AD-A106 398 NORTHEASTERN UNIV BOSTON MASS ELECTRONICS RESEARCH LAB F/G 9/2
A PROGRAMMABLE CONTROL UNIT FOR A BALLOON-BORNE MASS SPECTROMET--ETC(U)
SEP 79 V C GEROUSIS F19628-78-C-0218
UNCLASSIFIED SCIENTIFIC-1 AFGL-TR-79-0225 NL

1 OF
AD
A106 398

6

AFGL-TR-79-0225

LEVEL II (12)

A PROGRAMMABLE CONTROL UNIT FOR A BALLOON-BORNE MASS SPECTROMETER BASED ON INTEL 8085A MICROPROCESSOR

Vassilios C. Gerousis

Northeastern University
Electronics Research Laboratory
Boston, Massachusetts 02115

SCIENTIFIC REPORT NO. 1

7 September 1979

Prepared for

Air Force Geophysics Laboratory
Air Force Systems Command
United States Air Force
Hanscom AFB, Massachusetts 01731

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(9) REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (18) AFGL-TR-79-0225	2. GOVT ACCESSION NO. AD-A106 398 (14)	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) A Programmable Control Unit for a Balloon-Borne Mass Spectrometer based on Intel 8085A Microprocessor		5. TYPE OF REPORT & PERIOD COVERED Scientific [redacted]-1
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Vassilios C. Gerousis (15)		8. CONTRACT OR GRANT NUMBER(s) F19628-78-C-0218
9. PERFORMING ORGANIZATION NAME AND ADDRESS Northeastern University Electronics Research Laboratory Boston, MA 02115		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 2310G3AM (16)
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Geophysics Laboratory Hanscom AFB, Massachusetts 01731 Monitor/ Alan D. Bailey/LKD (11)		12. REPORT DATE 7 Sep[redacted]79
		13. NUMBER OF PAGES 136 (12) 137
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Microprocessor Control
8085-Microprocessor
Programmable Control

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A programmable control unit (PCU) for a balloon-borne quadrupole mass spectrometer is described. The PCU controls the input parameters to the quadrupole mass filter, and the data acquisition from the electron multiplier. An overview of the principles underlying quadrupole mass spectrometer systems is briefly discussed. Design of the PCU is based on Intel's high-performance

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

1

8-bit microprocessor, the 8085A and its support chips. Both hardware and software designs with the theory of operation are discussed.

The system was designed and developed at Northeastern University under Air Force Contracts F19628-76-C-0256 and F19628-76-C-0218.

ACKNOWLEDGEMENTS

I thank my thesis advisor, Professor J. Spencer Rochefort for his advice and guidance and in making this work possible. I thank Alan D. Bailey of AFGL for his initial suggestions of the system, and Raimundas Sukys of Northeastern University for his advice and help with the hardware portion of the thesis.

TABLE OF CONTENTS

CHAPTER 1 - BASIC SYSTEM OPERATION

1.1 Introduction

The PCU-85 system, directed by a software program, controls the mass scanning operation of the quadrupole mass spectrometer. In the meantime, it performs sampling and storing of the data output in a buffer memory. At the end of each time unit, defined as the time required for the buffer memory to become full, the stored data is directly read from a computer terminal such as a CRT terminal.

Software routines have been developed to provide for the human interaction with the system, such as reprogrammability and changeability of parameters.

Figure 1-1 shows a block diagram of the interface between the PCU-85 and the mass spectrometer.

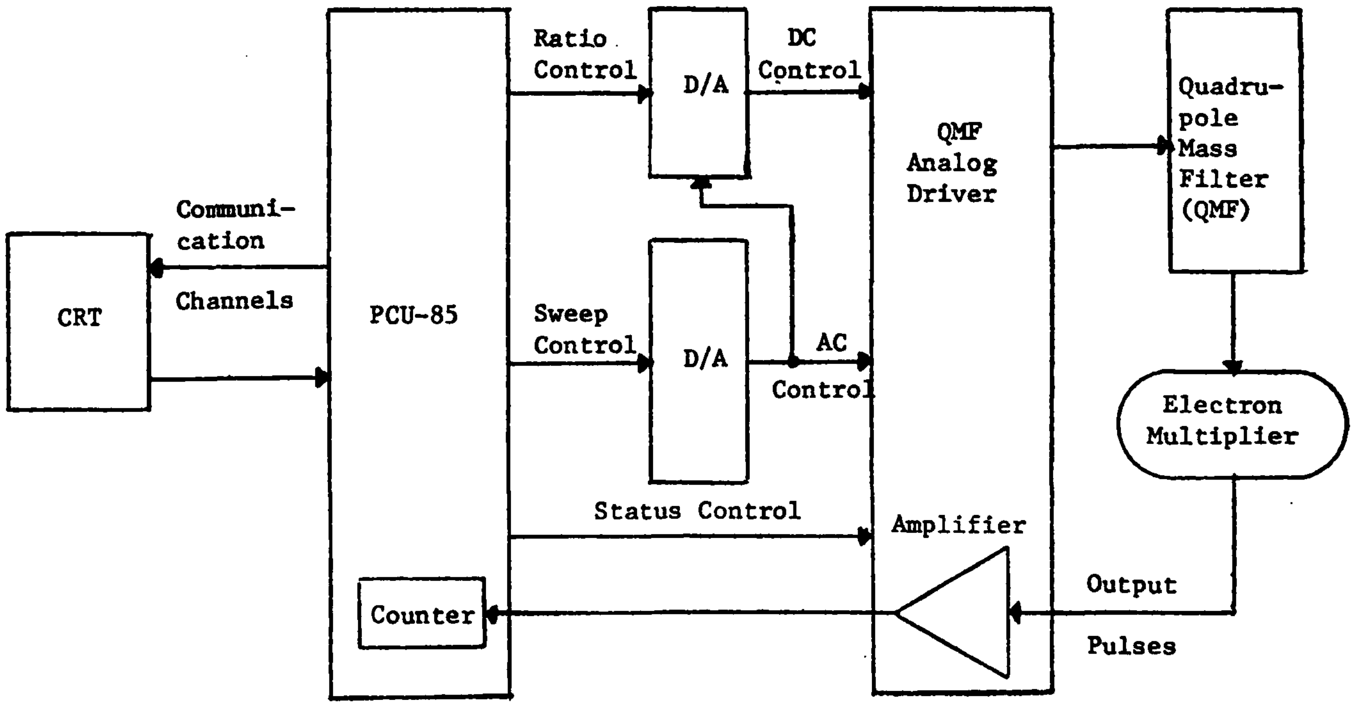


Figure 1-1. Interface between the PCU-85 and the analog driver of the Quadrupole Mass Filter.

1.2 Background

1.2.1 General Description

A balloon-borne ion mass spectrometer is being developed by the Aeronomy Division of the Air Force Geophysics Laboratory to investigate ambient ion and neutral clusters at an altitude of 30 to 40 kilometers. Positive and negative ions in the range of 14 to 1000 atomic mass units (amu) are to be detected.

The mass spectrometer consists of a quadrupole mass filter (QMF), some analog circuits to generate the necessary excitation signals for the filter, and the digital circuits to control the analog signal generator and to process the data. The mass filter is preceeded by a number of electrodes to focus the incoming ions into the quadrupole. The ions that pass through the filter are detected by an electron multiplier. Output pulses of the multiplier are counted. The counts in a given time period constitute the primary data.

1.2.2 Principles of Operation of a QMF

The quadrupole mass filter consists of four circular rods spaced 90 degrees apart in a parallel array shown in Figure 1-3. The excitation signals connected between the pairs of opposing rods consists of a dc component (U), and an RF component (V Cos ωt.). An ion injected in the longitudinal direction into the resulting electrical field oscillates between the opposing rods as it travels towards the electron multiplier. At some specified quadrupole field conditions, ions of a given mass undergo stable oscillations and reach the multiplier. Ions outside that mass range do not achieve stable oscillations and strike the rods.

The motion of a singly charged ion in the quadrupole field is described by Mathieu differential equations.[7] The equations can be solved in terms of the ion mass (m), the field radius (r_0) and the above mentioned quadrupole excitation signals. From the solutions the well known stability diagram of Figure 1-2 may be obtained. Any combination of the quadrupole parameters and ion masses that lie within the stable region result in passage of those masses to the multiplier.

The mass scan is performed by varying the amplitude of the quadrupole excitation signals while maintaining a constant U/V ration. The lines 1 and 2 in the stability diagram illustrate the effects of that ratio on the performance of the mass filter. The slope of each line is proportional to a given ratio. When the ratio is represented by line 2, the mass filter operates in a band-pass mode. All ions between masses M_1 and M_2 are passed. The two limits of the band-pass may be calculated from the stability diagram as:

$$M_1 = 4\ eV/q_1\ \ r_0^2\,\omega^2$$

$$M_2 = 4\ eV/q_2\ \ r_0^2\,\omega^2$$

where e is the charge of an ion. Operation with a ratio represented by line 1 produces a narrower band-pass. Thus the ratio controls the resolution of the instrument, while the amplitude of the excitation signal (frequency kept constant) determines the position of the filter within the mass spectrum.

Another useful mode of operation of the quadrupole filter is obtained when the ratio is reduced to zero. Under these conditions the instrument becomes a high pass mass filter.

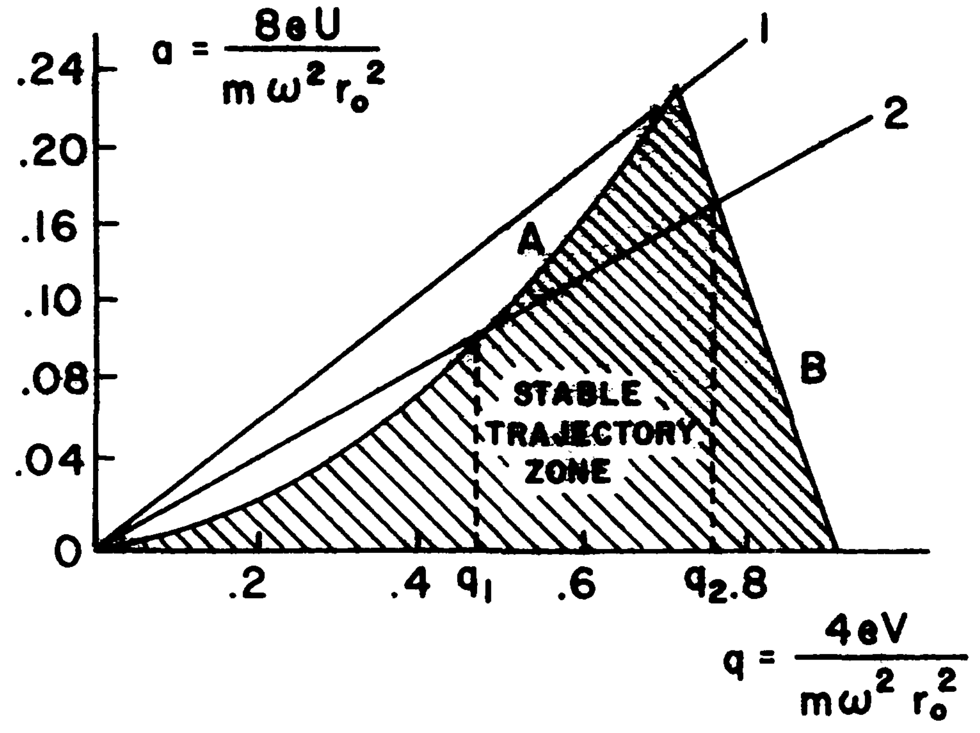


Figure 1-2. Stability diagram for quadrupole mass filter.

1.3 Mass Spectrometer Control

The task of the Programmable Control Unit (PCU-85) is to provide the digital signals to the analog circuits that determine the position and the bandwidth of the mass filter within the mass spectrum. The control unit also determines whether the filter operates in the band-pass or the high-pass modes. It establishes the necessary bias levels for the ion optics and discriminates between the positive and the negative ion species. Furthermore, the time interval spent examining a given amu domain is also defined by the program of the control unit. Limited data management capability is included among the other tasks of the PCU-85.

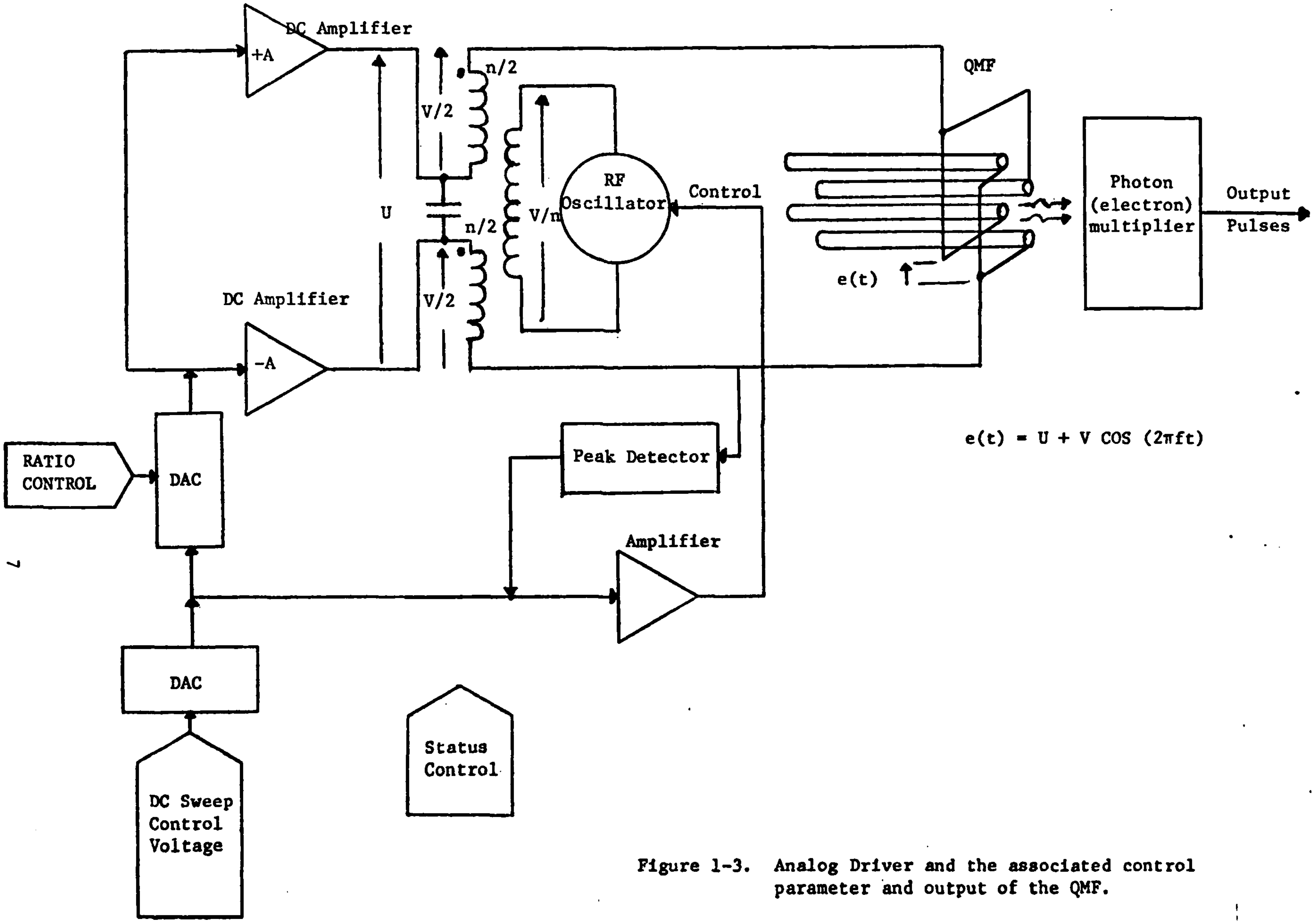
A dc voltage generated by an A to D converter controls the amplitude of the quadrupole excitation signals and thus determines the mass domain to be investigated. The control signal may be a ramp if the filter is to scan over a band of adjacent masses, or it may be a constant level when ions of a single mass are to be examined. The ramp is generated by incrementing the digital control word to the DAC. The slope is determined by the rate at which the PCU-85 increments the digital data. The slope as well as the end points of the ramp are defined by the resident control program. In this manner jumps and scans of differing rates may be combined into a program to examine areas of interest within the mass spectrum bounded by the capabilities of the instrument.

To control the resulution and the bandwidth of the mass filter the ratio U/V is digitally selected using a multiplying DAC. The same dc voltage that determines the amplitude of the RF signal to the quadrupole serves as the reference voltage for the DAC. The resulting output voltage is applied to a fixed gain amplifier and thus becomes the dc component of the quadrupole excitation signal.

Finally a number of control lines are used to select different modes of operation and to set some of the bias voltages. Band-pass or high-pass filter modes and the positive or negative ion detection modes are digitally selected. To extend the mass range covered by the filter the RF frequency may be switched. Lower frequency is used for the upper mass range; a higher frequency is selected for the lower range of the amu's. Since the selected mass is directly proportioned to the amplitude (V) and inversly proportional to the square of the RF frequency (ω^2), switching between the two frequencies reduces the dynamic range over which the amplitude of the RF signal must be controlled.

1.4 Mass Spectrometer Output

Ion impacts on the electron multiplier are counted. The count is allowed to accumulate within a controlled time interval. If a possibility of an overflow in the counter is detected the counting process is terminated prematurely. In any case, the accumulated count and the time interval during which the counter was active are transferred to a buffer memory. When this temporary storage is filled, the collected data becomes available for a storage in a bulk memory and/or transferred to some other device. In the present instrument, the data is displayed on CRT terminals.



$e(t) = U + V \cos(2\pi f t)$

Figure 1-3. Analog Driver and the associated control parameter and output of the QMF.

CHAPTER 2 - HARDWARE DESIGN

This chapter provides a functional description and a circuit analysis of the PCU-85 System.

2.1 Functional Description

As illustrated in Figure 2-1, the PCU-85 System is composed of the following blocks.

2.1.1 Intel 8085A Microprocessor (CPU)[1]

The CPU performs all the system processing functions and generates the address and control signals required to access memory and I/O ports.

Because the 8085A multiplexes the 8-bit data bus and the lower 8-bit address bus, the 8212 is used as a demultiplexer by latching the lower 8-bit address bus during the first part of every machine cycle (read or write cycle). At the same time, the 8155 latches the lower 8-bit address bus for its internal use. For the remainder of the machine cycle, the bus is used as a bidirectional data bus for memory and I/O data transfers.

Therefore, the 8085A outputs sixteen bits on the unidirectional address bus which are used to activate one of the devices connected on the bus for information transfer.

The control bus determines the direction and the transfer of data on the data bus within the proper timing windows dictated by the CPU operation characteristics.

The 8085A has five signal lines via which devices can request an interrupt. The CPU will respond by suspending its current operation and executes that specific interrupt service routine. Once completed, the CPU recovers the suspended program and continues its operation.

