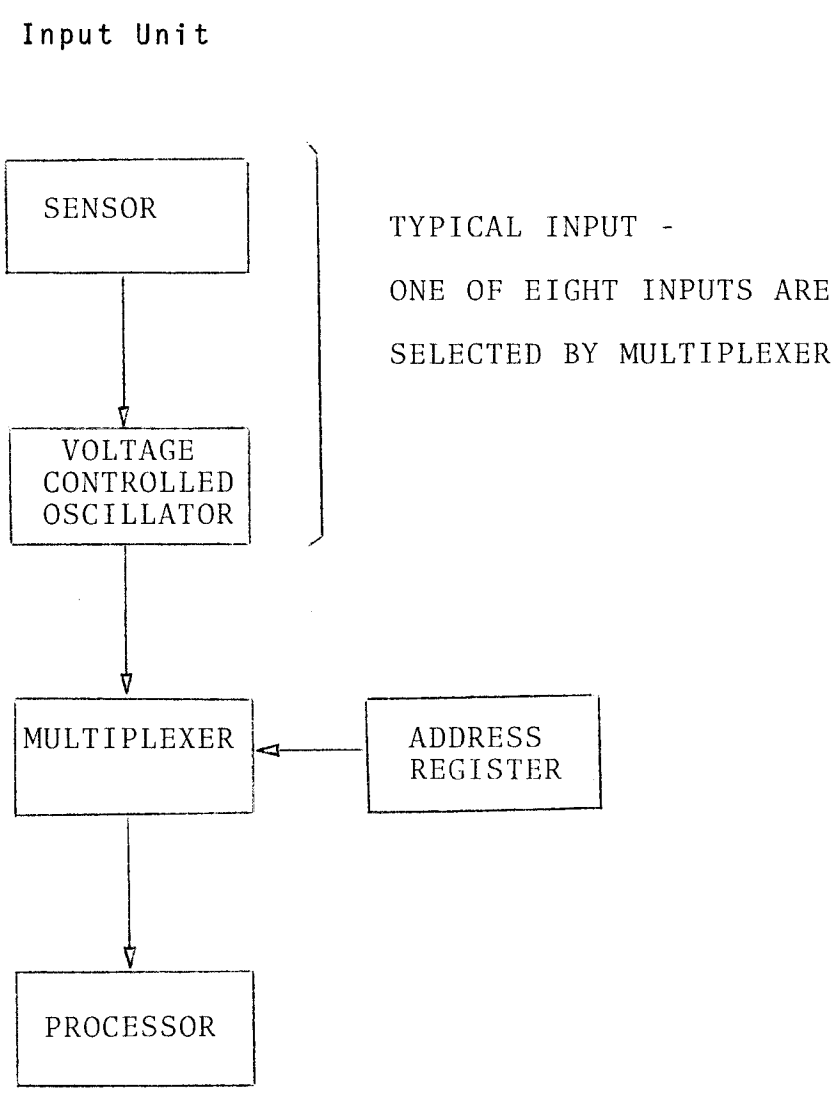



Figure 4.2--Input unit used to convert solar data into digital representations.

$$V_{o2} = V_{i2} \quad \frac{R2}{R1}$$

By superposition the two results are summed, that is

$$V_o = V_{o1} + V_{o2}$$

Now, when the field effect transistor is off, R4 approximates an infinite impedance and therefore

$$V_o = \frac{R1+R2}{R1} \quad V_{i1} \; - \frac{R2}{R1} V_{i2}$$

and

$$V_{i1} = V_{i2} = V$$

$$V_o = V \left(1 + \frac{R2}{R1} - \frac{R2}{R1}\right)$$

$$V_o = V$$

Now, if the FET is on, R4 is approximately zero and

If $R2 = R1$

then $V_o = V_i$

The above calculations show that in this application

$$V_o = \pm V_i$$

Referring to Figure 4.3, when $V_{o1}$ is negative, $V_2$ increases until it is greater than the breakdown voltage of the zener diode, $V_z$. Amplifier $A_3$ is saturated positively until $V_2$ exceeds $V_z$, and therefore the FET remains on. When $V_2$ exceeds $V_z$, $A_3$ saturates with a negative output and the FET turns off. The output from $A_1$ becomes negative and $V_2$ starts to increase until it exceeds $V_z$. Therefore $V_3$ is a square wave with a frequency f given by

$$f = \frac{Vin}{4RCV_z}$$

Linear Voltage Controlled Oscillator

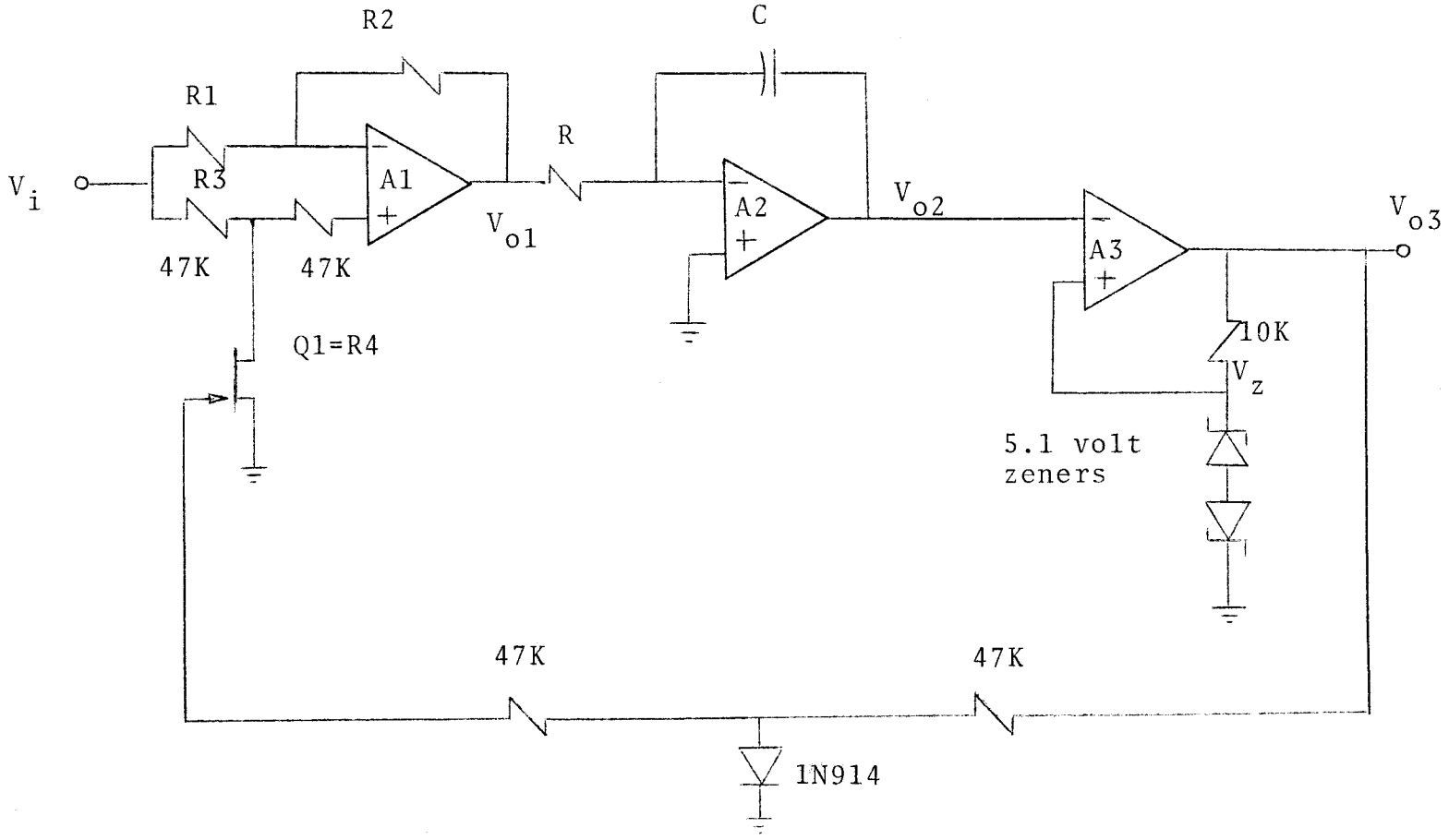



Figure 4.3--Linear Voltage Controlled Oscillator.