The serial I/O interface is accomplished via the Serial Input Data (SID) and Serial Output Data (SOD). Data on the SID and SOD lines are respectively

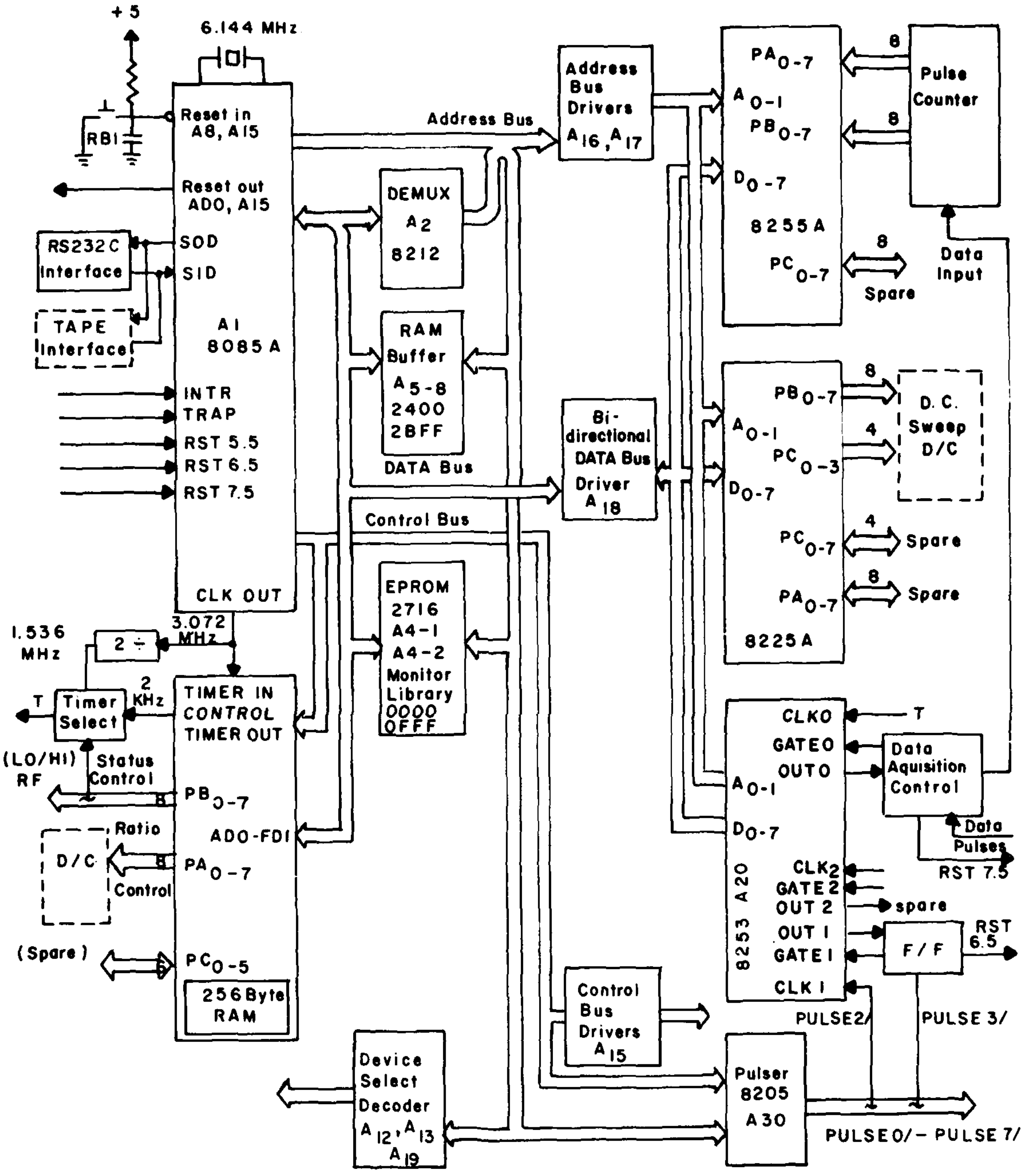


Figure 2-1 - PCU-85 System

transferred using RIM and SIM instructions. A level converter for the RS232C interface is provided primarily for testing the system. But other serial interface systems can be connected to these lines for communication or serial storage purposes.

2.1.2 Memory

Both Random Access Memory (RAM) and ultraviolet Erasable and Reprogrammable Read Only Memory (EPROM) are used in the PCU-85.

The PCU-85 provides 2K bytes of static memory for data storage in locations 2400 - 2BFB. The Intel 8155 RAM/IO/TIMER provides 256 bytes of static RAM in locations 2000-20FF, which are reserved for the system monitor use.

Two Intel 2716 EPROM are used to provide an additional 4K bytes of read-only memory. Locations 0000-07FF contain dedicated programs collectively defined as the PCU-85 system monitor. The remaining 2K bytes in locations 0800-0FFF contain the data library that are composed of blocks of five control parameters 16-bits each and blocks of pointers to the control parameters. The system monitor operates on the data library to control the PCU-85 system which in turn controls the mass spectrometer system operation.

2.1.3 Parallel I/O Interface and Programmable Timers

The parallel I/O interface consists of nine general purpose ports provided by the Intel 8155 and two Intel 8255A programmable peripheral interface (PPI). Each one of these ports can be programmed to be either an input port or an output port. Each device contains four internal registers - one register for command/status and one data register for each of the three ports.

Ports A and B in the 8155 are programmed in the output latched mode for Ratio and status control. The ports of 8155, when they are disconnected from the mass spectrometer sy s tem, can be used to program the 2716 and the 2758 EPROMs via a program that could be installed in the system monitor socket.

Ports B and the lower 4-bits of ports C in the 8225A (A21) are programmed in the output latched mode for dc sweep control.

Ports A and B in the 8225A (A22) are programmed in the input mode. These ports are connected to the output of a 16-bit counter. Whenever data pulses from the mass spectrometer output are to be sampled, the system monitor clears the counter and then enables counting. The counting process is stopped when a predetermined time interval has elapsed or when the counter becomes half full. An interrupt is generated so that the CPU can read the results and start the counting when needed.

The 8155 timer is a programmable 14-bit binary down counter that counts the input pulses and outputs either a square wave or a pulse when the "terminal-count" is reached. This timer is used to generate a 2kHz square wave.

The 8253 has three independent programmable 16-bit binary/decimal down counters that count the input pulses on the negative edge of input clock and output either a square wave or a pulse when the terminal-count is reached. Counter 0 is used to output a pulse whenever the terminal count of the time interval is reached. This pulse stops data collection and activates the RST 7.5 interrupt line. The clock input is selected to be either 1.536 MHz or 2 KHz by one of the signals (LO/HI)RF from the Status Control port. This gives uninterrupted time interval span between 0.651 μs. (period of 1.536 MHz clock) and 32.768 sec. (time for 2^{16} counts at 2 KHz clock).

Counter 1 is used to activate the RST 6.5 interrupt lines to indicate to the monitor that the RAM buffer is full. The monitor responds by filing the data into a non-volatile storage media. However, the present monitor transfers the data to be viewed on DEC scope VT 52 CRT Terminal. The monitor, starting from the beginning of the current break point, continues its control operation.

All I/O devices are interfaced to the bus structure as memory mapped I/O, that is, each device is assigned an area of memory address space. Locations 2100-21FF are reserved for memory mapped I/O devices. This hardware structure results in a significant increase in overall speed and at the same time reducing required program memory area.

The pulser which is a 1-out-of-8 decoder 8205 is interfaced to the bus system as an isolated I/O. When activated by software, it produces a 300 ns pulse on one of its selected output lines that are used for hardware control operation of some of the PCU-85 circuits.

2.2 Circuit Analysis

Both active-high and active-low signals are used in the PCU. A signal mnemonic that ends with a virgule (e.g., MEMR/) denotes that the signal is active-low ($\leq$ 0.4V). Conversely, a signal mnemonic without a virgule (e.g., ALE) denotes that the signal is active-high ($\geq$ 2.0V). Furthermore, a small circle at the start or at the end of a signal line in the diagrams produces the same result as the virgule does on the signal mnemonics. Therefore, the circle denotes that the signal is active-low. Conforming with industry standards, the small circle and the virgule have been used interchangeably in this paper.

The schematic diagram for PCU-85 is given in Appendix C. Each functional block of the PCU is reproduced in this chapter and analyzed.

2.2.1 Central Processor Unit CPU

2.2.1.1 Initialization

The 8085A is not guaranteed to work until 500 µsec after Vcc reaches 4.75V. Therefore, Intel suggests that RESET IN/ be kept low during this period. An RC network can satisfy this requirement. Therefore, the time constant T should be $T = RC \geq 500$ µsec. For reliable operation, T is found experimentally to be equal to 50 msec. With C = 1µf, R is calculated to be 50K. This is shown in Figure 2.2.

This power-up sequence is used to set the CPU to a known internal state. The 8085A responds by outputting a high pulse on RESET OUT and starts to

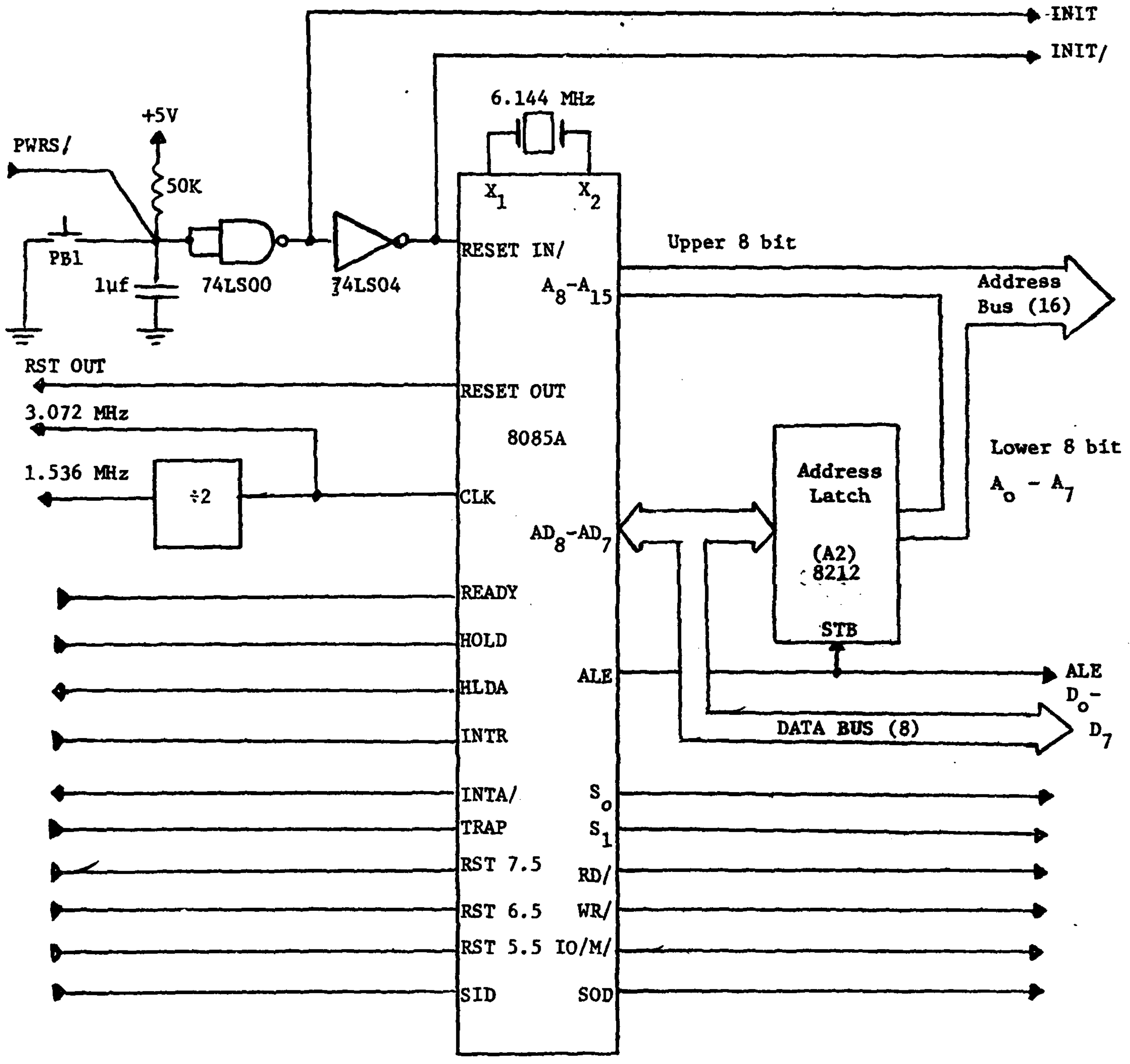


Figure 2-2. The 8085A and the 8212

execute instructions at location 0 with the interrupt system disable. The contents of the 8085A's registers (A, B, C, D, E, H, L) are changed with indeterminate results.

The RESET OUT, which is synchronized to the system clock, is provided for system reset. Whenever RESET IN/ is activated, RESET OUT initializes the PCU-85 to a known state. Reseting can be done manually via a push button or electrically by activating PWRS/line.

RESET IN/ input is provided with a Schmitt action input so that power-on reset only requires an RC network. The two gates that buffer the RESET IN/ from the RC network are included to provide TTL signals INIT/ and INIT for multiprocessor system synchronization.

2.2.1.2 The System Clock

The clock circuit is incorporated in the 8085A chip itself and provides two inputs: X_1 and X_2, that are connected to the 6.144 MHz crystal. The input frequency is divided by 2 to give the internal operating frequency which is available on the CLK output line. The CLK is used as the system clock. Since the maximum operating frequency of the timer 8253 is 2 MHz, the system clock is further divided to produce 1.536 MHz, to be used as an input to the timer.

2.2.1.3 Timing and Control

2.2.1.3.1 Bus Timing and Control: An instruction cycle is the time required to fetch and execute an instruction consisting of up to three bytes. Each instruction cycle consists of up to five machine cycles. A machine cycle (a READ or WRITE operation) is required each time the CPU accesses memory or an I/O port. The first machine cycle in every instruction cycle is an opcode fetch, even if the execution of that instruction requires no reference memory.

Each machine cycle consists of a minimum of three and a maximum of six state designated T_1 through T_6. A state is defined as the interval between two successive falling edges of the CPU clock which is equal to 326 nanoseconds. When the READY input to the CPU is pulsed low an integral number of wait states are inserted into the machine. This slows the speed of information transfer on the Data Bus to a compatible speed that the memory of I/O device can respond to the CPU requests.

Figure 2-3 illustrates the relationship between an instruction cycle, machine cycle, and T-state for a typical CPU instruction cycle.

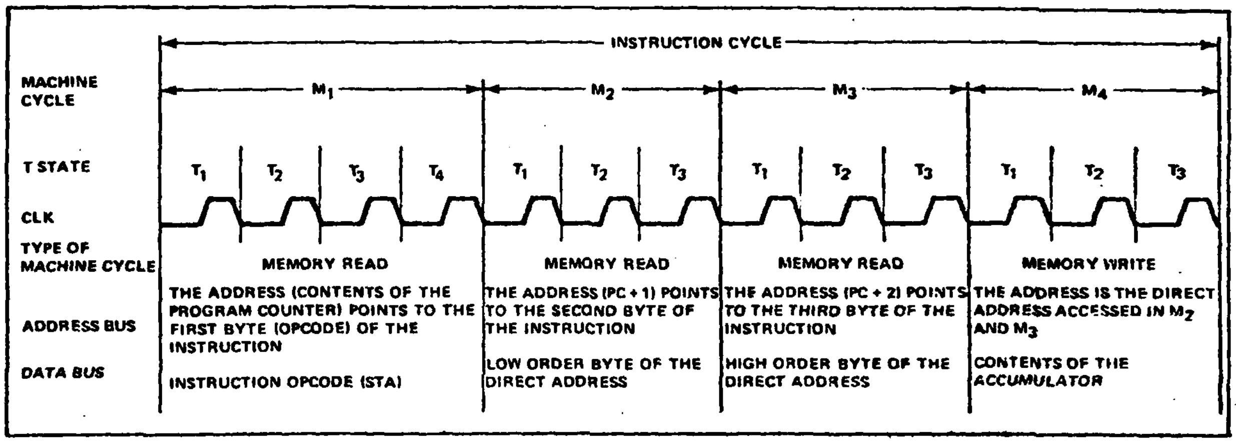

INSTRUCTION CYCLE
MACHINE CYCLE
M1
M2
M3
M4
T STATE
T1 T2 T3 T4
T1 T2 T3
T1 T2 T3
T1 T2 T3
CLK
TYPE OF MACHINE CYCLE
MEMORY READ
MEMORY READ
MEMORY READ
MEMORY WRITE
ADDRESS BUS
THE ADDRESS (CONTENTS OF THE PROGRAM COUNTER) POINTS TO THE FIRST BYTE (OPCODE) OF THE INSTRUCTION
THE ADDRESS (PC + 1) POINTS TO THE SECOND BYTE OF THE INSTRUCTION
THE ADDRESS (PC + 2) POINTS TO THE THIRD BYTE OF THE INSTRUCTION
THE ADDRESS IS THE DIRECT ADDRESS ACCESSED IN M2 AND M3
DATA BUS
INSTRUCTION OPCODE (STA)
LOW ORDER BYTE OF THE DIRECT ADDRESS
HIGH ORDER BYTE OF THE DIRECT ADDRESS
CONTENTS OF THE ACCUMULATOR

Figure 2-3. CPU timing for Store Accumulator Direct (STA) Instruction.

The 8085A uses a multiplexed Data Bus. The address is split between the higher 8-bit Address Bus and the lower 8-bit Address/Data Bus during the T_1-state of every machine cycle. The CPU pulses the Address Latch Enable (ALE) high to Latch the lower 8-bit of the address into the peripherals such as 8212 and 8155. ALE indicates the beginning of each machine cycle and is used as such in the debug circuitry.

During T_2 and T_3 states, RD/(WR/) is activated to transfer information into (from) the 8085A from (into) the selected device. Wait states are inserted during this time when READY line is pulled low. To differentiate between memory and I/O referencing, the CPU pulls IO/M/ line low to enable memory devices and pulls IO/M/ high to enable I/O devices. Decoding RD/, WR and IO/M/ signals, four OR gates and two inverters are used to provide four separate control signals MEMR/, MEMWR/, IORD/, IOWR/.

The CPU relinquishes the use of buses during the T_{RESET} state, T_{HALT} state, and T_{HOLD} state. T_{RESET} state occurs after RESET IN is pulled low. T_{HALT} state occurs after the execution of the HLT instruction, and T_{HOLD} state occurs after pulling HOLD input high. In these conditions, the Address, Data RD/, WR/, IO/M and ALE lines are tri-stated. Therefore, pull-up resistors of 50 kΩ each are provided for the main control signals RD/ WR/, and IO/M/.

The CPU responds to a HOLD request from another Master (such as a DMA controller or another 8085A) by raising HLDA line high. Once the HOLD request is removed, HLDA goes low,and the 8085 regains control of the buses.

<u>2.2.1.3.2 Interrupts</u>: Interrupt Request (INRT) input, a general purpose interrupt, when activated the requester device inserts a RESTART or CALL instruction synchronized by the Interrrupt Acknowledge (INTA/) signal issued by the 8085A. Since RESTART is a one byte instruction, it is usually used to jump to the interrupt service routine.

The CPU includes four vectored interrupts: TRAP, RST 7.5, RST 6.5, and RST 5.5. These restart interrupts cause an internal RESTART to be automatically inserted. Each of the three RST input (7.5, 6.5, and 5.5) has a programmable mask; TRAP is not maskable. The priority and vector location for each of these restart interrupts are given in Table 2-1.

<u>2.2.1.3.3 Status Information</u>: Partially encoded status of every machine cycle is directly available from the 8085A. S_o, S_1, and IO/M/ provide the system with advanced timing of the type of bus transfer being done. Table 2-2 shows the encoded status of the machine cycle.

Interrupt	Vector Location	Priority
Trap	24	Highest
RST 7.5	3C	Second
RST 6.5	34	Third
RST 5.5	2C	Lowest

Table 2-1. Interrupt Vector Memory Locations

Machine Cycle	IO/M/	S1	SO
X	0	0	0
Memory Write (MW)	0	0	1
Memory Read (MR)	0	1	0
Opcode Fetch (OF)	0	1	1
Halt *	1	0	0
I/O Write (IOW)	1	0	1
I/O Read (IOR)	1	1	0
Interrupt Acknowledge (INTA)	1	1	1

Table 2-2. 8085A Machine Cycle Chart.
X = Unspecified, * = IO/M/ is tristated but the 50k pull-up resistor pulls it high.

Figure 2-4 shows 74LS138, 1-out-of-8 decoder, used to provide output signals with the advanced status information given in Table 2-2. These outputs are used to directly drive small power LED indicators.

2.2.1.3.4 Serial I/O Communication: The data on the Serial Input Data Line SID is loaded into an accumulator bit 7 whenever a RIM instruction is executed. Similarly bit 7 is transferred to the Serial Output Data line SOD whenever a SIM instruction is executed.

Signals of RS232 are of -5V (logical one) and +5V (zero). After power-up

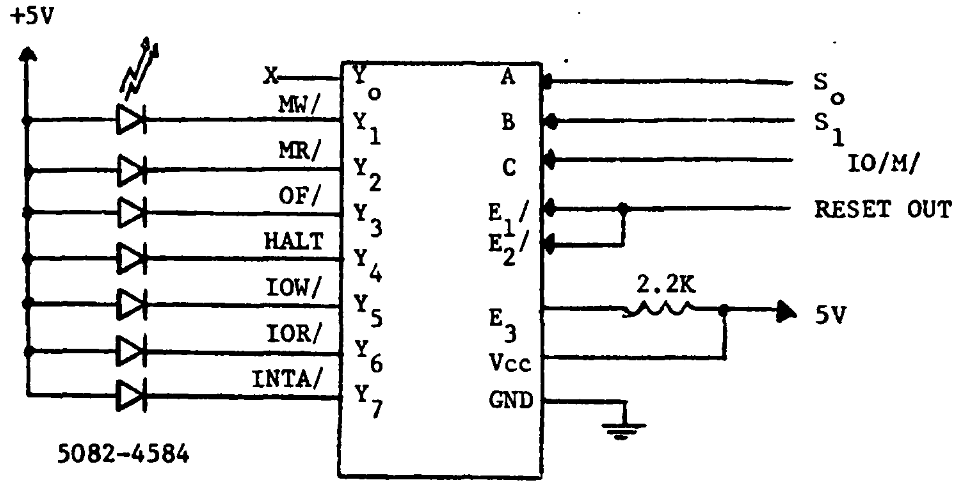


Figure 2.4. Decoding of the Advanced Status Information.

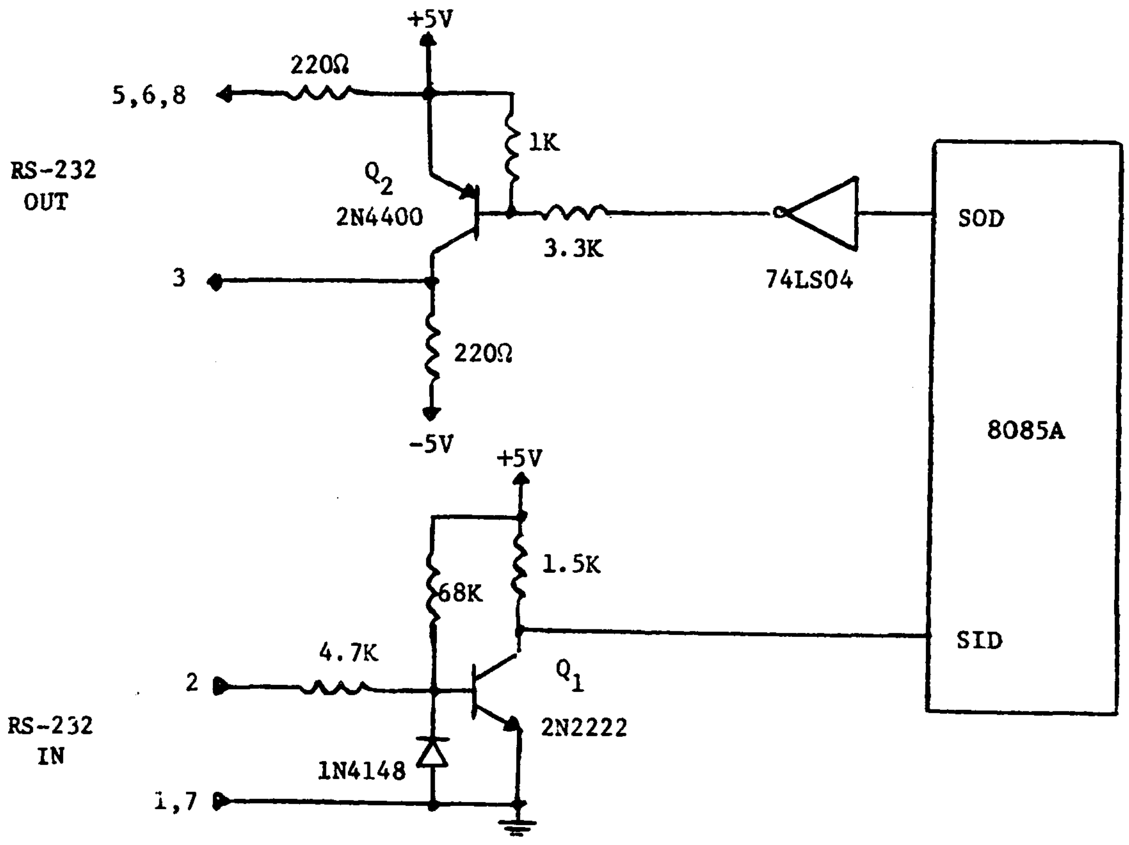


Figure 2-5. RS-232 interface circuits to SOD and SID. The numbers on the input and output lines indicates pin connections on the RS-232 connector (Cinch DB-255).

operation or following each reset, SOD is reset to zero. Logic zero is a start bit in an ASC11 word, therefore, a 74LS04 inverter is added between the SOD and the RS232 interface.

A high on the base of Q_2 turns off Q_2 and the output at pin 3 is -5V. With a low on the base of Q_2, the transistor is turned on and the output is at +5V. Q_1 also acts as an inverter with the output driven between 0 and +5V. The diode 1N4148 suppresses the negative going voltage on the base of Q_1 and protects its base. A better interface is the MC1489 and MC1488 but the 1488 requires higher positive and negative voltage between 9 and 15 volts.

2.2.1.3.5 DEBUG (Single Step) Circuit: The DEBUG circuit shown in Figure 2.6 enables the operator to examine the execution of a software program either a byte or an instruction at a time.

The heart of the circuit is 74120, dual pulse synchronizer. It generates either a single clock pulse when the mode control M is high or a train of clock pulses synchronized with the control inputs S_1, S_2, and R when M is low. The output pulses are started and stopped by the levels or pulses applied to the control inputs in accordance with the truth function in Table 2-3.

Clock pulses are passed to the output on a negative transition of 1S1(2S1) or 1S2(2S2) and are stopped by a negative transition on the reset input 1R(2R). But when either 1S1(2S1) is held low, pulses are passed regardless of the R input's status. The SR-latch 74LS279 is used to produce a low level on 2S2 when 2R/ of A39 is grounded by switch SW2. With the switch in this position, the WAIT/ position,A38-2 continues to pass clock pulses until a reset pulse is received from the OF/ or MC/ signal lines depending on the position of switch SW1.

ALE signal, inverted by a 74LS04 gate is used to indicate the beginning of a Machine Cycle (MC/). The Op code fetch (OF/) signal is developed when both

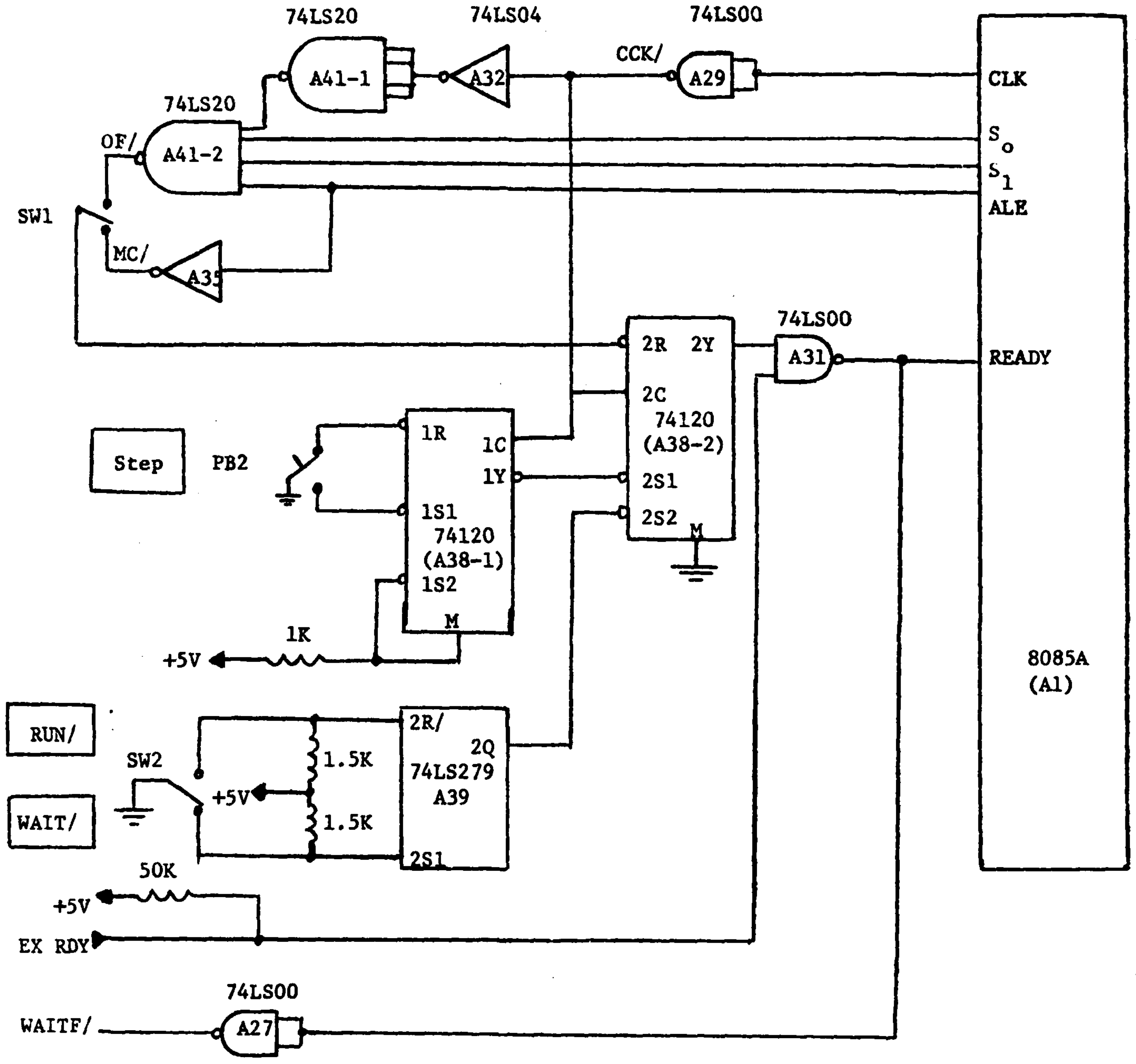


Figure 2-6. DEBUG (Single Step) Circuit.

Control Inputs			Function
R	S1	S2	
X	L	X	Pass Output Pulses
X	X	L	Pass Output Pulses
L	H	H	Inhibit Output Pulses
H		H	Start Output Pulses
H	H		Start Output Pulses
	H	H	Stop Output Pulses
H	H	H	Continue operation initiated by last transition

H = High Level	= Transition from H to L
L = Low Level	X = irrelevant

Table 2-3. Function Table for 74120

S_o and S_1 are each equal to logic one. Since S_o and S_1 are level signals, ALE and CLK/ are used to synchronize and gate one clock pulse at the output of the NAND gate A41-2. A32 and A41-1 produce a 20 ns delay in the CLK/ signal to allow S_o and S_1 to become stable levels since they change levels on the onset of each ALE pulse. Thus, the OF/ pulse marks the beginning of the fetch of the first byte of an instruction.

The 8085A samples the READY line just before the rising edge of the CLK of the T_2-state in every machine cycle. The clock pulses on the READY lines are delayed versions of the system clock. The 50 ns delay, produced via A29, A38-2 and A31 propagation delays, is long enough for the READY to be sampled high. Therefore, no wait states are inserted as long as these delayed clock pulses are generated on the READY line.

The single step circuit is activated when 2S2 is pulled down by SW2. Then OF/ or MC/ signal stops the CLK pulses into READY causing the processor to stop. The 8085A will wait in idle state until STEP/ push button is depressed. While waiting, the address bus will hold the address of the next instruction, and the data bus will hold the contents of that location. This information is always produced in T_1 and T_2 states of every machine cycle. When the STEP/ button is pressed, one clock pulse is gated to the output 1Y of A38-1 and is applied to 2S1 input of A38-2. The activation of 2S1 input causes the pulse synchronizer A38-2 to gate CLK pulses to the READY line and allows the processor to complete the machine cycle or the instruction cycle until the next reset pulse arrives from the mode selection switch SW1.

Repeated operation of the debug circuit allows the operator to examine the address bus and the data bus for a possible error in the software program.

2.3 Address Decoding

2.3.1 Memory Addressing

The CPU supplies 16 address lines to the memory to be used to access a specific byte memory location during a machine cycle. These 16-bits of the Address Bus are divided into two groups of signals. The low order bits of the bus are applied directly to the memory and the memory mapped I/O chips as address inputs while the high order bits are applied to the chip select decoding logic. The number of address bits assigned to each group is a function of the number of addressable locations contained in the chips used.

As shown in Figure 2-7, the PCU uses 1k of bytes chip selection lines K0/ through K15/.

The upper 6 bits of the Address Bus are used to generate 16 clusive chip-select lines via a 74LS154, 4-to-16 decoder(A12). This provides for 16K bytes of storage. The decoder is enabled by the active-low input signal A14, A15 and IO/M/ to select the lowest 16K bytes of storage. Other storage locations can be decoded by selective inversion of A14 and A15 and the addition of one decoder per 16K.

In the PCU system the lowest 8K bytes of storage is assigned to EPROM

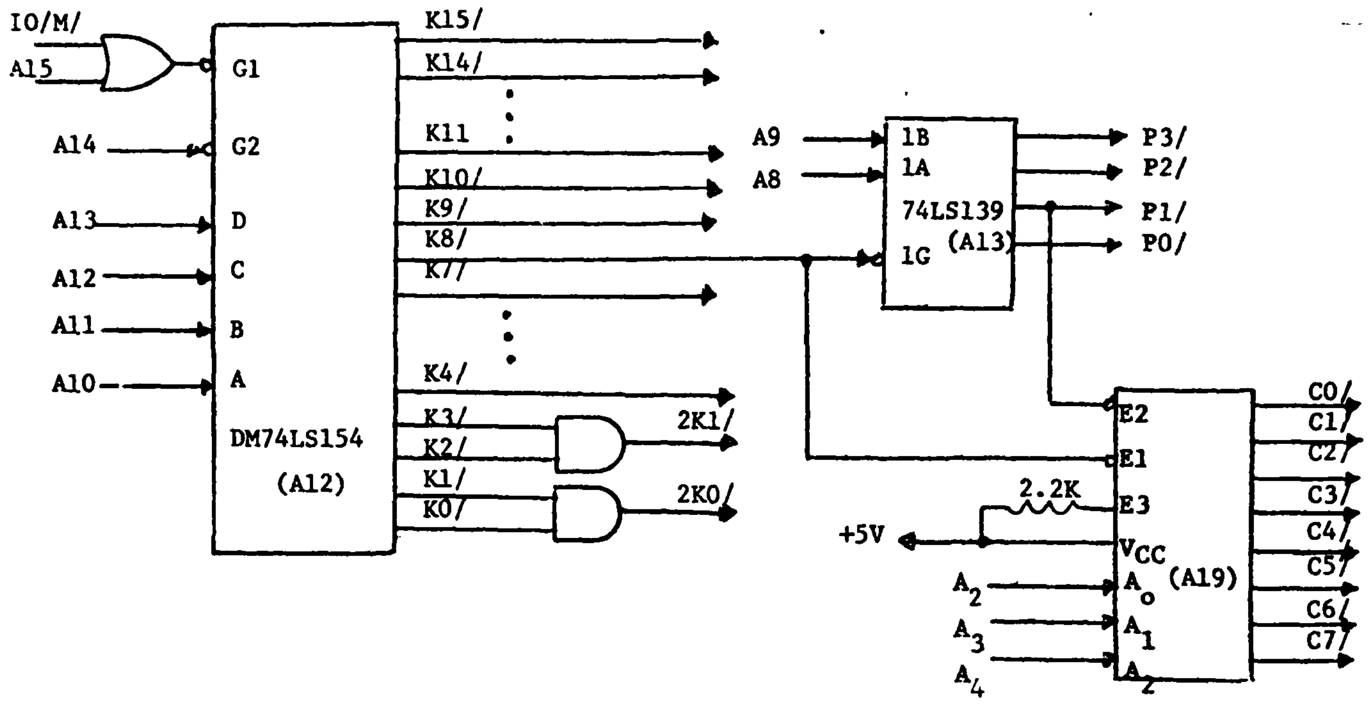


Figure 2-7. Chip Select Decoding Logic

selections, where the system monitor and data library occupy the lowest 4k bytes. Since the 2k bytes Intel 2716 EPROMs are used, the four lowest k-lines are combined to produce two select lines 2K0/ and 2K1. The first line selects the first 2k bytes of memory, and the second line selects the next 2k bytes.

The K8/ chip selection line is decoded further to provide for page (256 bytes) selection. A 74LS139, two-to-four decoder, (A13) converts K8/, A8 and A9 into four mutually exclusive select signals named P0/ through P3/. Table 2-4 gives a summary of page selection lines and the area of memory they select.

2.3.2 Memory Mapped I/O Addressing

An area of memory address space (2100-21FF) is assigned as I/O. This architecture allows the manipulation of the peripheral devices by using the same instructions that are used to manipulate memory locations. At the same time, a significant increase in overall speed and a reduction of program memory

Page Line	Area of Memory	Device Selection
P0/	2000-20FF	Working RAM memory 8155
P1/	2100-a1FF	Memory mapped I/O devices A9, A20, A21, A22
P2/	2200-22FF	Not used
P3/	2300-23FF	Not used

Table 2-4. Page Selection

area are achieved.

Most of Intel's programmable peripheral devices have four locations addressable by the least significant two bits of the Address Bus, A_o and A_1. Therefore, the upper bits can be used for chip selection decoding. In Figure 2-7, an 8205, three-to-eight decoder (A19) is shown. Address lines A_2, A_3 and A_4, which drive the 8205 inputs A_o, A_1, and A_2 respectively, are decoded to activate only one of the 8205 outputs named C0/ through C7/. The decoder is enabled by the two selection lines K8/ and P1/. Table 2-5 summarizes the used selection lines, associated devices and associated memory mapped I/O locations.

2.3.3 Isolated I/O Addressing

The 8085A separates the memory address space from I/O address space and uses two instructions, IN and OUT, for communication with the Accumulator. The second byte of each instruction contains the address of the I/O port which is duplicated onto both AD_o-AD_7 and A_8-A_{15}. The status signal IO/M/ when high indicates that data transfer is to or from an I/O port.

The PCU-85 uses an 8205, three-to-eight decoder, (A30), as I/O port shown in Figure 2-8. This port generates a low-active pulse onto one of its outputs, PULSE0/ through PULSE7/, selected by the address input lines A_o, A_1 and A_2. When IO/M/ and D8 have logic one level, the input E1/ is activated low via a

NAND gate 74LS00 (A29). But the 8205 is not enabled until the control signal WR/ is pulsed low. WR/ pulse duration defines the width of the pulse output of the 8205 which is equal to 406 ns. Table 2-6 gives a summary of the I/O address used to activate one of the 8205 outputs, and the control function associated with that output.

These output lines control various parts of the PCU-85 hardware system. Their control functions are utilized when an OUT instruction is executed with a port address 8X where X is a number between zero and seven specifying the output line to be pulsed.

Select Line	Device Selected	Register Selected	Address	Function
C0/	8155 (A9)	CSR	2100	Command/Status Register
		PA1	2101	Port A, Ratio Control Output
		PB1	2102	Port B, Mode Control Output
		PC1	2103	Spare
C1/	8155 (A9)	LBT	2104	Low Order Byte of Timer 2KHz
		HBT	2105	High Order Byte of Timer clock
			2106 2107	Not used
C2/	8255A (A20)	PA2	2108	Port A Connected to 74LS393
		PB2	2109	Port B Counter for data Counter
		PC2	210A	Spare
		CR	210B	Control Register
C3/	8253 (A22)	CR0	210C	Counter Register Zero, Time Interval Count
		CR1	210D	Counter Register One, Number of Data Collected
		CR2	210E	Counter Register Two (Spare)
		CWR	210F	Control Word Register
C4/	8255A (A21)	PA3	2110	Port A D.C. Sweep
		PB3	2110	Port B Generator
		PC3	2112	Port C, Spare
		CR	2113	Control Register

Table 2-5. Memory Mapped I/O ports and their assigned addresses.