Since the frequency is dependent upon the input voltage, the input voltage may be calculated by determining the frequency of the oscillator. This is easily done by counting the number of cycles in a specified amount of time.

This oscillator produces an output frequency which is a linear function of the applied input voltage. This makes the determination of the input voltage rather simple, but an economical alternative is shown in Figure 4.4. The output frequency as a function of the input voltage is given in tabular form in Table 1. Since this is a nonlinear characteristic, more calculations are required to compute the input voltage. However, this oscillator consists of only one inexpensive CMOS circuit, one resistor, and one capacitor.

It is assumed that the oscillators are located at a remote location with respect to the processing unit and that data is transmitted in digital form. Digital transmission is preferred due to its inherent noise immunity.

Although digital transmission is more tolerant of noise than is analog transmission, noise factors must still be taken into consideration. CMOS devices are low current circuits and therefore some precaution must be taken if they are to transmit data over long distances.

There are several possible ways of sending the data, but not all of these are reliable if the distance ranges between ten and one hundred feet. The most common type of transmission line is the one composed of just a single wire.

Nonlinear Voltage Controlled Oscillator

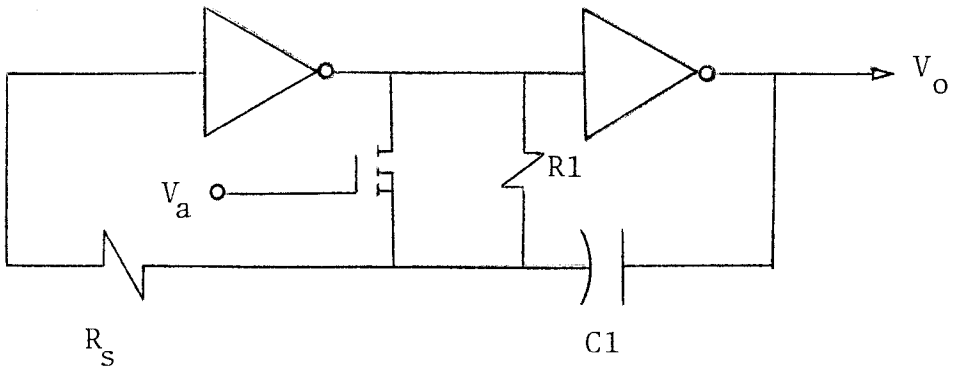



Figure 4.4--Voltage Controlled Oscillator using only two external components.

PERIOD -(microseconds)

| $V_A$ | $V_{DD}$=5V | $V_{DD}$=10V | $V_{DD}$=15V |
|---|---|---|---|
| 0 | 120 | 54 | 48 |
| 5 | 115 | 45 | 41 |
| 10 | --- | 32 | 30 |
| 15 | --- | -- | 24 |

Typical Values:

R1 = 10k Ohms  C1 = .001 -.004 microfarads
$R_S$ = 100k Ohms  $0 \leq V_A \leq V_{DD}$

Table 4.1--Typical characteristics of the CMOS oscillator of Figure 4.4.

An example of this is the direct connection of the output of one logic element to the input of another. This type can only be used where the interconnect distances are short and the system is relatively noise-free. Generally this connection is only used for digital circuits which are located on the same board.

An extension of the above scheme is to add an additional ground wire. The little advantage gained is generally not worth the extra effort. However, this connection does provide reliable communication between boards which are separated by no more than several feet. Loosely twisting the wires will improve the noise rejection but the only significant advantage is realized when the wire has a specific number of turns per inch. This is known as tight twisted pair and it has a definite impedance characteristic. This allows for longer transmission lines and proper termination of the line.

The twisted type of cable is available in many different configurations. The transmission characteristics are affected by the type of insulation used, the number of twists per inch, and the wire size.

One of the best ways of connecting logic elements separated by long distances is to provide coaxial transmission lines. There are several types which are suitable and they are in the 50 to 200 ohm impedance category. They provide a low loss and well shielded communication channel, but they are costly.

Which ever type of transmission line is chosen, the correct logic elements must be matched to it. A typical element does not have the current output to drive a line at the rate usually required for accurate communication. Therefore, special circuits, line drivers, are used to provide the required output current. Assuming a 3.5 volt minimum input voltage for a logic "1", the line driver must be capable of delivering 35 milliamps if a 100 ohm cable is selected. There are several line drivers in integrated circuit form which are capable of meeting these requirements.

Since in any but the most trivial applications more than one variable must be monitored, there must be a method of selecting each of the input signals, one at a time. The circuit selected to accomplish this function is the CD4051 Single Eight Channel Multiplexer. This is a digitally controlled switch, fabricated in CMOS technology. It may be used for either analog or digital applications. The circuit consists of eight switches and only one is turned on at a time. When a switch is on, its input is connected to the common output terminals. A switch is selected by applying a binary number to the three control inputs A, B and C.

These switches provide a low 'on' resistance of approximately 80 ohms and a very low 'off' leakage current of typically +10 picoamperes, at $V_{DD}$-$V_{EE}$ = 10 volts. The CD4051 also permits logic level conversion on chip. The quiescent power dissipation of this circuit is typically 1 microwatt.

The system is not limited to nomitoring only eight inputs. The CD4051 has provisions for easy expansion made possible by the 'Inhibit' input. A high level on this line will turn all of the switches off. Adding one more signal and an inverter, as shown in Figure 4.5, will allow the selection of one of sixteen channels. The most significant bit is used to select one of the multiplexers and the other three lines select the specific channel of that multiplexer.

The final portion of the input section to be considered is the four stage counting network. The units which provide the function are two CD4518 Dual Binary Coded Decimal Up Counters. Each circuit has two identical internally synchronous four-stage counters. The designer has the option of having the counters advance on either the positive edge or the negative edge of the clock through the use of the 'enable' input.

Binary Coded Decimal is a method of representing the ten decimal digits, 0 through 9, with a four bit binary word. These particular counters were chosen because the processor chosen requires BCD inputs.

In cascading these counters together, the most significant bit, Q4, of one counter is connected to the enable input of the next counter. The clock inputs are held low so that the state of the counters is advanced on a falling clock edge. In this manner, any desired number of counters may be cascaded. In the system shown, four stages of counting are used, providing any count from 0000 to 9999. The counters are cleared by a single high level on the 'Reset' line.