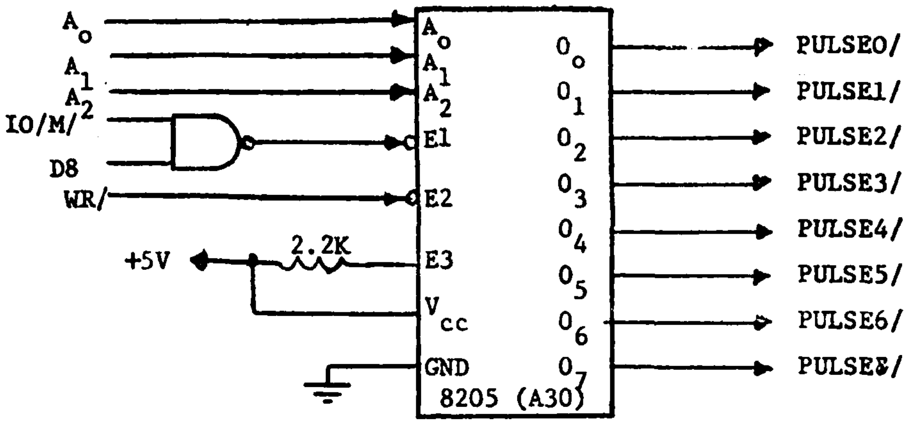


Figure 2-8. Pulser Circuit as an isolated I/O device.

I/O Address	Output Activated	Function
80	PULSE0/	Clear the data counter, data enable and time interval flip flops.
81	PULSE1/	Set data and time interval flip flops (A25).
82	PULSE2/	Decrement Counter One of 8253.
83	PULSE3/	Initiate Decrementation of Counter One and clear its flip flop (A11).
84 - 87	PULSE4/ through PULSE9/	Spare

Table 2-6. Address of the pulser outputs and functions.

2.4 Memory

2.4.1 Read-Only Memory (ROM)

PCU-85 utilizes 4k bytes of ROM comprised of two Intel 2716 EPROMs A4-1 and A4-2. The 2716 is a 16,384-bit ultraviolet erasable and electrically programmable read-only memory (EPROM). The lowest eleven lines of the address bus, A_o - A_{10}, are connected to their counterpart pins on the EPROM. When the chip select (CS/) input is activated, the signals on the address inputs of the EPROM are internally decoded to select one of the 2K bytes of memory locations.

Strobing the output enables (OE/) input to the 2716, releases the output lines $D_o - D_7$ from the high impedance state and allows the content of the selected memory locations to drive these lines.

As shown in Figure 2-9, two Intel 2716 EPROMs, (A4-1) and (A4-2), are connected to the Data Bus and Address Bus. The OE/ input on each device is strobed by the control line RD/. The chip selection lines 2KO/ and 2K1/ are connected to CS/ inputs of (A4-1) and A4-2) respectively. When one of the chips is selected, the read operation is initiated during T_2 state when the CPU pulls the RD/ line low which controls data from that chip on and off the Data Bus by way of OE/ input.

A4-1 with address 0000-07FF contains the system monitor program, while A4-2 with address 0800-0FFF contains the Data Library.

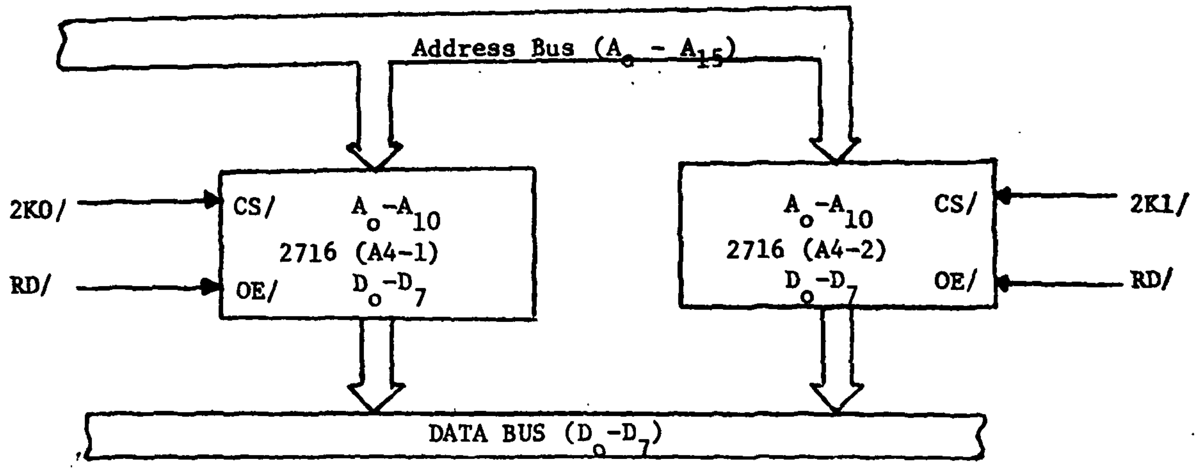


Figure 2-9. Two Intel 2716 EPROMs

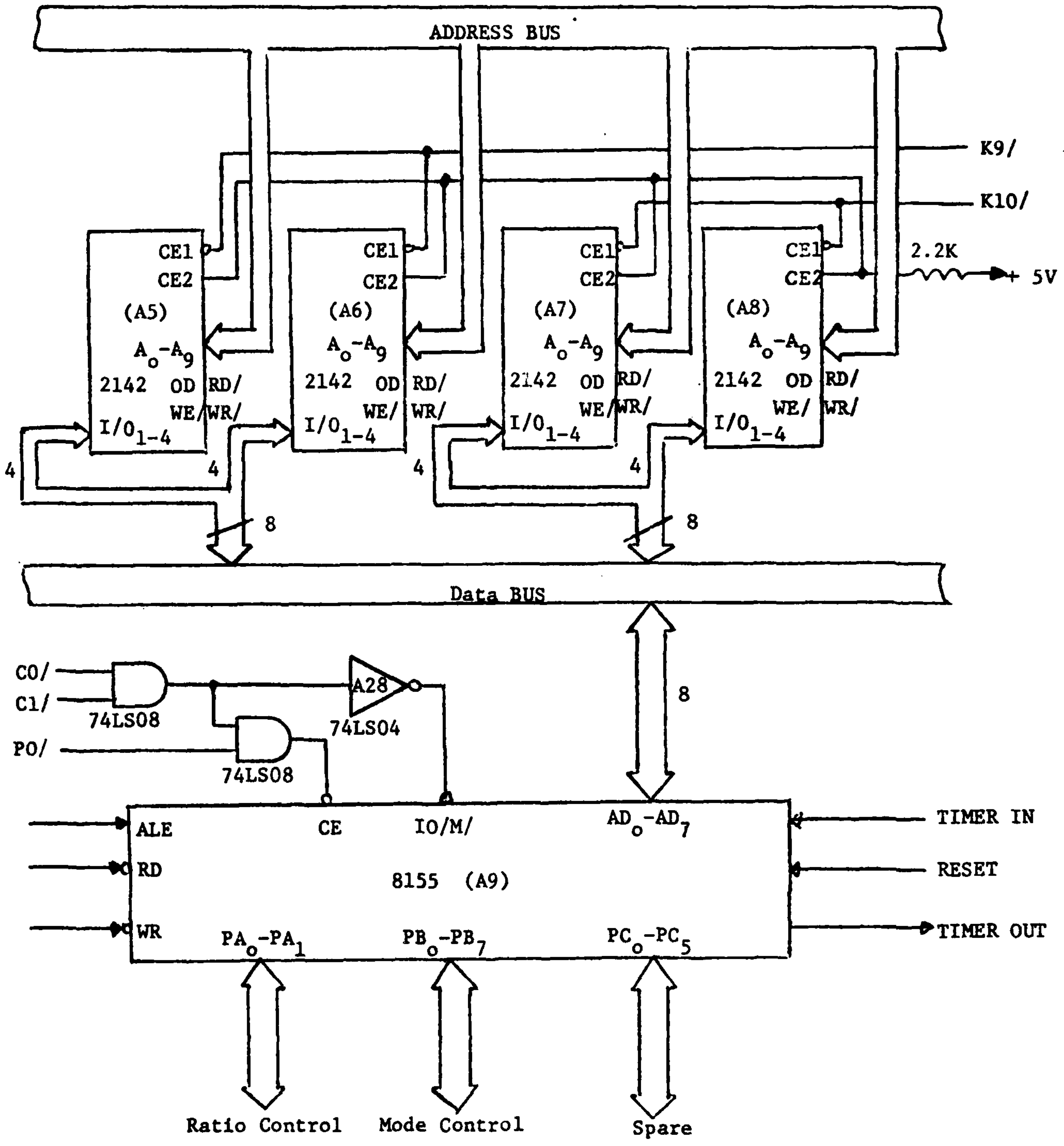


Figure 2-10. Memory System

2.4.2 Random Access Memory (RAM)

The PCU-85 includes 2048 bytes of static RAM in (A5) - (A8) by using four Intel 2142 memory devices for temporary storage of sampled data. The 256 bytes of RAM in 8155 (A9) is used as a working RAM to store pointers, parameters and the stack. Memory address block 2400-2BFF is contained in A5 - A8 and memory address block 2000-20FF is contained in A9.

2.4.2.1 Intel 2142 RAM

The Intel 2142 is a 4096-bit static Random Access Memory organized as 1024 words by 4-bits. Two chip selects (CS1/ and CS2) are provided for selection of individual packages when outputs are OR-tied. An output disable (OD) is included for direct control of the output buffers.

As shown in Figure 2-10, four of 2142 devices are organized in groups of two packages. (A5) and (A6) are selected by the chip select signal K9/, while (A7) and (A8) are selected by the chip select signal K10/. All chip enable inputs (CE2) are connected to one 2.2 KΩ pull-up resistor. Therefore, when CE1/ is driven low by K9/ or K10/, memory address block 2400-2BFF is selected.

When (A5) - (A6) or (A7) - (A8) is enabled, the target location is specified by address bits A_0 - A_9. During a read operation, the control line RD/ is activated low which enables the output buffer on the selected 2142 RAM. However, the data output buffer is disabled during a write operation by a false (high) RD/ signal applied to the Output Disable (OD) input pin. Thus, a write operation occurs to the selected target when WR/ is activated low by the CPU.

2.4.2.2 8155 RAM

The 8155, shown in Figure 2-10, is a RAM with I/O ports and timer. The RAM portion is designed with 2k bit static cells organized as 256 x 8. The 8155 is enabled by driving the CE input low. This is accomplished by activating the memory chip selection line P0/, or one of the memory-mapped chip selection lines C0/ and C1/. When low, IO/M/ input selects the memory section and when high it selects the I/O ports (PA,PB,PC) and timer section. IO/M/is always low, unless C0/ or C1/ is activated low which drives the IO/M/high via the inverter (A28).

The trailing edge of the address latch enable (ALE) signal latches in the target address bits specified by data bus bits AD0-AD7. During a write operation, the CPU WR/ is true and write occurs; during a read operation, the CPU RD/ output is true and a read occurs. Data is read from or is written into (A9) via its address/data pins AD0-AD7. The Reset input initializes the three I/O ports to input modes. Timer In and Timer Out are input and output pins respectively, and are associated with the 14 bit timer in the 8155.

2.5 BUS Buffers and Loading Calculation

A crucial point in the design of a bus-structured system is the consideration of the dynamic and the static loading of each device connected to the system bus. For critical loads, buffers should be added to provide driving capability and to add a safe margin in the operating conditions of all devices with surrounding environment changes such as temperature and humidity. In addition to increasing the relaibility of the system, buffers allow more devices to be connected to the system up to a limit.

In Figure 2-1, buffers for the system bus are provided in the circuit not only to enhance the driving capability of the three buses but also to reduce their dynamic loadings.

2.5.1 Static Loadings

As a general rule, loading on the bus must be kept less than the current driving capability of the weakest driver in the system. The PCU-85 has two types of bus structures. The first type is a uni-directional bus that consists of the Address Bus and the Control Bus onto which information is transferred in either direction at a time. Table 2-6 tabulates all the devices that are used in the PCU-85 system and the corresponding electrical characteristics. The contents of the table will be used in the System Bus Loading Calculation.

DEVICE	I_{IH}(MA)	I_{IL}(MA)	I_{OH}(MA)	I_{OL}(MA)	C_{IN}(pF)	C_{OUT}(pF)	Testload for A.C. Characteristics
8085A	0.010	0.010	-0.400	2.0	10	20	150 pF (2)
2142	0.010	-0.010	-1.000	2.1	5	5	100 pF
8155	0.010	-0.010	-0.400	2.0	10	20	150 pF (2)
8205	0.010	-0.250	-1.500	10.0	5	7	150 pF (3)
8212	0.010	-0.250	-1.000	15.0	9	12	150 pF (3)
8253	0.010	-0.010	-0.400	2.0	10	20	100 pF
8255A	0.010	-0.010	-0.400	2.5	10	20	100 pF
2716	0.010	-0.010	-0.400	2.1	6	12	100 pF
74LS244	0.020	-0.200	-15.0	24.0	10	12	(4) can drive 1000 pF
74LS245	0.020	-0.200	-15.0	24.0	10	12	(4) can drive 1000 pF
DM74LS154	0.020	-0.360	-0.400	8.0	10	12	(4) can drive 1000 pF
74LS00	0.020	-0.400	-0.400	8.0	5	7	(4)
74LS04	0.020	-0.400	-0.400	8.0	5	7	(4)
74LS08	0.020	-0.400	-0.400	8.0	5	7	(4)
74LS32	0.020	-0.400	-0.400	8.0	5	7	(4)
74LS139	0.020	-0.400	-0.400	8.0	5	7	(4)
74LS138	0.020	-0.400	-0.400	8.0	5	7	(4)
74LS38	0.020	-0.400	-0.250	24.0	5	7	(4)

Table 2-6. Electrical characteristics over recommended free-air temperature range of the devices connected to the bus structure.

1 - Current out of a terminal is given as negative value.
2 - C_{IN} and C_{OUT} of Intel devices are less than 10 and 20 pF respectively as quoted by Intel's application engineer in California.
3 - These values are sampled as typical test load from the manufacturer's "data to output delay versus load capacitance" graph. In this graph, the maximum propagation delay, specified in the data sheet, is introduced by a capacitive load of 300 pf for the 8212, and of 200 pf for 8205.
4 - Capacitance values were quoted by T.I.'s application engineer in Dallas.

2.5.1.1 Address Bus

Because of the multiplexed bus structure of the 8085A, the addition of 8212 for demultiplexing the data bus provides for the incidental buffering action for the lower 8 bits of the address bus. Tables 2-7 and 2-8 tabulate the devices connected to the lower 8 bits and upper 8 bits of the Address Bus, and their corresponding input current drive. The total load on the Address Bus is computed and subtracted from the drive capability of the 8085A and the 8212. In both cases, there is a substantial amount of reserve drive for both logic-low and logic-high situations.

A similar calculation procedure is carried out in Table 2-9 for the lower 8 bits of the Buffered Address Bus with the same results. The upper 8 bits of the Buffered Address Bus are not connected to any device, but are provided for system expansion.

DEVICE	I_{IH}(MA)	I_{IL}(MA)
74LS154, Decoder	0.020	0.360
74LS139, Decoder	0.020	0.400
2716 (TWO), EPROM	0.020	0.020
2142 (FOUR), RAM	0.040	0.040
74LS244, Buffer	0.020	0.200
TOTAL LOAD	0.120	1.020
Nominal drive from 8085A outputs	0.400	2.000
Reserve drive	0.280	0.880

Table 2-7. Static loading for the Upper 8 bits of the Address Bus.

It is observed from Tables 2-7, 2-8 and 2-9 that the Address Bus buffers 74LS244, (A16) and (A17), can be removed and leave the Address Bus with almost the same substantial amount of reserve drive. But, dynamic loading calculations show that these buffers are required to reduce the capacitive loading on the bus.

DEVICE	I_{IH}(MA)	I_{IL}(MA)
2142 (FOUR), RAM	0.04	0.040
2716 (TWO), EPROM	0.02	0.020
74LS244, Buffer	0.02	0.200
TOTAL LOAD	0.08	0.260
Nominal drive from 8212 outputs	1.00	15.000
Reserve drive	0.92	14.740

Table 2-8. Static loading for the Lower 8 bits of the Address Bus.

DEVICE	I_{IH}(MA)	I_{IL}(MA)
8205, Decoder	0.010	0.250
8255A (TWO), Programmable Peripheral Interface (PPI)	0.020	0.020
8253	0.020	0.020
TOTAL LOAD	0.050	0.290
Nominal drive from 74LS244 (A16)	15.000	24.000
Reserve drive	14.950	23.710

Table 2-9. Static loading for the buffered lower 8 bits of the Address Bus.

2.5.1.2 Control Bus

Figure 2-11 shows Control Bus Buffer using 74LS244, a 3-state output octal buffer. The buffer is enabled all the time, since its control inputs 1G/ and 2G/ are grounded.

An example of static loading calculation of the control bus is shown in Table 2-10 for RD/ control line, as a typical control line.

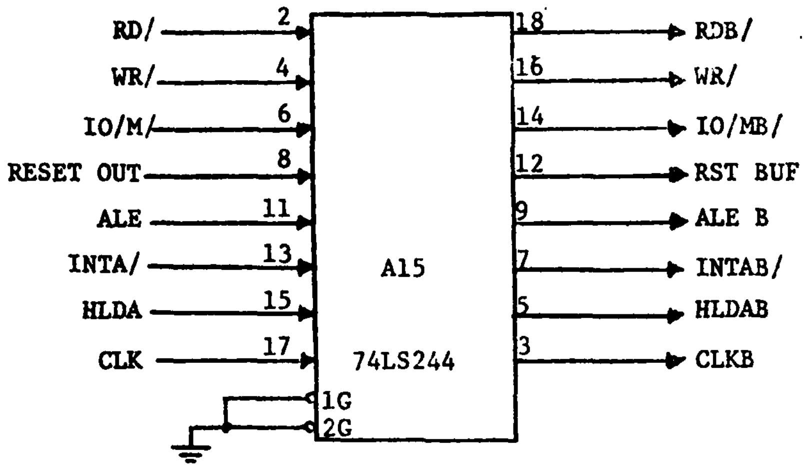


Figure 2-11. Control Bus Buffer.

DEVICE	I_{IH}(MA)	I_{IL}(MA)
2142 (FOUR), RAM	0.040	0.040
8155, RAM/IO/TIMER	0.010	0.010
74LS32 (TWO) OR gate	0.040	0.800
74LS244, Buffer	0.020	0.400
TOTAL LOAD	0.110	1.250
Nominal drive from 8085A	0.400	2.00
Reserve drive	0.290	0.750

Table 2-10. Static loading for RD/ control line.

2.5.1.3 Data Bus

The Data Bus, being bi-directional by nature, requires each drive that transfers information into it to be capable of driving the rest of the devices on the bus. These devices, when enabled, must be capable of driving the Intel's 8085A Data Bus. They are:

1. The 8085A, which drives the bus during all write operations.
2. The 8155, which drives the bus during read operations.
3. Two of the 2142, which drive the bus during read operations.
4. The 2716, which drives the bus during read operations.
5. The 74LS245, which drives the bus during read operations.

The first four devices have the lowest, although equal, driving capability. Proper operation of the system requires the total load on the Data Bus to be less than any of the first four devices' drive capability. The loads presented to the bus are summarized with respect to the 8085A microprocessor shown in Table 2-11. There is a substantial amount of reserve drive on the Data Bus.

DEVICE	I_{IH}	I_{IL} (MA)
2716 (TWO), EPROM	0.020	0.020
2142 (TWO), RAM	0.020	0.020
8155, RAM/I/O/TIMER	0.010	0.010
8212, Latch	0.010	0.250
74LS00, Nand Gate connected to D8	0.020	0.400
74LS245, Buffer	0.020	0.200
TOTAL LOAD	0.100	0.900
Nominal drive from 8085A	0.400	2.000
Reserve drive	0.300	1.100

Table 2-11. Static loading calculation for the Data Bus.

There is a substantial amount of reserve drive on the Data Bus.

Figure 2-12 shows 74LS245, an octal bus transceiver, that is used to buffer the 8085A Data Bus. The outputs are enabled when P1/, the memory mapped selection line, activates the control pin G/. P1/ also activates the control pins of the Address Bus Buffers 74LS244.

The direction of data transfer from the A-Bus to the B-Bus or from B-Bus to the A-Bus depends upon the logic level at the direction control (DIR) input, as shown in Table 2-12.

ENABLE G1	DIRECTION CONTROL (DIR)	OPERATION
L	L	B data to A bus
L	H	A data to B bus
H	X	Isolation

Table 2-12. Function Table of 74LS245.

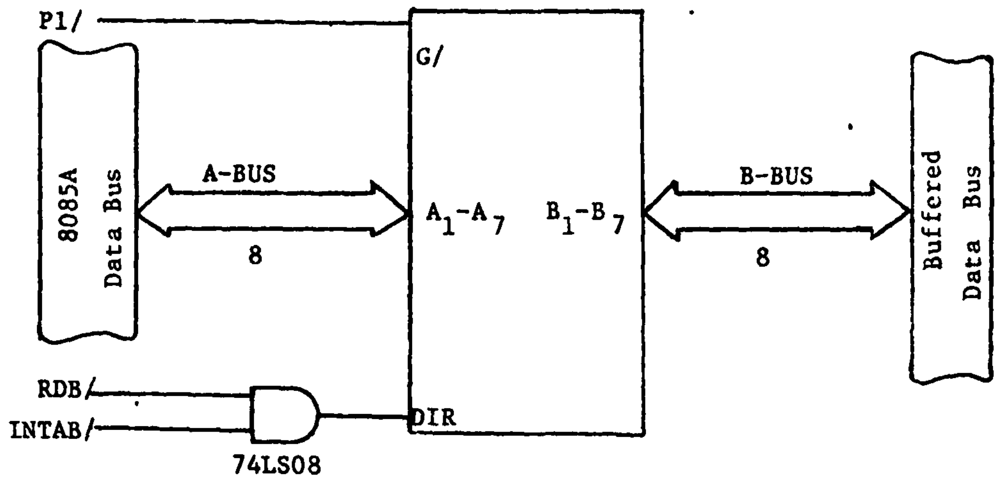


Figure 2-12. Data Bus Buffer

During read operation RDB/ is pulled down, or INTAB/ is pulled down (when interrupt on INTR input of the 8085A is acknowledged). These two signals are ANDed by a 74LS08 gate and the output is connnected to DIR input. Therefore DIR goes low when a read operation by the 8085A is required from the Buffered Data Bus.

Table 2-13 shows that the buffered Data Bus has an ample of driving capability with respect to the weakest device, the 8253.

DEVICE	I_{IH}(MA)	I_{IL}(MA)
8255A, (TWO), PPI	0.020	0.020
74LS245, Buffer	0.020	0.200
TOTAL LOAD	0.040	0.220
Nominal drive from the 8253	0.400	2.000
Reserve drive	0.360	1.780

Table 2-13. Static Loading for the Buffered Data Bus.

2.5.2 Dynamic Loading

The maximum allowable memory or I/O access time is the interval between the device address becoming available to the device and the output from that device becoming available to the Data Bus. AC characteristics including access time of integrated circuits are specified by the manufacurer at some particular capacitive load. If this load is exceeded, the system bus will operate too slowly and causes unreliable operation. In addition, propagation delays of all devices such as buffers which interface the microprocessor to the target device reduce further the device access time.

2.5.2.1 Capacitive Loading

When the 8085A is to be operated near its maximum operating frequency for the system designs the read and write cycle times become critical. Then the capacitive load, which is composed of wiring capacitance and component capacitances on the Data and Address Buses, should be considered. This consideration, rather than static load, will dictate the addition of buffers on the system bus.

2.5.2.1.1 Address Bus

The capacitive loads of the address input pins of all devices connected to the Address Bus must be added and compared with the rated capacitive load that can be driven by the Address Bus. For the 8085A, this specification is rated at a capacitive load of 150 pf. However, larger loads can be driven if an increase in the delay of the Address Bus's signals can be tolerated. Tables 2-14 and 2-15 provide a summary of capacitive loading calculation. A substantial reserve capacitive driving capability is indicated in both tables.

DEVICE	Max. C_{IN} (pF)
2142 (FOUR)	20
74LS154 (ONE)	10
2716 (TWO)	12
74LS244 (ONE)	10
Total Capacitance	52
Nominal Load 8085A	150
Reserve	98

Table 2-14. Capacitive loading calculation for the Upper 8-bit Address Bus.

DEVICE	Max. C_{IN} (pF)
2142 (FOUR)	20
2716 (TWO)	12
74LS244 (ONE)	10
Total Capacitance	42
Nominal Drive from 8212	150
Reserve	108

Table 2-15. Capacitive loading calculation for the Lower 8-bit Address Bus.

A similar consideration for the lower 8 bits of the Buffered Address Bus is provided in Table 2-16. The address buffer, 74LS244, is tested by Texas Instruments with a 45 pF capacitive load which produced a maximum propagation delay of 18 ns. The delay times increase at an average[1] of 0.08 ns/pF for larger values of capacitive load. Therefore, an increase of a 100 pF load will add an increase of 8 ns. delay. Each of Inte's peripheral devices requires a minimum access time of 400 ns. Since the 8085A allows a 590 ns. "Valid address to valid data in" time interval, a 190 ns. of time margin is available. The above additive time delays of (18 + 8) 26 ns., and the 30 ns. propagation delay produced by the lower 8 bits address latch 8212 (A2), are added to 56 ns. This is well within the 190 ns. design margin.

In conclusion, as seen from the above calculations, The Address Buffers 74LS244 are not needed in the present system. However, it is included for future system expansion.

DEVICE	Max. C_{IN} (pF)
8205 (ONE)	5
8255A (TWO)	20
8253 (ONE)	10
Total Capacitance	35
Nominal Drive Available from 74LS244	145
Reserve	110

Table 2-16. Capacitive loading calculation for the lower 8-bit Buffered Address Bus.

1. Calculated from Output Loading Performance Bulk Capacitance, "Low Power Bus Drivers and Tranceivers the LS240 Series Engineering Guide", by Texas Instruments Incorporated, 1977. pp. 23-24.

2.5.2.1.2 Data Bus

Similar considerations are carried out in Tables 2-17 and 2-18 for the Data Bus and the Buffered Data Bus. The capacitive loading is most critical on the output pins of memory components, and I/O components since their rated capacitive load is specified at 100 pF which is lower than that of the 8085A address/data pins. All bus drivers and transceivers of the 74LS240 series have similar electrical characteristics such as capacitive loading. With the previous justification, the 74LS245 outputs can drive a 145 pf load with a maximum of 26 ns. time delay.

DEVICE	Max. C (pF)
8085A (ONE)	10
8155 (ONE)	10
2142 (TWO)	10
8212 (ONE)	9
74LS245 (ONE)	10
2716 (TWO)	24
Total	73
Nominal load 2142	100
Reserve	27

Table 2-17. Capacitive loading calculation for the Data Bus.

DEVICE	Max. C (pF)
8255A(ONE)	20
8253 (ONE)	20
74LS245 (ONE)	12
Total	52
Nominal load 8255A	100
Reserve	48

Table 2-18. Capacitive loading calculation for the Buffered Data Bus.

Capacitive loadings are calculated and compared to the capacitive driving capability of the weakest device connected on the Data Bus. If the data bus buffer 74LS245 is removed, the 2142 would have a 128 pF capacitance load on its output pins, which exceeds its 100 pF limit. The only effect of additional capacitance is the slowing of the signal's rise time. But the 8085A provides a 590 ns. access time for the target device which exceeds the maximum access time required by the 2142 by 140 ns. This excess in timing design margin allows additional capacitance to be added, so long as the 140 ns. is not exceeded. An excess of 100 pF capacitance will impose an estimate of 25 ns. additional delay. Although not mandatory, the 74LS245 is provided to allow for future system expansion while at the same time it reduces the capacitive loading on the 8085A Data Bus.

Wiring capacitance per line on the bus might add an estimate of 14 pF to the total capacitance. This is based on the assumption that each integrated circuit added on the bus line is going to accumulate a 2 pF of wiring capacitance to the total. In each of the previous capacitive loading calculation's tables, the wiring capacitance is below the reserve driving capability of the Bus.

2.5.2.2 Propagation Delay:

Connecting buffers between the 8085A immediate system bus and additional memory and/or I/O devices isolates the static and capacitive loads from the bus system while adding 30 ns. to the apparent access times of components located beyond the buffers. Fortunately, the bus timing of the 8085A provides ample design margin compared to its predecessor the 8080A-1.

2.5.2.2.1 Read Operation

Figure 2-13 shows maximum available access timing with which the 8085A will function over the full environmental operating range. The parameters shown in Figure 2-13 are defined in Table 2-19 and are used to calculate the maximum allowable access time. The time delay for chip select decode logic is not included in the access time calculation. The decode logic circuits 74LS154, 74LS139 and 8205 will collectively introduce (19 + 33 +18) = 70 ns. maximum. However, the specification for typical memory or I/O component show that the chip select signal can be applied somewhat later than the address without introducing extra delays in the availability of data.

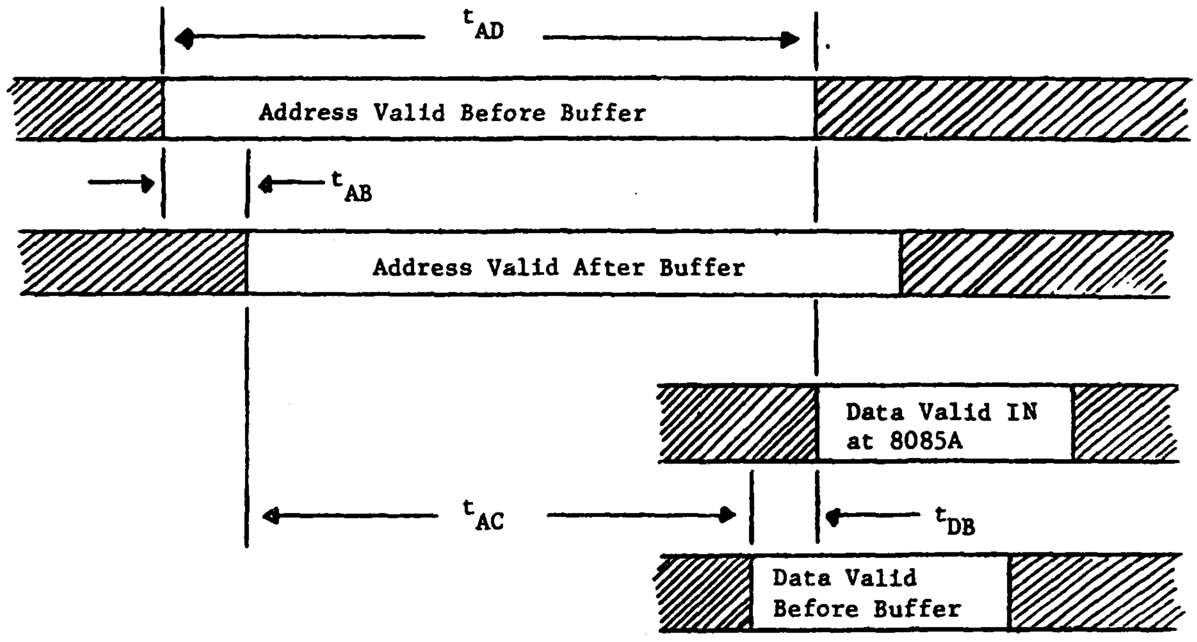


Figure 2-13. Read Cycle Timing

Parameter	Definition	Circuit	Maximum (ns.
t_{AD}	Valid address to Valid Data In = $\left(\frac{5}{2}\right) \times t_{CYC} - 225$	8085A	590
t_{AB}	Address Buffer Delay	8212 and 74LS244	60
t_{DB}	Data Buffer Delay	74LS245	30
t_{AC}	Maximum Allowable Access Time	Target Device	500
t_{CYC}	8085A Clock Period	8085A	326

Table 2-19. Definition of Parameters used in Figure 2-13.

The linear equation for calculating t_{AC} is derived from Figure 2-13 as follows: $t_{AC} = t_{AD} - t_{AB} - t_{DB} = 590 - 60 - 30 = 500$ ns.

Any device with 500ns. or less access time can be used in the buffered area. The 8255A and 8253 require a maximum access time of 250 ns. and 350 ns. respectively.

2.5.2.2.2 Write Operation

The write operation requires an address and the decoded chip select signal to the target device to be valid before the initiation of the write pulse. Also the data that is placed on the Data Bus must be valid before and after the end of the control pulse. A typical write timing for memory or I/O devices is shown in Figure 2-14. A summary of write timing parameters required by the 8253 and the 8255A, and the corresponding timing parameters provided by the 8085A are given in Table 2-20.

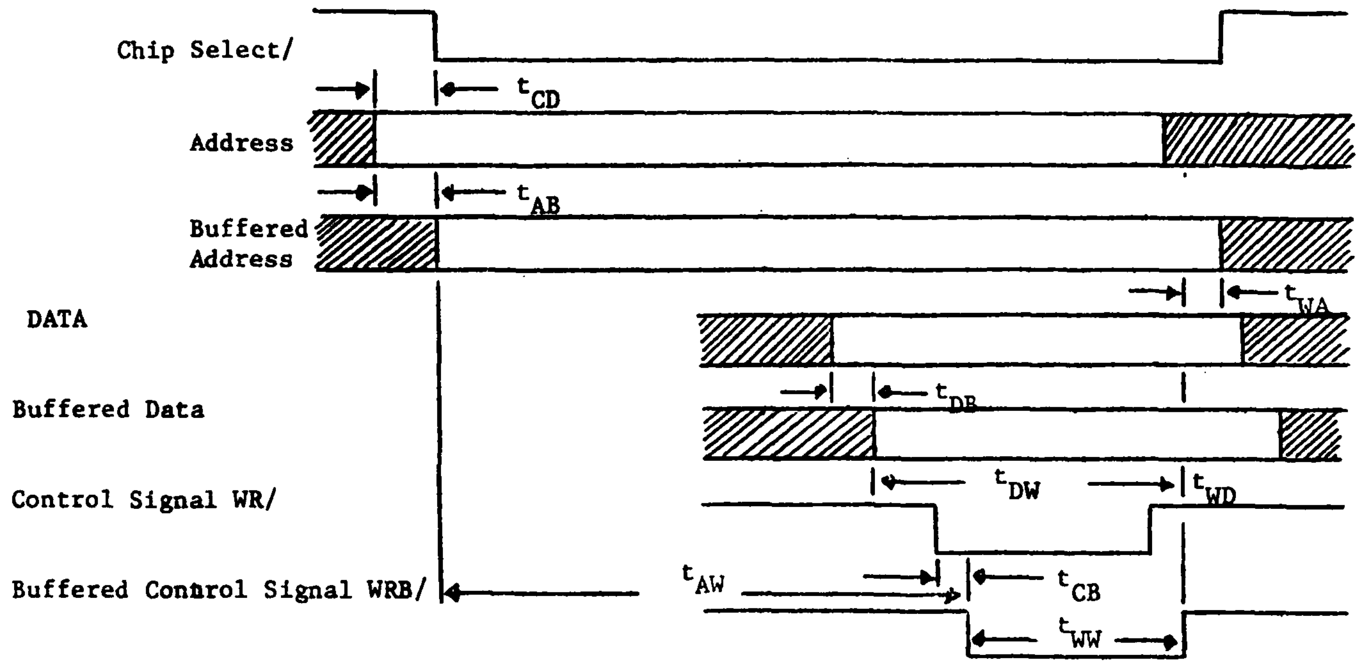


Figure 2-14. Write timing and delay parameters.

The critical timing delay is the length of time it takes to generate the address and decode the chip select signal to the target device. This delay must be completed before the write pulse is initiated. An analysis of the timing parameters will indicate that there is sufficient time for the address to become stable before the write pulses are applied. The equation for calculating the address set up time at the device input is given by: $t_{AWB} = t_{AW}$ (provided by the 8085A) $- t_{AB} + t_{CD} = 270 - 60 + 30 = 240$ ns.

Parameters	Definition	Required By		Provided by 8085A	Buffer Delay
		8253	8255A		
t_{AB}	Address Buffer Delay	--	---	---	60
t_{CD}	Chip Select Delay	--	---	---	70
t_{DB}	Data Buffer Delay	--	---	---	30
t_{CB}	Control Buffer Delay	--	---	---	30
t_{AW}	Address Stable Before Write	50	0	270	
t_{WA}	Address Stable After Write	30	20	120	
t_{DW}	Data Valid Before Write	300	100	420	
t_{WD}	Data Valid After Write	40	30	100	
t_{WW}	Write Pulse Width	400	400	400	

Table 2-20. Write Timing Parameters.

This and the rest of the AC parameters provided by the 8085A are larger than the required parameters by Intel's 8080A microporcessor peripheral devices and all compatible memory devices. However, for a better reliable system, the 8085A peripheral devices, differentiated from the 8080A peripheral devices by the suffix (-5), should be used instead. The 8085A peripheral devices have improved AC characteristics. For example, the 8253-5 requires a write pulse width of 300 ns. instead of 400 ns. required by the 8253.

2.6 I/O Devices

As shown in Figure 2-15, four of Intel's peripherals (8155, 8253, and two 8255A's) are used to interface the processor section (the 8085A and memory) to the analog driver of the Quadrupole Mass Filter (QMF). Structurely configured

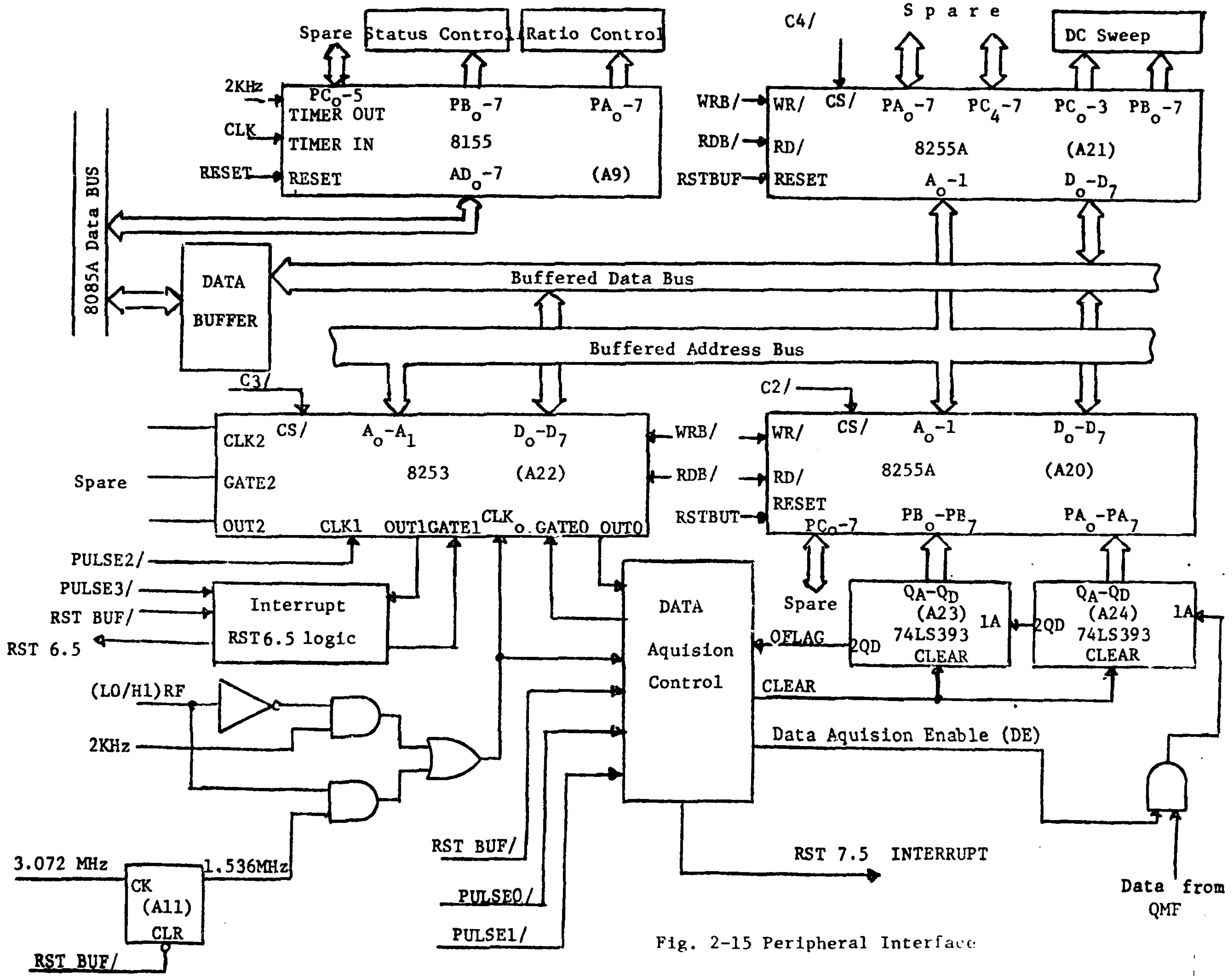


Fig. 2-15 Peripheral Interface

as memory mapped I/O, each device is connected to the buffered bus system, except the 8155 which is connected to the 8085A Data Bus and Control Bus. The chip select signal CE/ on each device is used to enable communication between the selected peripheral and the 8085A CPU. The direction of data transfer is controlled by the activation of RD/ or WR/ signals.