One Of Sixteen Multiplexer

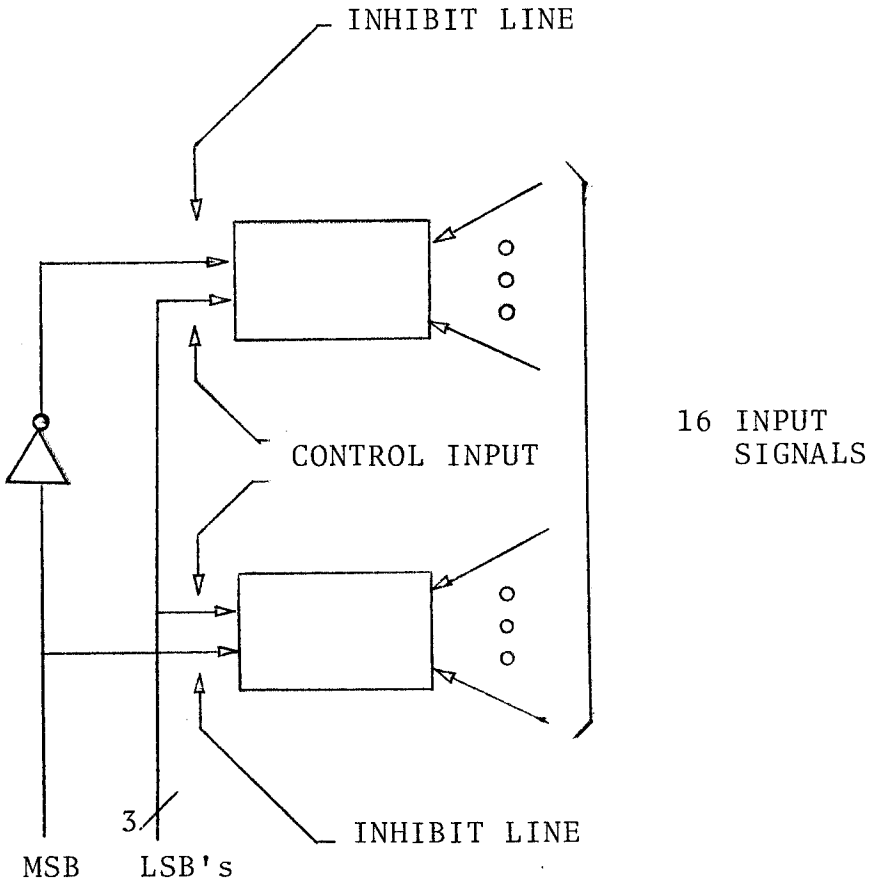



Figure 4.5 A one-of-sixteen multiplexer constructed using two one-of-eight multiplexers.

## The Memory Unit

The memory unit used in this controller is capable of storing 256 BCD digits and is considered adequate for most applications. However, it may easily be modified to handle a greater amount of data. The basic layout of the memory is shown in figure 4.6. Although the memory is actually only a single block, it may be thought of as consisting of two separate parts, a Random Access section and a Stack.

The memory is an array of bistable elements and is arranged as 256 words by 4 bits. Data is written into the memory by first applying a binary word to the address inputs and another binary word to the data lines. The 'Memory Enable' and 'Write Enable' inputs are then momentarily taken low. Information may be read from the memory by applying an address word and holding the 'Memory Enable' low and the 'Write Enable' high. Suitable memories are available in CMOS technology.

In this unit the inputs to the memory are outputs from the processor. If a particular number will be of use in more than one calculation, it can be stored in memory. This saves programming, as the same calculations do not have to be done repeatedly on the same data. As was mentioned earlier in this section, the memory can be considered of consisting of two distinct parts, this is explained below.

Memory Unit

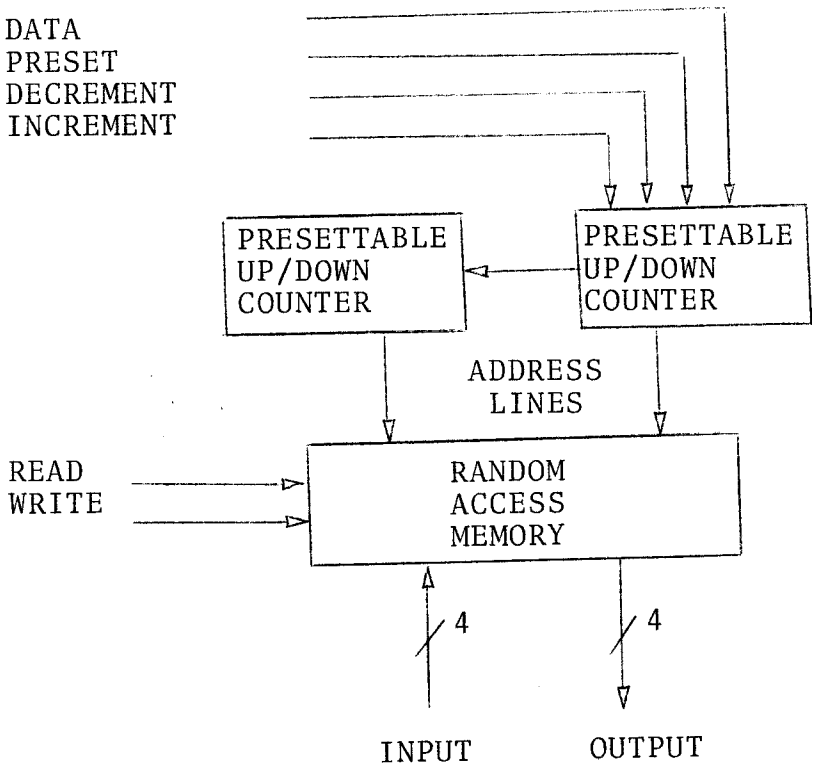



FIGURE 4.6 Basic layout of memory unit. The address registers may be incremented, decremented, or set directly to any value.

The type of memory that is the most familiar is the Random Access Memory (RAM). With this memory any stored data may be retrieved by applying the correct address to the appropriate inputs. That is, data may be read from and written into the memory in any order that is desired by the user.

The second portion of the memory is similar in function to a shift register and is called a Stack. Its current address is held in an appropriately named location, the Stack Pointer. The Stack is a type of sequential memory in which data in consecutive locations is the most accessible. The Stack Pointer can be easily incremented and decremented. This feature requires less program storage and less execution time than addressing the RAM.

In higher power processing systems the RAM address and the Stack Pointer are held in different registers. In this unit they are loaded into the same register and therefore some of the flexibility is lost.

The CD4029 Presettable Up/Down Counter was chosen as address register and Stack Pointer. This circuit is very versatile as the number in the counter may be incremented, decremented, or set to any desired number. The increment and decrement functions are used in stack operations, and the preset feature is used for the RAM address.

## The Processing Unit

The processing unit is the part of the controller that requires the most thought. Although each section is necessary for proper operation of the unit, the processor can be considered the most important. There are many microprocessors on the market and many are considered to be more powerful than the Texas Instruments TMS 0117. The microprocessors are usually capable of completing operations in microseconds, where it takes the TMS 0117 milliseconds to accomplish comparable functions. Also, the typical microprocessor recognizes between fifty and one hundred instructions. This is to be compared to the chosen processor's twenty three. Finally, most mircoprocessors on the market can address 64K bytes of memory, without external hardware.

The TMS 0117 receives competition from the many other calculator circuits on the market. These are generally very powerful and still inexpensive. A typical calculator chip may have ten on chip memory locations and also be able to perform complex arithmetic operations such as computing trigonometric functions and be accurate to ten decimal digits.

The TMS 0117 cannot alone address memory, nor can it compute anything more complex than a product or quotient, however, it has advantages over each of the alternatives

mentioned. It is better suited to this application than a microprocessor because it will perform division and multiplication with a single command. The microprocessor must accomplish these functions through lengthy add and shift routines. Further, typical microprocessors are eight bit machines and therefore are limited to two BCD digits at a time. The chosen processor can handle up to ten digits at a time. These features of the TMS 0117 makes it simpler to program, and the need for a knowledgeable software person is eliminated.