2.6.1 <u>I/O Ports</u>

The I/O section of the 8155 or the 8255A consists of four separate registers connected to an internal bus which is managed by an internal logic section. These registers--ports A, B, C and the Control Word Register-- are selected by the two port select signals (A_o,A_1). These signals, in conjunction with the RD/ WR/ and CS/ control the selection of one of the four registers. A summary of the control operation is given in Tables 2-21 (b) and (c). When ports A,B, or C is specified, the operation is an I/O port data transfer. The internal logic of the device will select the specified I/O port and perform the data transfer between the I/O port and the CPU.

The functional configuration of each port is controlled by the system's software. When the control word register is selected, the internal logic performs the operation described by the control word. The modes of operation for ports A, B and C are separately selected by an 8-bit control word. During system initialization, the PCU-85 software monitor programs each I/O port to be either an input or an output depending on the control word for that device, and the interfacing function assigned to it. Table 2-22 summarizes these functional assignments and the associated device control word in hexadecimal system. The I/O sections of A9 and A21 are programmed as latched output interface to the QMF by loading the command registers with the control words CF and 80 respectively.

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A_1	A_0	Operation
0	1	0	0	0	Load Counter No. 0
0	1	0	0	1	Load Counter No. 1
0	1	0	1	0	Load Counter No. 2
0	1	0	1	1	Write Mode Word
0	0	1	0	0	Read Counter No. 0
0	0	1	0	1	Read Counter No. 1
0	0	1	1	0	Read Counter No. 2
0	0	1	1	1	No-Operation 3-State
1	X	X	X	X	Disable 3-State
0	1	1	X	X	No-Operation 3-State

(a) 8253 Basic Operation

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	Operation
0	0	0	1	0	Read Port A
0	1	0	1	0	Read Port B
1	0	0	1	0	Read Port C
0	0	1	0	0	Load Port A
0	1	1	0	0	Load Port B
1	0	1	0	0	Load Port C
1	1	1	0	0	Load Port Control Reg.
X	X	X	X	1	Disable 3-State
1	1	0	1	0	Illegal Condition
X	X	1	1	0	No-Operation 3-State

(b) 8255A Basic Operation

IO/M = 1, I/O Operation.						
CS/	RD/	WR/	A2	A1	A0	Operation
0	1	0	0	0	0	Load Command/Status Register
0	1	0	0	0	1	Load Port A
0	1	0	0	1	0	Load Port B
0	1	0	0	1	1	Load Port C
0	1	0	1	0	0	Load Low-order 8-bit of timer count
0	1	0	1	0	1	Load high 6-bits of timer count and 2-bits of timer mode
0	0	1	0	0	0	Read Status Register
0	0	1	0	0	1	Read Port A
0	0	1	0	1	0	Read Port B
0	0	1	0	1	1	Read Port C
0	0	1	1	0	0	Read low-order bits of timer count
0	0	1	1	0	1	Read high-order bits of timer
1	X	X	X	X	X	Disable 3-state
0	1	1	X	X	X	No operation

(c) 8155 I/O Port and Timer Basic Operation.

Table 2-21, I/O ports and timers addressing and operational modes with respect to the Data Bus.

DEVICE	PORTS	CONFIGURATION	CONTROL WORD (HEX)	FUNCTION IN CONJUNCTION WITH THE QMF
8155A (A9)	A1	Output		Ratio Control (8 bits)
	B1	Output		Status Control (8 bits)
	C1	Output		Spare (6 bits)
	CWR		CF	
8255A (A20)	A2	Input		Lower 8 bits of Pulse Data Count
	B2	Input		Upper 8 bits of Pulse Data Count
	C2	Output		Spare (8 bits)
	CR		92	
8255A (A21)	A3	Output		Spare (8 bits)
	B3	Output		Lower 8 bits of Voltage Sweep Control
	1/2 C3	Output		Upper 4 bits of Voltage Sweep Control
	1/2 C3	Output		Spare (4 bits)
	CR		80	

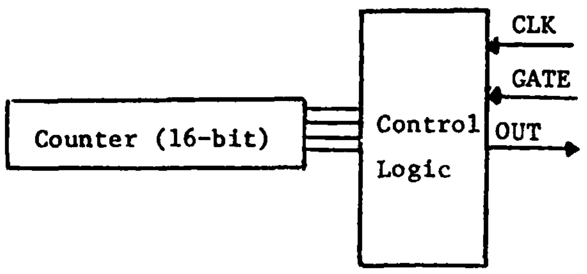
Table 2-22. I/O Ports Functional Configuration and Control Word Assignment.

Output pulses from the QMF are accumulated in a 16-bit counter. The counter is composed of two catenated dual negative-edge-triggered 4-bit binary counters 74LS393, A23 and A24. The 16-bit output lines of the counter are connected to ports A and B of the 8255A (A20). The sampling of each word count is controlled by the software monitor via the data acquisition control. Prior to each sampling, the counter is cleared by pulsing the CLEAR line high. Then the data pulses are gated to the input of the counter (1A) by pulling the Data Acquisition Enable (DE) signal high. At any time, the 16-bit word count can be read by the microprocessor through ports A and B of A20.

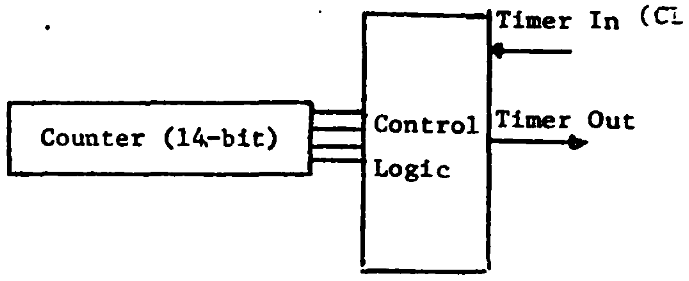
To prevent overflow of the word count, the most significant bit of the counter is used as an overflow flag (OFLAG). Whenever the OFLAG goes high indicating that the counter is half full it pulls the DE signal down via the Data Acquisition Control to inhibit pulse counting. At the same time, RST7.5 interrupt line is pulsed high, and the 8085A responds by reading and storing the word count. Then the sampling process is repeated under the control of the software.

2.6.2 Programmable Interval Timers/Counters

PCU-85 includes four timers/counters composed of a 14-bit down-counter in 8155 (A9), and three independent 16-bit down-counters in Intel's programmable interval timer 8253 (A22). Each timer/counter is functionally configured via software to implement one mode of operation such as an event counter, a rate generator or a one-shot. Under the control of the software monitor, the 8085A sends out a set of control words to initialize each counter with the desired mode and quantity information. Once initiated to perform its assigned timing task, each timer counts its input count pulses and provides either a square wave or pulse when terminal count (TC) is reached. The initiation of counting with an 8155 timer is accomplished via software only, while the counting of each 8253 counter is initiated by software and/or hardware. Figure 2-16 shows an added control line, GATE, to an 8253 counter compared to the 8155 counter. For most of the modes of operations, counting is disabled/enabled by the logic low/high on the GATE line. For the rest of the modes, counting is initiated by the rising edge of the GATE line.



(a) An 8253 Counter.



(b) The 8155 Counter.

Figure 2-16. 8155 and 8253 typical counters

Table 2-23 summarizes the functional assignment to each timer and its mode of operation as defined by Intel's data sheets on the 8155 and 8253 chips. With reference to Figure 2-15, it is shown that the clock input to counter 0 and the data acquisition control can be selected to be either a 2kHz clock or 1.536MHz clock by one of the control lines from the mode control lines defined as (LO/HI) RF line. The (LO/HI) RF signal when it is pulled high selects the 1.536MHz and selects the 2kHz clock when it is pulled down. The 2kHz clock is generated using the 8155 timer by dividing the system clock by the count 1536. The 1.536MHz is derived from the system clock by using a D-flip-flop 74LS74 (A11) configured as a divide-by-2 counter as shown in Figure 2-15.

DEVICE	COUNTER	OPERATIONAL MODE	FUNCTION
8155 (A9)	14-bit counter	Square-wave output	divides the input clock of 3.072 MHz and produces an output of 2kHz.
8253 (A21)	Counter 0 (16-bit)	Mode 4	programmed as an event timer that measures the sampling time window.
	Counter 1 (16-bit)	Mode 5	programmed as an event counter that counts the number of samples collected.
	Counter 2 (16-bit)	------	Spare

Table 2-23. PCU-85 TIMERS/COUNTERS Functional Organization.

Communication between the 8155 timer and the CPU is similar to the 8155 I/O ports. The lower 3 bits of the Address Bus are used to select the timer, in addition to the control signals WR/, RD/ and CS/. The command/status register is used for the timer as well as for the I/O ports.

The 8253 is internally structured as three independent 16-bit counters and a control word register. Each counter consists of a single pre-settable, down-counter with logic control for independent operation. Counting in either binary or BCD, the timing input gate and output are configured by the information to be stored in the control word register. There are five operational modes--defined in the 8253 data sheet--that each counter can be programmed to perform for the required timing tasks.

Addressing of each timer using Address and Control Buses is summarized in Tables 2-21(a) and (c).

2.7 Data Acquisition Control

Although initiated by software control, data sampling is accomplished via hardware control. This control unit is shown in Figure 2-17.

Time window count is loaded in COUNTERO of the interval timer, the 8253. Then, the software activates PULSEO/which pulses the CLEAR line high through inverter gate number 6. The CLEAR line resets the data counter A23 and A24 in Figure 2-15. When PULSE1/ is pulsed low by software, it sets flip-flop F1. Thus pulling the data enables DE line high. This allows data pulses to pass through gate 5 to the input of the data counter to be counted.

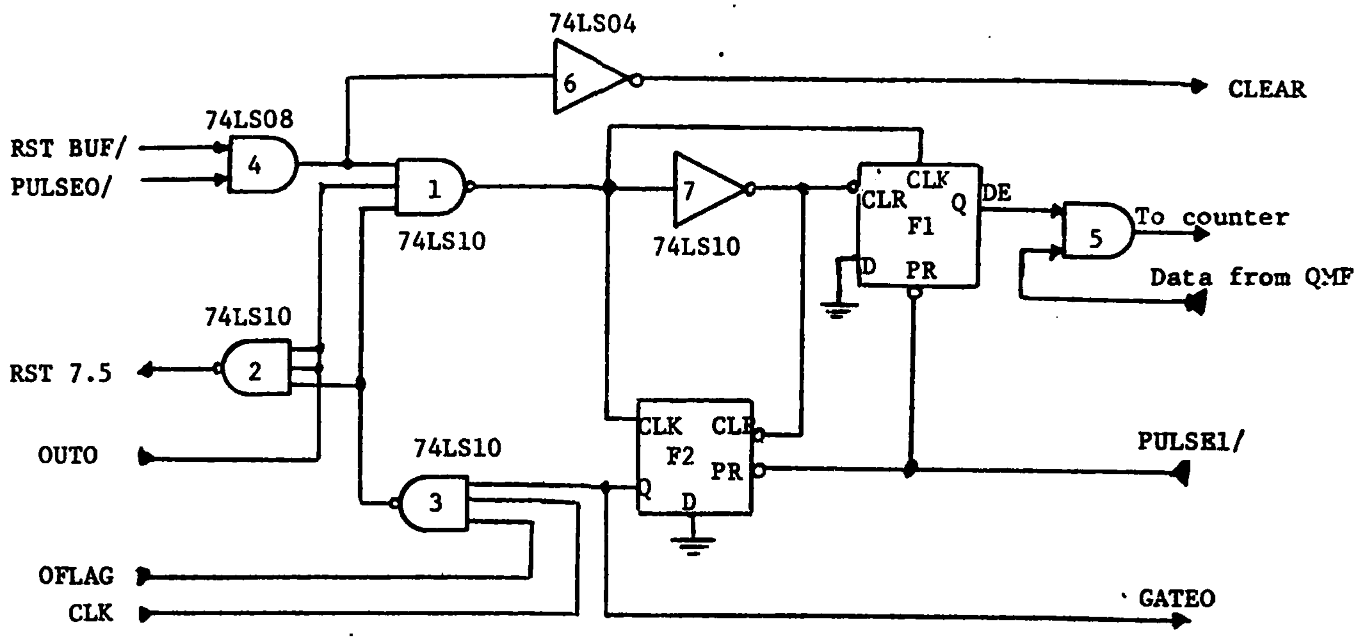


Figure 2-17. Data Acquision Control

At the same instant PULSE1/ sets flip-flop F2. The output Q of F2, being connected to GATEO of COUNTERO, enables COUNTERO to decrement the count it holds in its register at the rate of its clock input CLKO. On terminal count, the output OUTO is pulsed low for one input clock period. This activates the three inputs NAND gates numbers 1 and 2 by pulsing their respective outputs high. NAND gate 1 clears flip-flops F1 and F2 via inverter gate 7. Therefore F1 pulls DE line low and stops any more data pulses to pass through gate 5. Also, F2 inhibits COUNTERO from counting by pulling GATEO input low. NAND gate 2 activates interrupt line RST 7.5 requesting the 8085A service.

If the data count becomes half full before the time window count is decremented to zero, the overflow flag OFLAG goes high. Syncronized by the clock CLK, the output of NAND 3 is pulsed low for a period of one positive-edge clock pulse. Thus activating gates 1 and 2 which in turn inhibit dat pulse counting and time window counting, and activates the RST 7.5 interrupt line.

On power-up, RST BUF/ goes low and resets the data acquisition control.

2.8 RST 6.5

COUNTER1 contains a count which is proportional to the size of the memory buffer. Each data sampling requires four bytes of storage - two bytes for data count, and the other two bytes for the time window count. Since the size of the present memory buffer is 512 bytes, then the count to be loaded in COUNTER1 is equal to 128. But, due to internal design of the 8253 the counter operating in mode 5 counts two more extra clock pulses. Therefore, the actual count, defined as memory buffer count, to be loaded in COUNTER; is equal to 126.

PULSE2/, being used as a clock input to COUNTER1, is used to decrement the memory buffer count after each data sampling. Therefore, whenever RST 7.5 interrupt routine is executed, the monitor pulses PULSE2/ low. When terminal count is reached, OUT1 will go low for one clock period. Being connected to PR input of flip flop F3 in Figure 2-18, OUT1 will set the Q output and reset the Q/ output of F3. RST 6.5 interrupt is activated by level high of the output Q. The 8085A responds [illegible]vicing RST 6.5 interrupt routine. The routine will activate PULSE3/ which will reset F3. Therefore, RST 6.5 goes low and at the same time GATE1 will go high. The rising edge of GATE1 will retrigger COUNTER1 by reloading it with the same memory buffer count, and re-initiating the counting procedure.

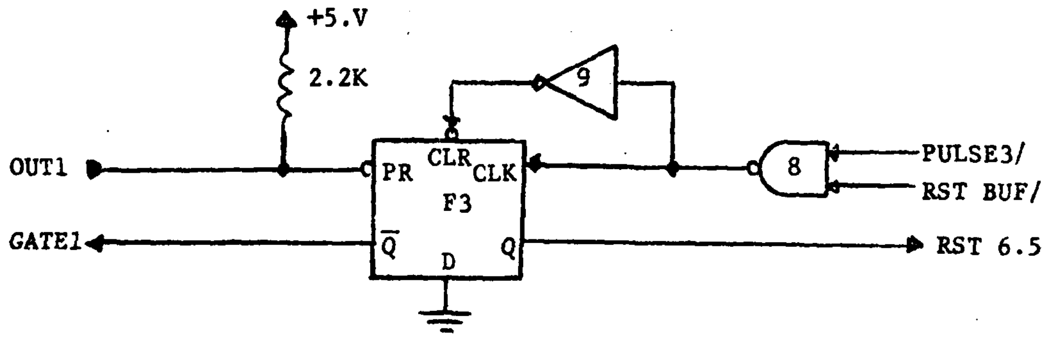


Figure 2-18. RST 6.5 Interrupt Control Logic.

Similar to all circuits in PCU-85 system, RST 6.5 interrup control logic can be reset by activating RST BUF/.

CHAPTER 3
SOFTWARE DESIGN

The PCU-85 software system performs the actual tasks required upon the quadrupole mass spectrometer filter, be it a control function or data acquisition. In the first section, these tasks and their operating parameters are described in some detail. Then, the remainder of the chapter discusses the following items:

1. The PCU-85 software structure used to implement the operational tasks of the QMF.
2. Software analysis and design using Intel's 8085A Assembly Mnemonics.

3.1 Control Operation

With reference to Figure 1-1 and 1-2, a typical voltage sweep to be generated is shown in Figure 3-1. This voltage sweep and the ratio control are used to amplitude modulate the rf signal, the result is applied to the mass spectrometer rods.

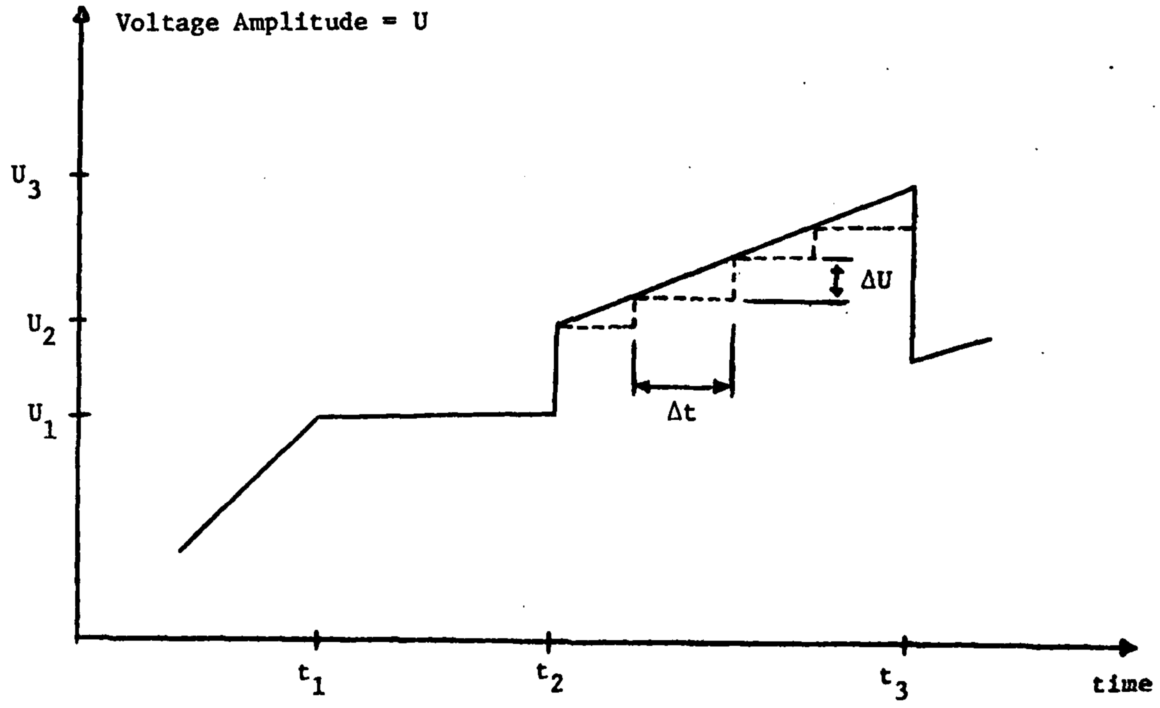


Figure 3-1. Voltage Sweep Versus Time.

As seen from the graph, there are three modes of operations;

(1) Total Ion Mode

In this mode, the QMF becomes a high-pass filter where only ions with mass greater than the cutoff value M_0 pass through the quadrupole. During this mode, the voltage sweep is usually held constant at a voltage U_1 in a time interval $t_2 - t_1$. In addition, the DC excitation voltage is reduced to zero. In some cases, the voltage sweep is allowed to vary and accordingly the lower cutoff mass M_0 is varied with the corresponding ramp voltage.

(2) Jump Mode

A jump in the voltage sweep from U_1 to U_2 at time C_2 allows the QMF to skip scanning masses that are selectable by the voltages that lie between U_1 and U_2.

(3) Scan Mode

To scan masses of all ions lying in a mass domain M_2 and M_3 selectable by the voltages U_2 and U_3, it is required to generate a ramp sweep of slope equal to $(U_3 - U_2)/(t_3 - t_2)$. Simultaneously, the ratio U/V is kept constant to keep the filtered mass domain in the stability region of the QMF.

The ramp is generated using step-wise approximation of amplitude ΔU in a time interval Δt. The start of each step is called a steppoint.

The above operations are separated by breakpoints. In Figure 3-1, these points are located at t_1, t_2 and t_3. At these breakpoints, the ration control, status control and sweep control are updated according to predetermined binary words, defined as control parameters, that are stored in memory. The present system uses Intel's 2716 EPROM as storage elements for these parameters.

3.2 Data Acquisition

At the onset of each breakpoint and each steppoint, just after updating the control parameters, the pulse counter in Figure 2-15 is commanded to start counting the incoming data pulses. At the same instant, the time interval window count (Δt) in COUNTERO is allowed to decrement. When overflow of the pulse count occurs, or when the time interval Δt is decremented to zero, pulse counting is automatically terminated. The pulse count and the actual time interval count are transferred into buffer memory for temporary storage. Then the next breakpoint or steppoint is initiated.

Since the pulse counter is 16-bit wide, then for a maximum data pulse rate of 3MHz, the minimum expected window time Δt, during which half of the maximum count is accumulated, is equal to $2^{15} \times (10^{-6}/3) = 110$ msec. An upper limit for Δt is estimated to be a 100 sec, which corresponds to data pulse rate of 327Hz. COUNTERO is used to cover the range values of Δt, by decrementing the time interval count loaded in its register at an optimum clock rate generated at the output of the 8155 timer.

3.3 System Overview

The PCU-85 software structure can be described as a real-time multitasking system. It schedules, controls and responds to asynchronous events occuring concurrently. In contrast to sequential operation, this occurrence of events is a distinguishing characteristic of real-time systems. Synchronization and prioritization of events is accomplished through hardware interrupts assigned to each event, and through software programming.

The PCU-85 software is block-structured into modular routines. Each module, known as a task, is a separate routine assigned to an event. Tasks share resources such as memory devices and software subroutines. Therefore, information exchange is possible among tasks via these resources. Most of these tasks are event driven initiated by a hardware interrupt activation.

A flow diagram of the software system is given in Figure 3-2. Upon

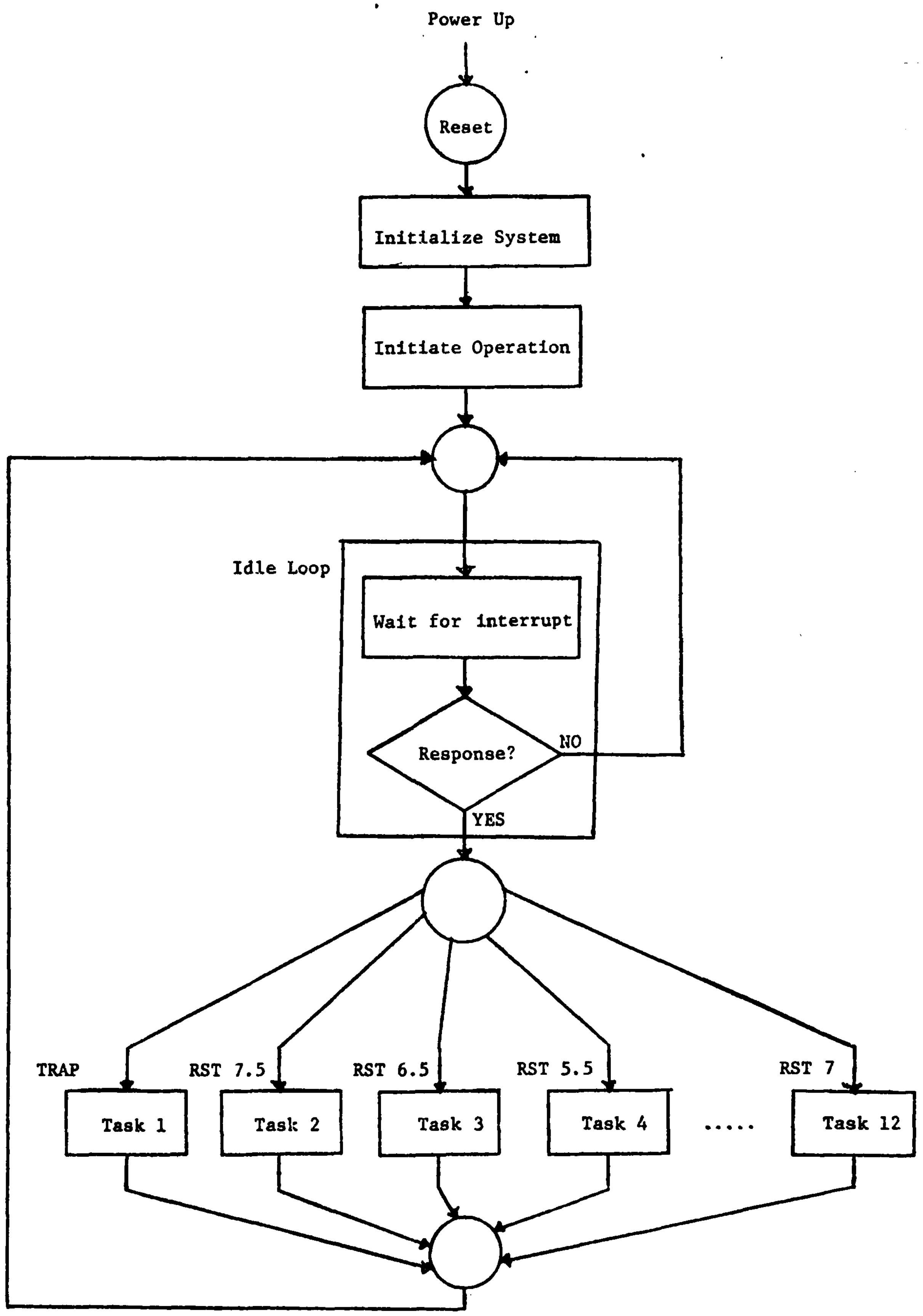


Figure 3-2. Flow Diagram of the PCU-85 Software System.

reset, the software and hardware systems are initialized to a predetermined state. After initiating control operation, the 8085A enters into a timed idle loop. In this state, the microprocessor executes the No Operation (NOP) instruction repeatedly until it receives a request on at least one of its interrupt lines. Accordingly, the corresponding task takes control of the processor when no other tasks of higher priority are ready to execute. This task performs some action upon the request and then simply resumes waiting via the idle loop until the next request is received. Task waiting period is scheduled according to COUNTERO time interval and/or the occurrence of an event.

The PCU-85 uses the following three tasks for its programmed operation:

1. Task 2

Task 2 is the executive routine that accomplishes the control operations described in sections 3.1 and 3.2. It initiates the function by sending control parameters to the QMF input ports and then surrenders control of the processor to the idle loop. At this time, one of the other tasks can execute its function when it is activated. Task 2 continues to wait until it receives a response from the QMF ports. Being the highest priority, it resumes control of the 8085A by storing the sampled data into buffer memory, and then it initiates the control functions of the next steppoint.

2. Task 3

Task 3 will be ready to execute whenever the memory buffer becomes full. If no higher priority task is currently running, Task 3 takes control of the processor. It arranges the data in memory into a file with appropriate headings and displays it on the CRT screen. Then, Task 3 completes its functions by adjusting memory pointers and the current breakpoint pointer, and by initiating control function of the QMF at the start of the interrupted breakpoint. This task continues to wait until the next response is activated. The second priority level is assigned to Task 3.

3. Task 4

The operator can interrupt the current control operations of the PCU-85 by activating Task 4. This executive allows the operator to adjust memory pointers or control parameters in the scratch pad. In addition, control operations of the current block of parameters can be altered to another block of parameters in the same set of blocks or repertoire. Task 4 enters the waiting state whenever one of its commands is executed. The third priority level of interrupts is assigned to Task 4.

The four restart interrupts can be increased up to 12 interrupt levels by using the vectored interrupt line INT and a single AM 9519 Universal Interrupt controller device manufactured by Advanced Micro Devices. The AM 9519 manages the masking, priority resolution and vectoring up to eight interrupts through its eight interrupt request lines. Each of the added eight hardware interrupt levels has a set of priorities, of which one must be assigned to the task that services the interrupt. The highest priority interrupt level, TRAP, is assigned to Task 1. Task 1 is reserved for servicing catastrophic errors such as power failure or bus error.

3.4 Memory Organization

A map of memory addresses with their corresponding functional blocks that are utilized by the PCU-85 software monitor is shown in Figure 3-3. The memory map is divided into two parts. The first part of the map is occupied by EPROMs for monitor program and control parameters storage. RAM for scratch-pad and data accumulation occupies the second part, with memory-mapped I/O section in between. Unassigned sections are left for memory expansion to be occupied by EPROMs and/or RAM as needed. The structure of each utilized memory block is discussed in this section.

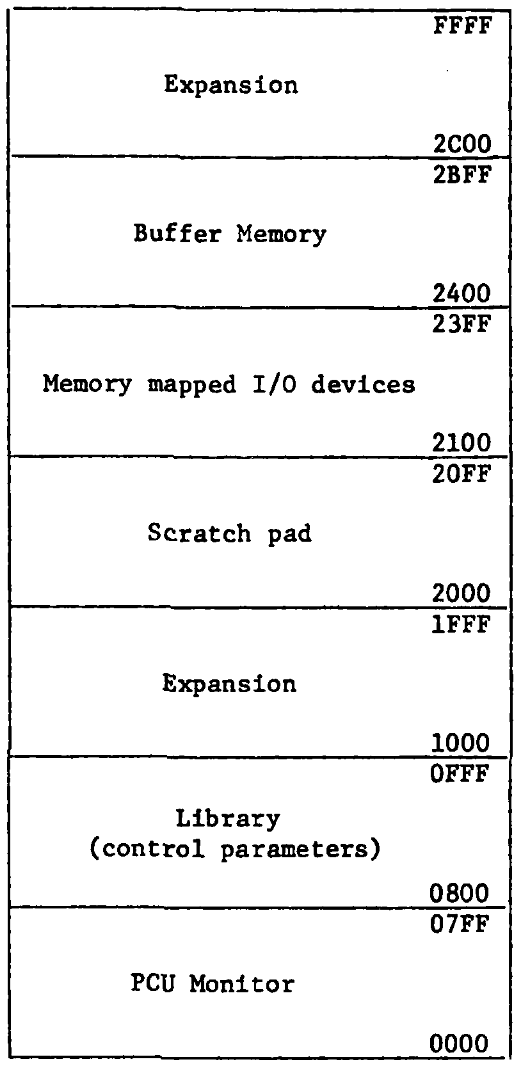


Figure 3-3. PCU-85 Memory Map.

3.4.1. Monitor: The general data structure of the PCU-85 monitor is discussed in the previous section 3.3.

3.4.2. Library: A Library is a collection of pointers and control parameters organized in an orderly structure. Each pointer is a 16-bit address that belongs to one of four hierarchy levels of organization. These levels are:

1. Archive: an archive is a collection of repertoire pointers (REP) that must be stored in consecutive memory locations. Each repertoire pointer REP (i) identifies a specific block of pointers.

2. Repertoire: a repertoire is a collection of program pointers (PROG) that are stored in consecutive memory locations. Each program pointer PROG (j) identifies a specific block of pointers.

3. Program: a program is a collection of KBLOK pointers that are stored in consecutive memory locations. Each KBLOK pointer KBLOCK (L) identifies a specific block of control parameters.

4. KBLOK: a KBLOK is a collection of K numbers or control parameters that are stored in consecutive memory locations. These parameters identify a breakpoint with the associated mode and status operation of the QMF and the sampling time interval for data pulse counting.

Each hierarchy of pointers can be of arbitrary length, arbitrary sequence, and arbitrary location in the memory space. Two bytes of zeros at the end of each of the first three hierarchies marks the end of list of pointers for that hierarchy. When this condition is detected, the monitor starts processing with a pointer from the next higher hierarchy level. If the end of the archive is reached, the monitor will repeat processing starting with the first pointer in the archive. A typical block structure of these blocks and interconnection is illustrated in Figure 3-4.

Hierarchy of pointers follows a tree-type of data structure. The branches of this tree are pointers, and its roots are control parameters. Figure 3-6 shows a simple library configured in a tree structure.

3.4.3. Scratch-pad: Scratch pad is an area of memory where the microprocessor stores intermediate computations, current pointers, stack, current breakpoint control parameters, and other relevant information. Figure 3-5

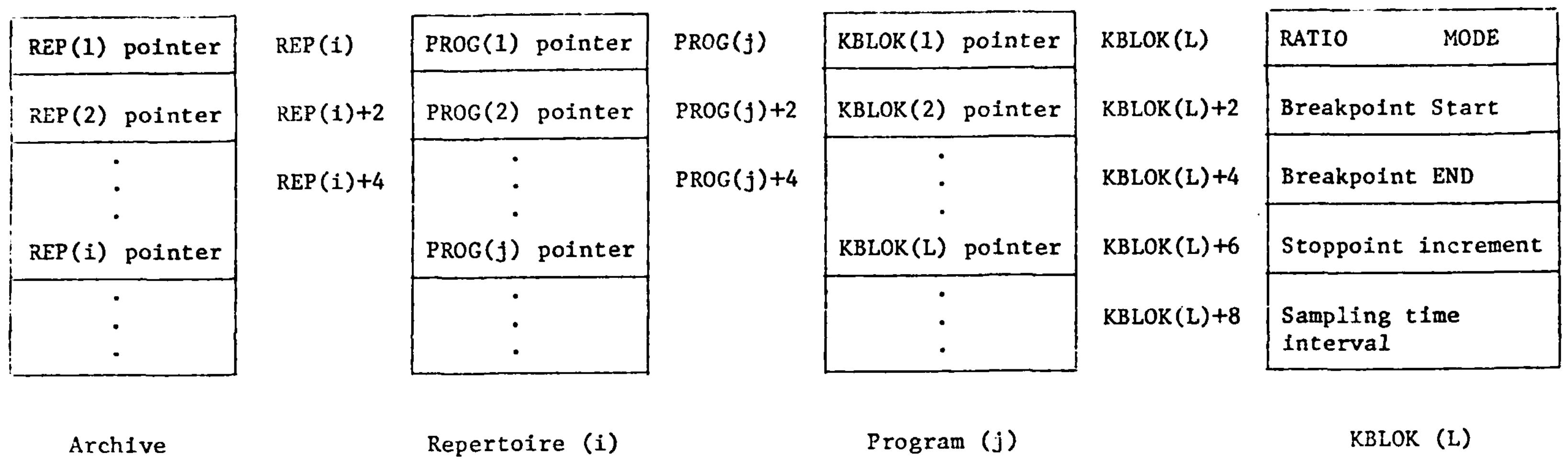


Figure 3-4. Hierarchy of Block Pointers.

current 'K' block storage	2000	Ratio
	2001	Mode
	2002	BKPOINT (AMU) Start Low Byte
	2003	BKPOINT (AMU) Start High Byte
	2004	BKPOINT (AMU) End Low Byte
	2005	BKPOINT (AMU) End High Byte
	2006	BKPOINT (AMU) Increment Low Byte
	2007	BKPOINT (AMU) Increment High Byte
	2008	Sampling Time Interval Low Byte
	2009	Sampling Time Interval High Byte
	200A	
	200B	
	200C	
	200D	
	200E	
	200F	
		Stack
	20A1	

Save locations for CPU register	20B1	E Register	13 bytes reserved for CPU registers
	20B2	D Register	
	20B3	C Register	
	20B4	B Register	
	20B5	FLAGS	
	20B6	A Register	
	20B7	L Register	
	20B8	H Register	
	20B9	Interrupt Mask	
	20BA	PC Low Byte	
	20BB	PC High Byte	
	20BC	SP Low Byte	
	20BD	SP High Byte	
	20BE	TEMP Low Byte	
	20BF	TEMP High Byte	
		⋮	
	20DF		
	20E0		
	20E1		
	20E2		64 bytes of current storage of data or address for the monitor
	20E3		
	20E4	⋮	
	20E5		
	20E6		
	20E7		
	20E8		
	20E9		
	20EA		
	20EB		
	20EC		
	20ED		
	20EE		
	20EF		
	20F0	File Number	
	20F1		
	20F2		
	20F3		
	20F4		
	20F5		
	20F6	BKPOINT end buffer storage pointer low	
	20F7	BKPOINT end buffer storage pointer high	
	20F8	current buffer storage pointer low	
	20F9	current buffer storage pointer high	
	20FA	current "K" block pointer low	
	20FB	current "K" block pointer high	
	20FC	current program pointer low	
	20FD	current program pointer high	
	20FE	current repetoire pointer low	
	20FF	current repetoire pointer high	

Figure 3-5. Memory Map for the Scratch-Pad.

shows a memory map where 256 bytes of RAM is assigned to the scratch pad.

The first 10 bytes are used for storage of current breakpoint control parameters. Whenever a new breakpoint is to be initiated, control parameters are transferred from the library to the scratch pad. Any task can easily refer to these memory locations and use the parameters for its assigned function.

The second part of the scratch pad is assigned to the CPU stack. Information, such as the content of the processor's registers and the program counter, is pushed down in the stack on a Last-In-First-Out (LIFO) basis. The stack is usually used for subroutine linkage, interrupts and temporary data storage.

The last part of the scratch pad is reserved for current pointer storage or temporary data storage. The topmost locations are used to store current repertoire, program and KBLOK pointers. The next two pointers are associated with buffer memory. The first one is buffer storage pointer which indicates the first empty location in buffer memory. The second one is breakpoint end buffer pointer, which the address of the last stored data from the previous breakpoint. If the buffer memory becomes full before the current breakpoint is completed, then data transferred from the buffer memory to the CRT will be up to and including the data addressed by the breakpoint end buffer pointer disregarding the data collected in the present breakpoint. When the transfer is completed, control operation will be continued from the beginning of the current breakpoint.

Memory space is provided for future expansion of any part in the scratch pad.