The calculator circuits, on the other hand, have the required accuracy, but they lack flexibility. Their outputs are designed to drive seven segment displays, and these present a problem in that they must be decoded to be used effectively in arithmetic calculations.

## Processor Description

The chosen processor is well suited to this type of application because it provides several control signals and because it processes data in BCD format. It is constructed using Metal-Oxide-Semiconductor/Large Scale Integration (MOS/LSI) technology. Its features include ten digit accuracy and three internal registers. Its instruction set includes addition, subtraction, multiplication and division. It can also shift data in either direction, and its maximum operation time for any instruction is 100 milliseconds.

The processor operates in the following manner. A binary word is applied to the data inputs and the processor is enabled. The binary data is then entered into the processor and decoded internally to execute the desired instruction. The output is then made available in word parallel, decade serial form. That is, the output is only available one digit at a time in a multiplexed fashion. Timing signals are provided to latch the output digits. Although the output is only available one digit at a time, it can be used to illuminate a ten digit seven segment display. The digits are repeated so quickly that the display appears flicker free.

The TMS 0117 is an asynchronous device, that is, different computations require varying amounts of time. Some typical, and also worst case examples are shown in table 4.2. This data shows that the instruction requiring the least amount of time is the shift operation, which takes 1.72 milliseconds. The most time consuming operations are multiplication and division. Under worst case conditions, division may take as long as 80 milliseconds. These figures are based on a nominal clock frequency of 250 kHz. The clock is an oscillator, external to the device, and must be supplied by the user.

The clock is only a single phase device and must oscillate at any frequency between 100 and 400 kHz. The minimum frequency of 100 kHz arises because the device is a dynamic circuit. This means that data is not held in static

| FUNCTION | TIME |
|---|---|
| Number Entry, Single Digit | 5.2 ms maximum |
| Operation Instruction Entry | 6.9 ms maximum |
| Shift Left or Right | 1.72 ms maximum |
| Increment or Decrement | 3.4 ms |
| Exchange Operands | 5.2 ms |
| Add, Subtract | 8.6 ms |
| Multiplication | 70 ms maximum |
| Division | 80 ms maximum |

TABLE 2. Cycle execution times of TMS 0117 PROCESSOR. Times are based on a 250 kHz clock.

registers and it therefore must be continually refreshed. The maximum clock frequency is the highest frequency that the circuit will operate continually at, without overheating.

As stated earlier, different instructions take different amounts of time. One method of assuring that all of the instructions had a sufficient amount of time for completion is to allow enough time to accomplish the most time consuming operation. In this case, allowing 100 milliseconds for each instruction would be sufficient since the maximum amount of time required for any instruction is less than that, namely 80 milliseconds.

The Texas Instruments circuit chosen provides the means for a better solution. A Busy/Ready signal has been included on the chip which signals when the processor has completed its previous instruction. This allows the entire system to run at a much higher speed and consequently the throughput is increased significantly.

One characteristics of this processor that could be considered to be a drawback is that it does not incorporate a floating decimal point. There is a signal on the integrated circuit itself, KN, which may be used to imply a decimal point. It relies on timing signals since the user may imply his own decimal point location by entering leading zeros into the processor.

The functions that the processor performs are of three types. The commands mentioned previously such as addition, subtraction, multiplication, and division, are

referred to as arithmetic operations. Operations such as shift left are labled register operations since these functions simulate a shift register. The third type is known as internal control. They are mentioned last, but they are definitely not least in importance.

The internal controls are the programs etched into the processor which actually instructs the circuit what to do when, for instance, the 'Increment' instruction is encountered. It is because the internal "microprograms" are of varying length that the different instructions are executed at different speeds. Some instructions require fewer microprogram steps and are therefore executed more quickly.

## The Timing And Control Unit

The timing and control unit shown in figure 4.7 is the section that keeps order in the whole system. Its main function is to decode the instructions, one at a time, and issue a signal to initiate the proper program counter is simply a binary counter that is used to point to the next sequential program instruction. The instruction register is the memory into which the section has the responsibility of providing the timing signals so that each instruction is executed properly.

The most challenging aspect of this project way to co-ordinate the hardware and the software. That is, defining an instruction set that was sufficiently powerful enough, but

yet did not require an extensive amount of hardware for implementation. The instruction set decided upon consists of thirteen unique instructions which may be broken down into four basic groups.

The first four are denoted as "Load Immediate" commands to imply that data is transferred immediately from the instruction register to some other register. The first of these is the Load Processor Immediate (LPI) instruction and is used for entering commands and data into the processor much the same as in working with a pocket calculator. When the LPI instruction is decoded, as when any other instruction is decoded, a well defined set of timing signals is generated. The first of these causes the program counter to be incremented by one, and in this way the data to be transferred is fetched from memory. The second and final signal of this instruction cause the data to be entered into the processor. These signals are diagrammed in figure 4.8.

When the instruction word is fetched from meory, it is latched into a temporary holding register and decoded by a 74C154 Four Line to One of Sixteen decoder. The line that is thus selected goes to a low state while the other fifteen remain at a high state. This signal is used as the input to several CD4016 Quad Bilateral Switches. The switches are sequentially turned on by the CD4013 Decade Counter which has ten decoded outputs. The ten outputs go to a high state one at a time, in sequential order. The outputs of switches

Timing And Control Unit

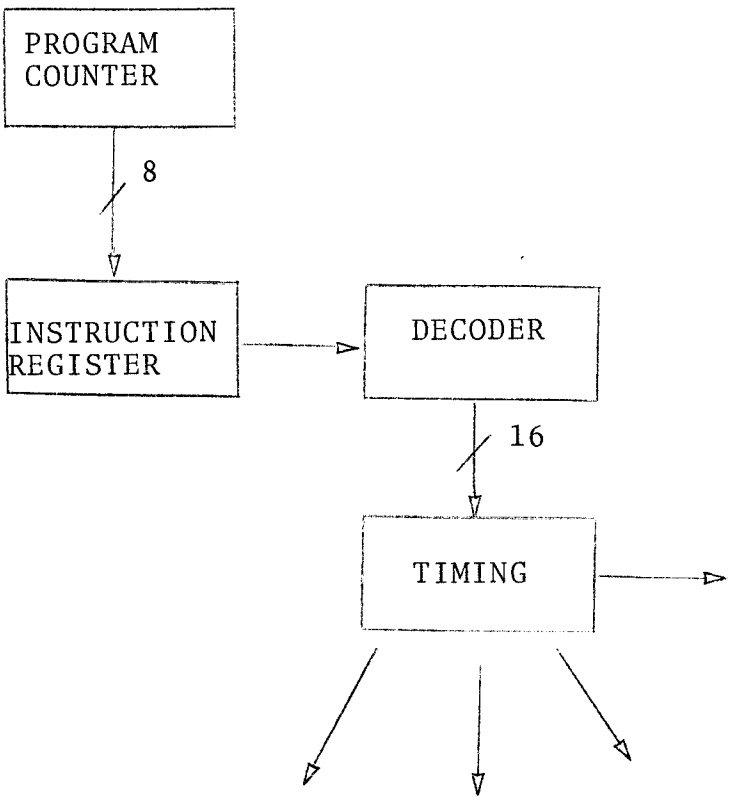



Figure 4.7 The timing and control unit provides all of the timing signals necessary to execute the various instructions.

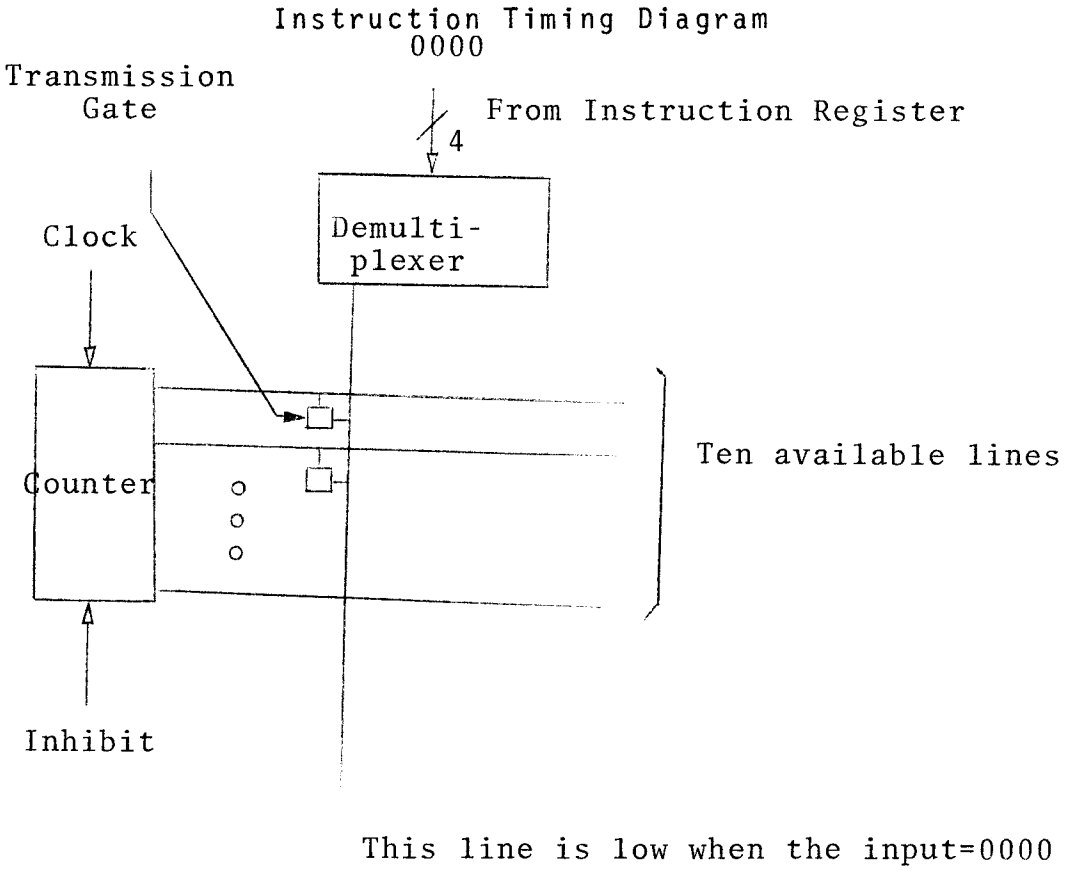



Figure 4.8 This diagram illustrates the control signals for the Load Processor Immediate instruction. The code for LPI is 0000 is latched into a register and causes a particular line to be selected. When its associated gates are turned on by the counter, the needed transfers are accomplished.

selected by the One of Sixteen decoder go to a low state when they are turned on by the Decade Counter. These outputs are tied to the appropriate inputs to cause the desired response. All of these switches are in a high impedance state when they are off and therefore may be tied in the wired or configuration shown. That is, more than one combination of signals may generate the same response.

The next two words of this series cause two data words to be transferred from the instruction register to other parts of the system. The Load B, C Immediate command causes the next word after the instruction to be transferred to the C register. These two registers are used to hold temporary data that is to be transferred to some output medium such as a teletype or Cathode Ray Tube terminal. The final instruction of this set is the instruction is loaded into the lower four bit registers, and the second word after the instruction is loaded into the higher four bits. The Stack Pointer is used to address an independent memory into which temporary results from the processor may be stored. There are other commands that affect the Stack Pointer and these are explained later.

The only other command in this set is used to transfer data from the instruction register to the memory and is termed to the Load Memory Immediate instruction.

The next group of instructions put the data that is currently addressed in memory onto the data bus. The first

of these is the Load Processor Direct and is used to transfer data from the memory to the processor. The only other instruction in this categroy is the Load Output Direct command and is used to transfer data from the memory to the output registers.

The next set is a group of three instructions of which only two put any data onto the bus. The first is the Store Processor Direct and is used to transfer data from the Processor to the memory. The data is written into the memory location currently addressed by the Stack Pointer. The Store Processor command is the main instruction that gives the unit its controller capability. When this instruction is encountered the next word determines the destination register for the transfer. Since four bit data words are used, one of sixteen output registers may be selected. The third instruction of this series is the Halt code and is used in a program. It signifies the end of a program, and is used once in a program. It signifies the end of a program, and is used to power down the system. The system is powered up when the delay timer times out. This delay is set by the user.

There are only four other instructions available to the user. Two of these were alluded to above, that is, those that affect the Stack Pointers. One of these increments the Stack Pointer, and one decrements it. These are valuable aids to the programmer in that he can easily store and retrieve data without keeping track of the absolute address locations.

Another of the instructions in this set is used to signal the unit that data previously loaded into the B and C registers in to be transferred to a Universal Asynchronous Receiver/Transmitter (UART). The UART is an LSI device, and as its name implies, it is used for asynchrounous data transmission. It is capable of adding start bits, stop bits, and also parity bits under programmer control. It requires a clock signal of sixteen times the desired data transmission rate.

The final command is the Input command and this gives the unit the ability to sense changes in its surroundings. When this command is decoded, the Instruction register is incremented and the next word tells the unit which location it is to check for input data. Again, one of sixteen locations is chosen. This instruction causes the Increment instruction to be jammed into the processor and it is held there for a specific amount of time. After this specified time has elapsed, the next instruction is fetched from memory. Prior to executing the Input command is recognized, it gates the specific input line to the Enable pin of the processor. The input must be a square wave raging anywhere between D.C. and 200 hertz maximum. The maximum frequency is specified due to the time it takes the processor to perform this instruction. It is suggested that some form of voltage controlled oscillator be used to provide the function. In that way, a tranducer monitoring a variable of interest

can be chosen to provide a voltage output that is proportional to the input. This output voltage is then used to control a voltage controlled oscillator, and by counting the number of cycles in a given time period, the variable of interest can be computed. This data can then be used in preprogrammed steps to provide the needed control signals.

## System Operation

Now that the five sections have been explained in detail, the entire unit may be understood as a whole. The block diagram of figure 4.9 will aide the reader in following this explanation. The timer, an astable oscillator, is programmed by the user to provide the required delay between successive program runs. This may vary between several seconds and several hours. An ideal circuit for this is the 555 timer provided by several manufacturers. It is easily set up to provide delays from microseconds to hours by the selection of only two external resistors and one capacitor.

One initialization of the system the oscillator is reset and begins its timing cycle. When it completes its cycle its output goes low and this initiates a new program cycle for the controller. As was mentioned, the final statement in a program is the Halt instruction, and this pulses the monostable once more. The program is run continuously until the unit is shut off.

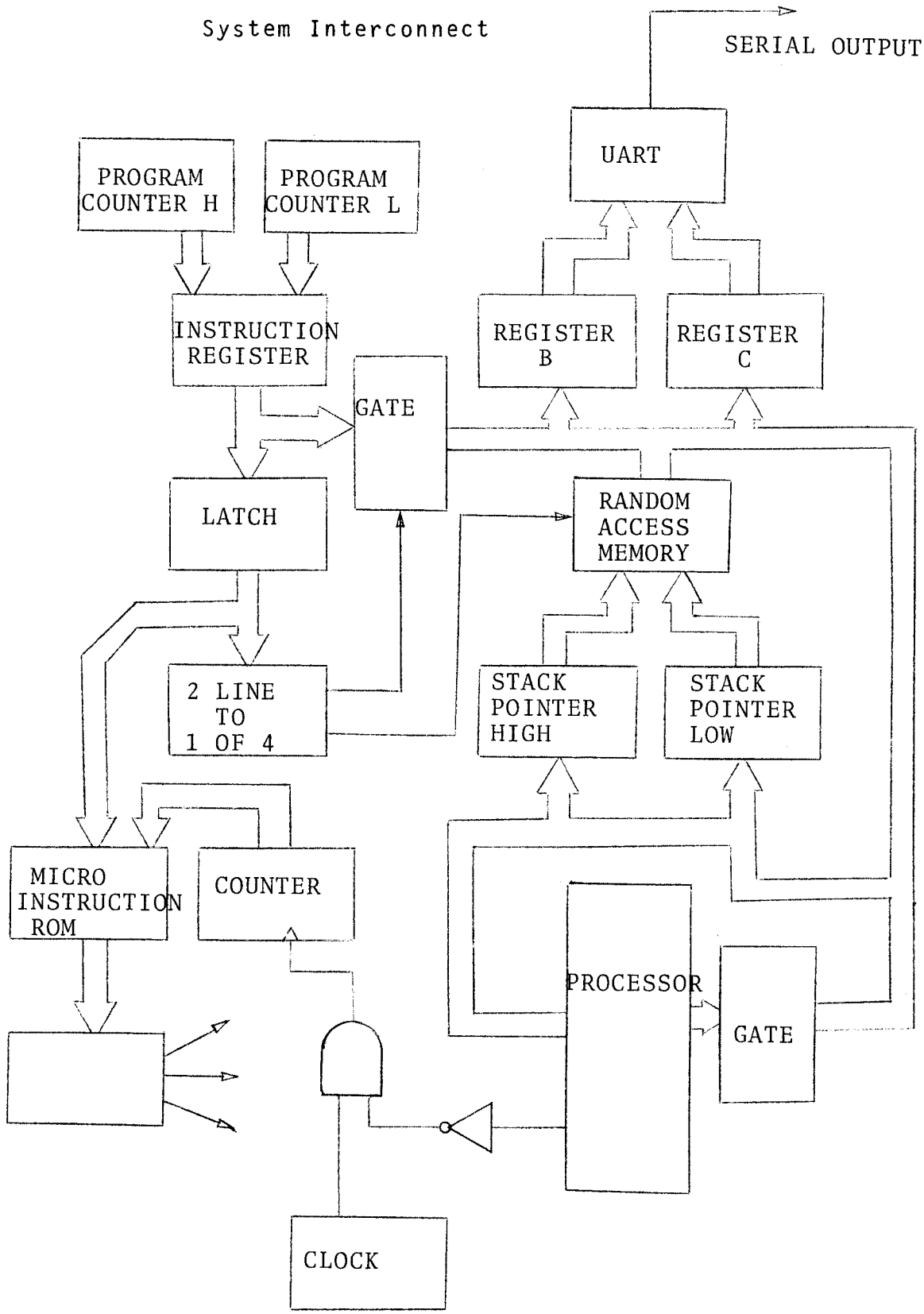



Figure 4.9 General relationship of all system components.

# CHAPTER V

## PROGRAMMING

Programming the unit is not difficult and the instructions were written so as to be easily learned. Actual programming is begun by turning the power switch to "ON" and momentarily depressing the "RESET" button. This switch zeroes the program counter. The user sets the four data switches to correspond to his first program step and depresses the "ENTER" button. This deposits the data from the front panel into the first location in memory. It also increments the program counter so that the next location in memory is addressed. This sequence is repeated until the entire program has been entered. The size of the program is limited only by the size of the memory installed in any particular system.

A simple programming example is now given. Although the power up sequence automatically resets the processor, it is good practice to begin each program by clearing the processor. It is noted from the appendix that the Clear instruction for the TMS 0117 processor is binary 10000. Since only a four bit data path is constructed in this unit, the fifth bit must be provided for differently. It is taken as an output from the most significant bit of the B register. Therefore to execute the clear instruction, the B register

is loaded with a 1 in the most significant position. For instance it may be loaded with 1000. The load processor immediate instruction is then executed, with the second data word being 0000. This operation resets the processor register to an all zero output.

The next step is to input from one of the external sensors. This is accomplished by setting the B register to the selected number and then executing the Input command. This command jams the Increment instruction into the processor for a predetermined length of time. When the instruction is completed, a digital representation of the selected input is left in the processor. This number is then used in any calculations desired by the user to determine appropriate control signals. These control numbers are then sent to the appropriate registers via the Store Processor command. Up to 256 output registers may be installed by the user and any combination of these may be used for a particular control signal. The store Processor command only transfers the Least Significant Bit of the processor to a register, therefore to make other digits available, the Shift Right command of the Processor must be used.

# CHAPTER VI

## CONCLUSION

Extensive experimentation was done in evaluating the CMOS circuitry. No difficulties were experienced and the devices used performed according to the specification sheets. The processor itself, the TMS0117, is well documented and it also performed as expected. The connection between the PMOS processor and the CMOS circuitry requires no special interface circuitry. That is, the specific devices are both voltage and current compatible.

The memories used were TTL circuits and did require external pull-up resistors. There are CMOS memories manufactured, but none were readily available at the time the unit was being tested.

The prototype was partially assembled on a Continental Specialties plug board, and no major difficulties were encountered.

This paper has presented an economical, yet effective, controller. Due to CMOS technology it has low power requirements and uses few parts. However, there are several improvements that would greatly extend the usefulness of the device.

The first of these would be in inclusion of circuitry enabling data from an entry device, such as a keyboard, to be transferred into the processor. Since the UART, already

included as a standard part, is a bi-directional device, this upgrade does not require complete redesign. Also the UART used in the model requires both +5 and -12 volt supplies. Since this is the only device requiring a -12 volt supply, a reduction in circuitry could be realized if a UART needing only +5 volts was used.

Finally, the parts count would be drastically diminished if an actual microprocessor were used. Although the microprocessor requires the user to develop more software, it is a much more powerful device. Also, the processor chosen here was a PMOS device and a significant reduction in power requirements would be achieved if a CMOS microprocessor were chosen. A minimal microprocessor system that outperforms the unit shown could be constructed using only several external parts. The amount of programming increases, but the unit would be even more flexible than the unit shown.

# APPENDIX

| Control Bit | | | | | Numeric Data |
|---|---|---|---|---|---|
| $A_5$ | $A_4$ | $A_3$ | $A_2$ | $A_1$ | |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 2 |
| 0 | 0 | 0 | 1 | 1 | 3 |
| 0 | 0 | 1 | 0 | 0 | 4 |
| 0 | 0 | 1 | 0 | 1 | 5 |
| 0 | 0 | 1 | 1 | 0 | 6 |
| 0 | 0 | 1 | 1 | 1 | 7 |
| 0 | 1 | 0 | 0 | 0 | 8 |
| 0 | 1 | 0 | 0 | 1 | 9 |
| | | | | | Instructional Codes |
| 1 | 0 | 0 | 0 | 0 | CLEAR |
| 1 | 0 | 0 | 0 | 1 | EQUALS |
| 1 | 0 | 0 | 1 | 0 | MULTIPLY |
| 1 | 0 | 0 | 1 | 1 | DIVIDE |
| 1 | 0 | 1 | 0 | 0 | ADD |
| 1 | 0 | 1 | 0 | 1 | ADD 1 |
| 1 | 0 | 1 | 1 | 0 | SUBTRACT |
| 1 | 0 | 1 | 1 | 1 | SUBTRACT 1 |
| 1 | 1 | 0 | 0 | 0 | ADD 1 TO OVERFLOW |
| 1 | 1 | 0 | 0 | 1 | SUBTRACT 1 TO ZERO |
| 1 | 1 | 0 | 1 | 0 | SHIFT RIGHT |
| 1 | 1 | 0 | 1 | 1 | SHIFT LEFT |
| 1 | 1 | 1 | 0 | 0 | EXCHANGE OPERANDS |

Table A1. Input Coding used by TMS 0117 Processor

| Hexadecimal Code | Microcode Instruction |
|---|---|
| 0 | Halt |
| 1 | Strobe B |
| 2 | Strobe C |
| 3 | Strobe UART |
| 4 | Strobe P |
| 5 | Strobe Stack Pointer (Low) |
| 6 | Strobe Stock Pointer (High) |
| 7 | Stobe Memory |
| 8 | Strobe Program counter (High) |
| 9 | Increment Stack Pointer |
| A | Decrement Stack Pointer |
| B | Increment Program Counter |
| C | Toggle Control Bit |
| D | Enable Input |
| E | Enable Output |
| F | Strobe A, reset microcode counter |

Table A2. Microcode used in Controller

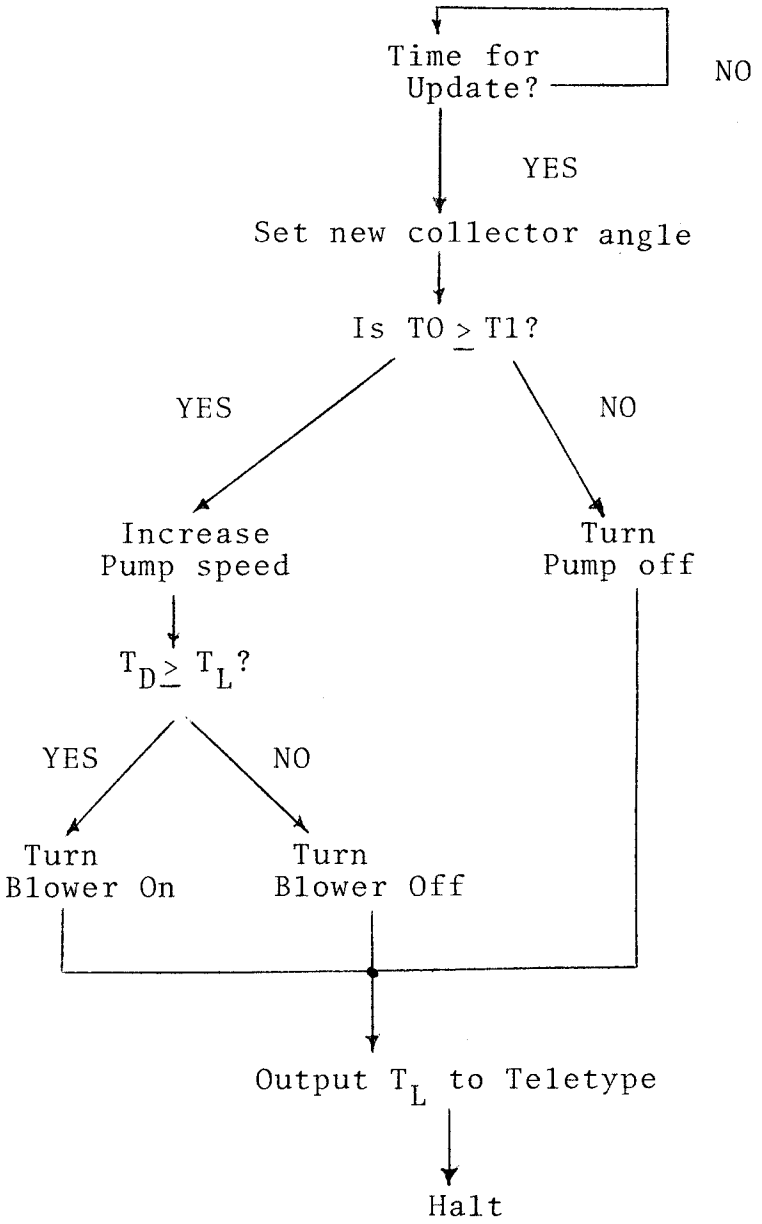



Figure A1. Flowchart used in developing sample program.

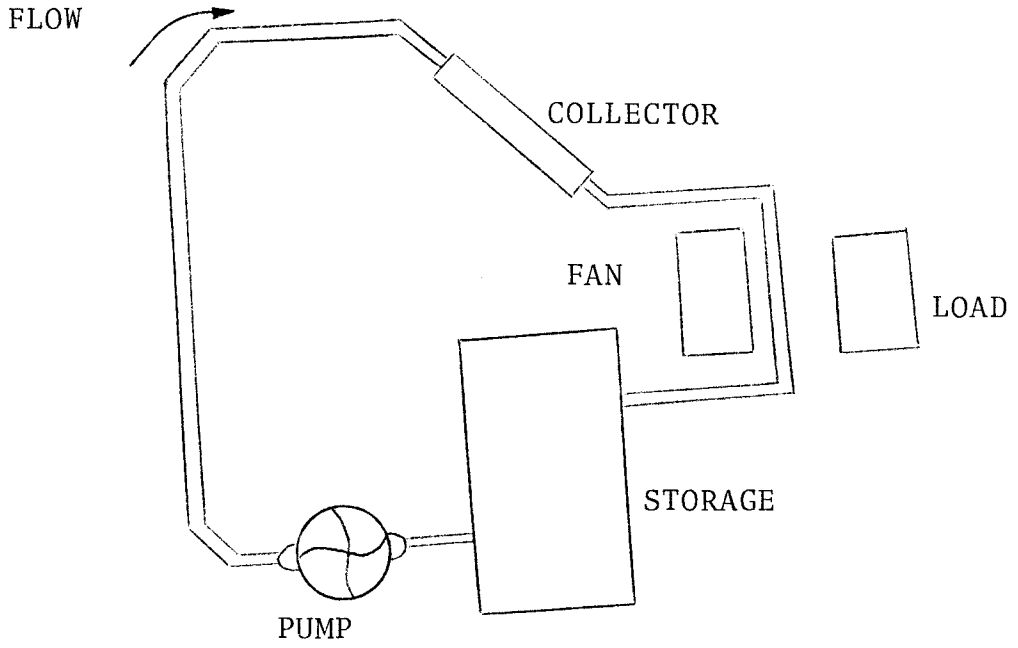
| Hexadecimal Code | Instruction | Date Transfer |
|---|---|---|
| $0_n$ | Load Processor Immediate | n P |
| 1 $n_1$ $n_2$ | Output Immediate | $n_1$ $n_2$ BC |
| 2 $n_1$ $n_2$ | Load Stack Pointer Immediate | $n_1$ $n_2$ $SP_1$ $SP_h$ |
| 3n | Branch on Positive | $PC_h$+n $PC_h$ (conditional) |
| 4 | Load Processor Direct | M P |
| 5n | Load Output Direct | M Output (B) |
| 6 | Pop | SP-1 SP;$M_{SP}$ P |
| 7 | Halt | 0 Timer |
| 8 | Store Processor Direct | P M |
| 9 | No operation | PC+1 PC |
| A | Load Output (C) | P Output (C) |
| B | Push | P M;SP+1 SP |
| C | Increment Stack Pointer | SP+1 SP |
| D | Decfement Stack Pointer | SP+1 SP |
| E $n_1$ $n_2$ | Enable Input | $n_1$ $n_2$ BC; Input |
| F | Strobe UART | UART transmits data |

Table A3. Coding used by programmer.

| Operator Command (Hexadecimal) | Microde |
|---|---|
| on | B,4,B,F |
| $1n_1n_2$ | B,1,B,2,B,F |
| $2n_1n_2$ | B,5,B,6,B,F |
| 3n | B,8,B,F |
| 4 | 4,B,F |
| 5n | E,B,F |
| 6 | A,4,B,F |
| 7 | 0 |
| 8 | 7,B,F, |
| 9 | B,F |
| A | E,B,F |
| B | 7,9,B,F |
| C | A,B,F |
| D | 9,B,F |
| E | (Add 1 jammed into processor for specified time) |
| F | 3,B,F |

Table A4. Composition of operator commands from microcode

A sample program is now shown, and it can be seen that even a relatively simple control system requires extensive programming. The system is shown below.




Inputs

| | |
|---|---|
| 0 | $T_0$ = Temperature of fluid entering collector |
| 1 | $T_1$ = Temperature of fluid leaving collector |
| 2 | $T_2$ = Temperature of load |
| 3 | $T_3$ = Temperature of storage fluid |
| 4 | $S_1$ = Status of fan (on/off) |
| 5 | $S_2$ = Speed of pump |
| 6 | $P_1$ = Position of collector |

Outputs

| | |
|---|---|
| 0 | $P_1$ = Position of collector |
| 1 | $S_1$ = Status of fan |
| 2,3,4 | $S_2$ = Speed of pump |
| 5 | $T_1$ = Teletype on/off control |

SAMPLE PROGRAM

| Address | Instruction Code | Comments |
|---|---|---|
| | ... Compute new collector angle ... | |
| 00 | 01 | Enter "1" into processor |
| 02 | A3 | Move processor to output 4. The least significant bit of output 4 is used as On/Off control for teletype. |
| 04 | 180 | Set Control bit. The most significant bit of the B register is used as the control bit by the processor. |
| 07 | 00 | Enter "Clear" into processor. |
| 09 | E06 | Sample input 6 (Collector angle). |
| 0C | 05 | Enter "Add 1" into processor. |
| 0E | A0 | Set new collector angle. |
| 10 | 00 | Enter "Clear" into processor. |
| | ... Compute T0-T1 ... | |
| 1B | F01 | Sample input 1, temperature of fluid leaving collector |
| 1C | B | Push least significant digit onto stack |
| 1D | 0A | Enter "Shift Right" command. |
| 1F | B | Push |

| | | |
|---|---|---|
| 20 | 0A | "Shift Right" |
| 22 | B | Push |
| 23 | 0A | "Shift Right" |
| 25 | F00 | Sample input 0, temperature of fluid entering collector |
| 28 | 06 | Enter "Subtract" into processor |
| 2A | 100 | Reset Control bit |
| 2D | 6 | Pop |
| 2E | 6 | Pop — Reenter T1 into processor |
| 2F | 6 | Pop |
| 30 | 180 | Set Control bit |
| 33 | 01 | Enter "Equals" (T0-T1=) |
| 35 | 31 | Branch to 46 if T0≥T1 |
| 37 | 00 | Enter "Clear" instruction |
| 39<br>3B<br>3D | A2<br>A3<br>A4 | Set pump speed (outputs 5,6, & 7) to 0. Energy was being lost by collector. |
| 3F | 7 | Halt |

... If program branches to 46, energy is being gained by collector. Therefore, pump speed will be increased ...

| | | |
|---|---|---|
| 46 | F07 | Sample input 5, pump speed |
| 49 | 02 | Enter "Add" |
| 4B | 100 | Reset Control bit |
| 4E | 02 | Enter "2" |
| 50 | 05 | Enter "5" |
| 52 | 180 | Set Control bit |
| 55 | 01 | Enter "=" Speed=Speed +25 |

| | | |
|---|---|---|
| 57 | A2 | Set least significant digit of pump speed |
| 59 | 0A | Enter "Shift Right" |
| 5B | A3 | Set next digit of pump speed |
| 5D | 0A | Enter "Shift Right" |
| 5F | A4 | Set most significant digit of pump speed |

... Now determine if the temperature of the load is satisfactory. Turn blower on to transfer heat to the load if it is below desired temperature ...

| | | |
|---|---|---|
| 61 | F02 | Sample load temperature |
| 64 | 06 | Enter "Subtract" |
| 66 | 100 | Reset Control bit |
| 69 | 01 | Enter "1 |
| 6B | 00 | Enter "0" 100 = desired temperature |
| 6D | 00 | Enter "0" |
| 6F | 180 | Set Control bit |
| 72 | 01 | Enter "Equals" |
| 74 | 31 | Branch to 85 if $T_L \geq 100$ |
| 76 | 100 | Reset Control bit |
| 79 | 01 | Enter "1" |
| 7B | F01 | Turn blower on |
| 7E | 7 | Halt |
| . | | |
| . | | |
| . | | |
| 85 | 00 | Enter "Clear" instuction |
| 88 | F01 | Turn blower off |

... Sample teletype output. Standard 7 bitt ASCII code is used ...

| | | |
|---|---|---|
| 8B | 14C | "L" |
| 8E | F | Strobe UART |
| 8F | 14F | "0" |
| 92 | F | Strobe UART |
| 93 | 141 | "A" |
| 96 | F | Strobe UART |
| 97 | 144 | "D" |
| 9A | F | Strobe UART |
| 9B | 120 | "Space" |
| 9E | F | "Space" |
| 9F | 154 | "T" |
| A2 | F | Strobe UART |
| A3 | 120 | "Space" |
| A6 | F | Strobe UART |
| A7 | 13D | "=" |
| AA | F | Strobe UART |
| AB | F02 | Sample load temperature |
| AE | 1B0 | Load B register with 3. This will convert BCD numbers to ASCII code. |
| B1 | B | Push |
| B2 | 0A | Enter "Shift Right" |
| B4 | B | Push |
| B5 | 0A | Enter "Shift Right" |
| B7 | AF | Move processor to register B. This is also output F. |

| | | |
|---|---|---|
| B9 | F | Strobe UART |
| BA | 6 | Pop |
| BB | AF | Move processor to register B |
| BD | F | Strobe UART |
| BE | 6 | Pop |
| BF | AF | Move processor to register B |
| C1 | F | Strobe UART |
| C2 | 180 | Set Control bit |
| C5 | 00 | Enter "Clear" command |
| C7 | A5 | Trun teletype off |
| C9 | 7 | Halt |

... End of Program ...

# BIBLIOGRAPHY

## Books

Barna, Arpard R. and Porat, Dan I. Integrated Circuits in Digital Electronics, New York: Wiley Interscience, 1978.

Hill, Fredrick J. and Peterson, Gerald R. Digital Systems: Hardware Organization and Design. New York: John Wiley and Sons, Inc. 1973.

Kostopoulos, George K. Digital Engineering. New York: Wiley Interscience, 1975.

Larsen, David G. and Rony, Peter R. Bugbook IIA: Interfacing & Scientific Data Communication Experiments. Derby, Connecticut: E & L Instruments, Inc. 1975.

McGlynn, Daniel R. Microprocessors: Technology, Architecture, and Applications. New York: Wiley Interscience, 1976.

Pippenger, D.E. and McCollum, C.L. Linear and Interface Circuits Applications. Dallas, Texas: Texas Instruments Learning Center Publications.

RCA COS/MOS Integrated Circuits Data Book. Printed in USA, 1975.