3.4.4. Buffer Memory

Buffer memory is used to compensate for a difference in the rate of

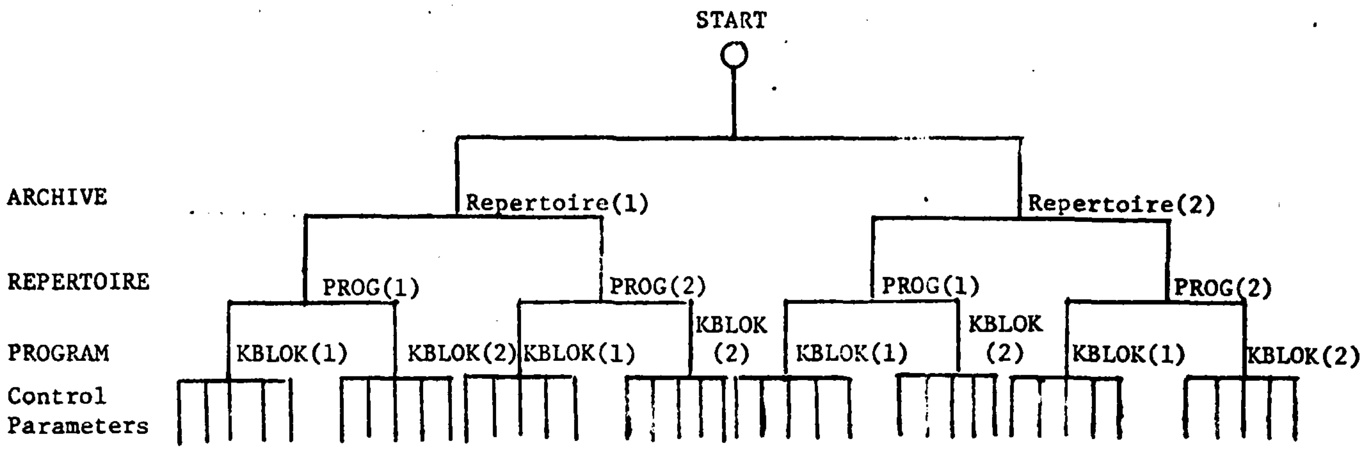


Figure 3-6. A Tree Structure for a Simple Library.

flow of information or the time occurrence of events when transferring information from one device to another.

Data, composed of pulse count and actual time interval count, are temporarily assembled in buffer memory everytime Task 2 is activated. Transfer of the buffered data from memory to another device such as the CRT screen occurs whenever a sufficient number of data is collected.

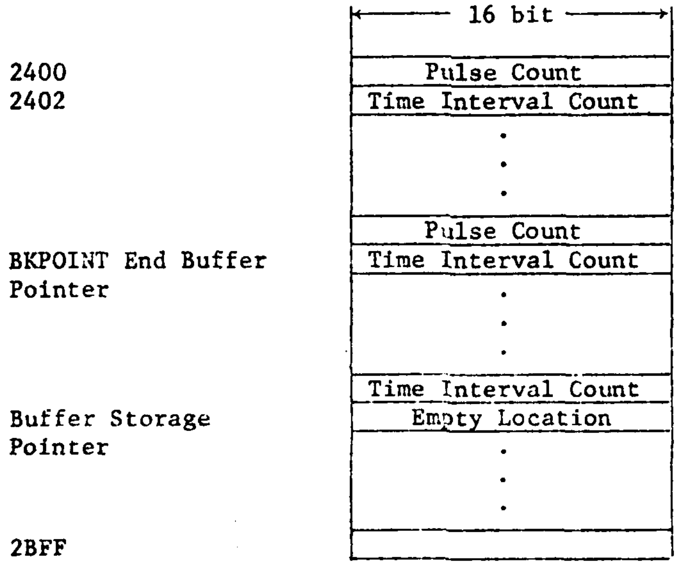


Figure 3-7. Memory Map for Buffer Memory.

3.5 Initialization

Upon power-up or following each reset signal activation, the hardware and the software systems are initialized to a predetermined state. The initialization routine executes this task by performing the following functions:

1. Disables all interrupts and initializes the stack pointer.
2. Initializes system ports and counters to the desired state required by the design and dictated by an initialization table. The table contains control words that control the operational mode of peripheral devices.
3. Initializes pointers and scratch pad to the organization described in the previous section.
4. Starts the first control operation of the QMF, enables the interrupts system and then enters the idle loop.

Due to the structure of the 8085A, the lower portion of memory from 0 to 38 hexadecimal are reserved for interrupt vectoring and software restart entry points. A branch instruction is written in each one of these entry points, to direct program execution to the associated service routines that are found in higher portion of memory. Restart zero (RST 0) is used as the entry point to the initialization routine. After initialization of the stack pointer, a branch instruction is inserted for program execution to jump to the main body of the initialization routine (INITZ). Figure 3-8 shows a flowchart of the logic operation of the initialization routine. The assembly listing of the routine can be found in Appendix A.

3.6 Task 2 (RST. 7.5 Interrupt)

3.6.1. Data Fetch

Whenever RST 7.5 interrupt signal is activated, Task 2 routine is invoked to commence processing. Execution starts with disabling all future and pending interrupts as shown in the flowchart diagram of Figure 3-10.

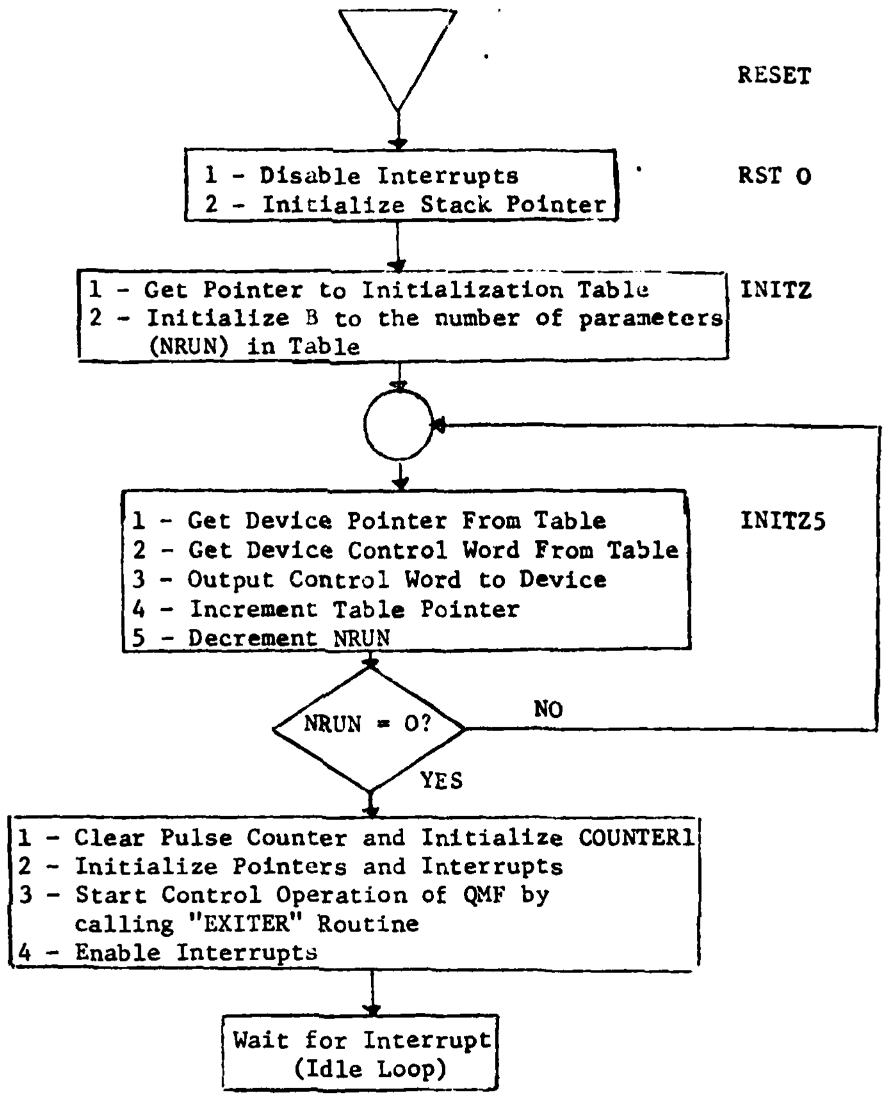

RESET
1 - Disable Interrupts
2 - Initialize Stack Pointer
RST 0
1 - Get Pointer to Initialization Table
2 - Initialize B to the number of parameters (NRUN) in Table
INITZ
1 - Get Device Pointer From Table
2 - Get Device Control Word From Table
3 - Output Control Word to Device
4 - Increment Table Pointer
5 - Decrement NRUN
INITZ5
NRUN = 0?
NO
YES
1 - Clear Pulse Counter and Initialize COUNTER1
2 - Initialize Pointers and Interrupts
3 - Start Control Operation of QMF by calling "EXITER" Routine
4 - Enable Interrupts
Wait for Interrupt (Idle Loop)

Figure 3-8. Flow-Chart of Initialization Routine.

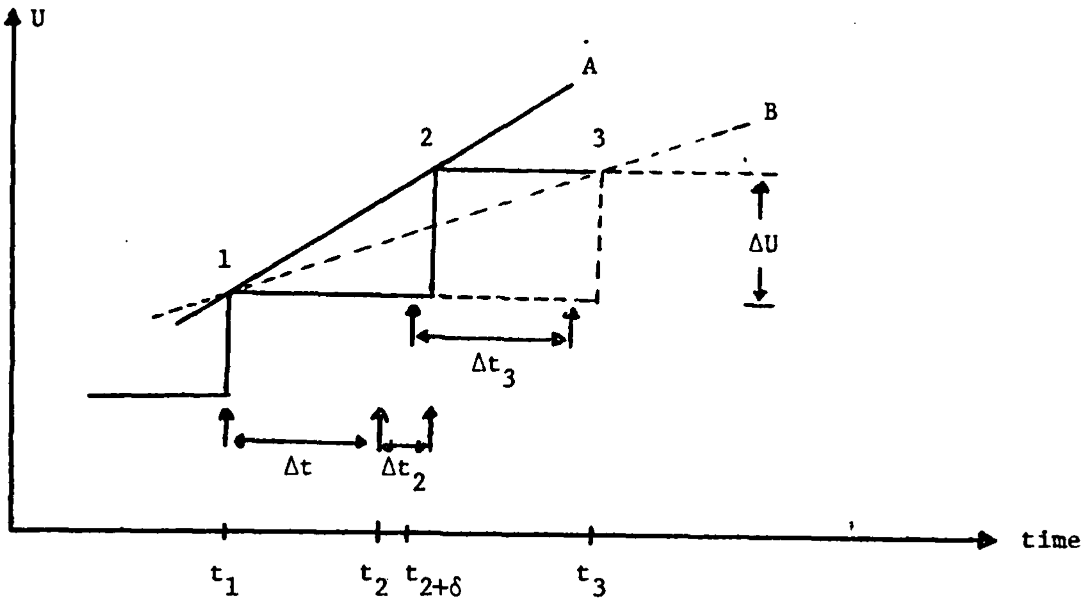


Figure 3-9. Voltage Sweep Slope Error

This is essential for the proper control of the QMF operation. Any interruption of Task 2 will add time delay to the execution time of Task 2 routine. In turn, the time invertal Δt will be increased and thus decreasing the slope of the voltage sweep and causing different masses to be scanned instead of those desired by the programmer.

Figure 3-9 shows an example of an error introduced to the voltage sweep slope when interrupts are not disabled, and another interrupt has occurred during the execution of Task 2. RST 7.5 interrupt occurs at t_2, and its service routine requires Δt_2(μs) to complete the task assigned to it. An interrupt for another task might occur at time $t_2+\delta$ during Δt_2. Task 2 will be suspended and the other task takes over the control of the microprocessor and starts executing. Once it completes its functions after Δt_3(μs) then Task 2 is allowed to resume execution to completion. The apparent time of execution of Task 2 is $(\Delta t_2 + \Delta t_3)$(μs). One of the control parameters that is updated at the end of Task 2 execution is voltage sweep

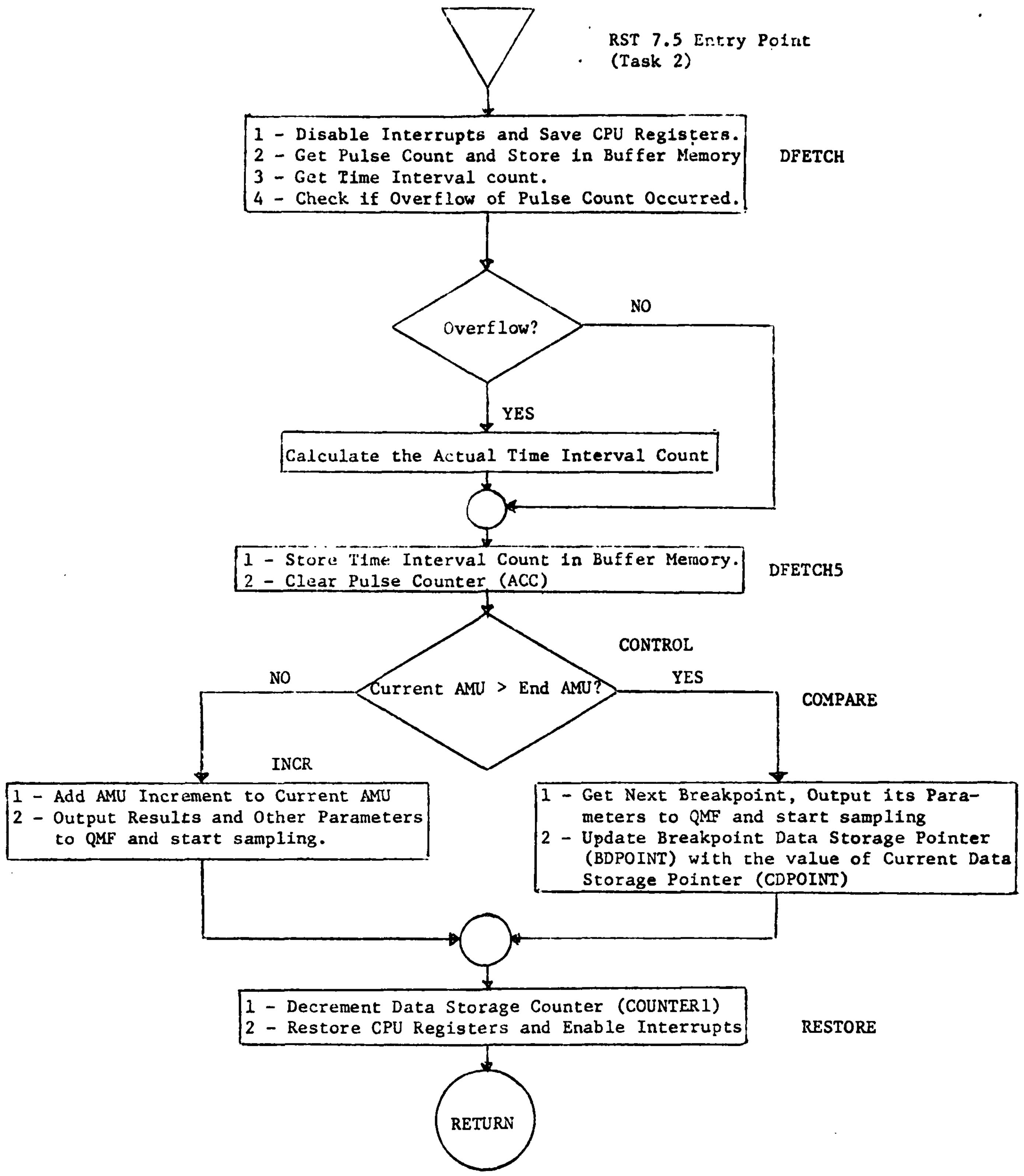

RST 7.5 Entry Point
(Task 2)
1 - Disable Interrupts and Save CPU Registers.
2 - Get Pulse Count and Store in Buffer Memory
3 - Get Time Interval count.
4 - Check if Overflow of Pulse Count Occurred.
DFETCH
Overflow?
NO
YES
Calculate the Actual Time Interval Count
1 - Store Time Interval Count in Buffer Memory.
2 - Clear Pulse Counter (ACC)
DFETCH5
CONTROL
Current AMU > End AMU?
NO
YES
COMPARE
INCR
1 - Add AMU Increment to Current AMU
2 - Output Results and Other Parameters to QMF and start sampling.
1 - Get Next Breakpoint, Output its Parameters to QMF and start sampling
2 - Update Breakpoint Data Storage Pointer (BDPOINT) with the value of Current Data Storage Pointer (CDPOINT)
1 - Decrement Data Storage Counter (COUNTER1)
2 - Restore CPU Registers and Enable Interrupts
RESTORE
RETURN

Figure 3-10. Flow-Chart of Task 2 (RST 7.5 Interrupt) routine

parameter. Because of Δt_3 delay time the voltage sweep is incremented at point 3 instead of its assigned place in time at point 2. As a result, Line B will be controlling the QMF instead of the calculated one Line A. To avoid this problem, one of the following solutions should be exercised:

1. Disable interrupts at the start of Task 2 routine and enable interrupts at the end of the routine.

2. If TRAP interrupt line is to be used to indicate power failure, all interrupts should be masked at the start of Task 2 routine and then unmask them at the end of the routine. Since TRAP is unmaskable interrupt, it is acknowledged when it is activated.

In this prototype, TRAP is not used, therefore disabling interrupts is the proper choice.

Because of the reasons given above, the highest priority interrupt used is assigned to Task 2. At the same time, the execution time Δt_2 of Task 2 is optimized to the smallest value possible. The maximum value of Δt_2, when a new breakpoint is initiated, is calculated to be 362.2 μs.

Task 2 continues operation by saving the 8085A registers, on the stack, and then fetches the data accumulated in the Pulse Counter and store it in buffer memory. The most significant bit of the pulse count is tested. If it is set, overflow has occurred before the time interval counter is decremented to zero. Task 2 subtracts the remainder of the time count from the interval time count found in the scratch pad and stores the result as the actual time count into buffer memory. If the most significant bit of the pulse count is found to be zero, then no overflow has occurred, and Task 2 stores the time inverval count in buffer memory with no adjustment of its value. To prepare for the next steppoint control, the pulse counter is cleared to zero.

3.6.2. Control

Task 2 starts control operation by comparing the current voltage sweep amplitude (current AMU) with the voltage sweep amplitude (End AMU) that marks the end of the current breakpoint. If it is larger, the next breakpoint's parameters are transferred to the scratch pad and are used to control the operation of the QMF. Otherwise, the current AMU is incremented by the value of the AMU increment. The resultant value and other parameters of the current breakpoint are used to control the operation of the QMF. In both cases, steppoint control is initiated just after the required parameters are output to the QMF ports.

At the beginning of each new breakpoint, breakpoint data storage pointer (BDPOINT) is updated with the value of the current data storage pointer (CDPOINT). The first address points to the last data collected in the previous breakpoint, while the second address points to the last data collected in the current breakpoint. Therefore, BDPOINT always points to the end of valid data buffer to be transferred. When the data buffer memory becomes full during the current breakpoint, only the data collected up to the start of the current breakpoint will be transferred to the CRT. Then control operation is resumed from the beginning of the current breakpoint. The above procedure is essential to insure that the data collected during a breakpoint is done within the time period allowed to it. Otherwise, errors are introduced in the interpretation of the data result.

Task 2 sends PULSE2 to decrement the count held in (COUNTER1) indicating that four bytes of data were added to the memory buffer area.

3.7 EXITER Subroutine

EXITER subroutine is the library manager which when called by other routines provides the scratch pad, and the QMF ports with a specified breakpoint or steppoint parameters. It also initiates time interval counting and data sampling.

3.7.1. Library Management

Library management is processed through three consecutive routines: Get Repertoire (GREP), Get Program (GPROG) and Get KBLOK (GKBLOK). Each one of these routines follows the same data structure, and operates on one hierarchy level pointer to obtain the next lower level pointer. Then it stores both pointers in scratch pad memory. As it is shown in Figure 3-11, the EXITER obtains the archive pointer (the first repertoire pointer) from memory location 800 hexadecimal and makes it available to be used by GREP via the scratch pad.

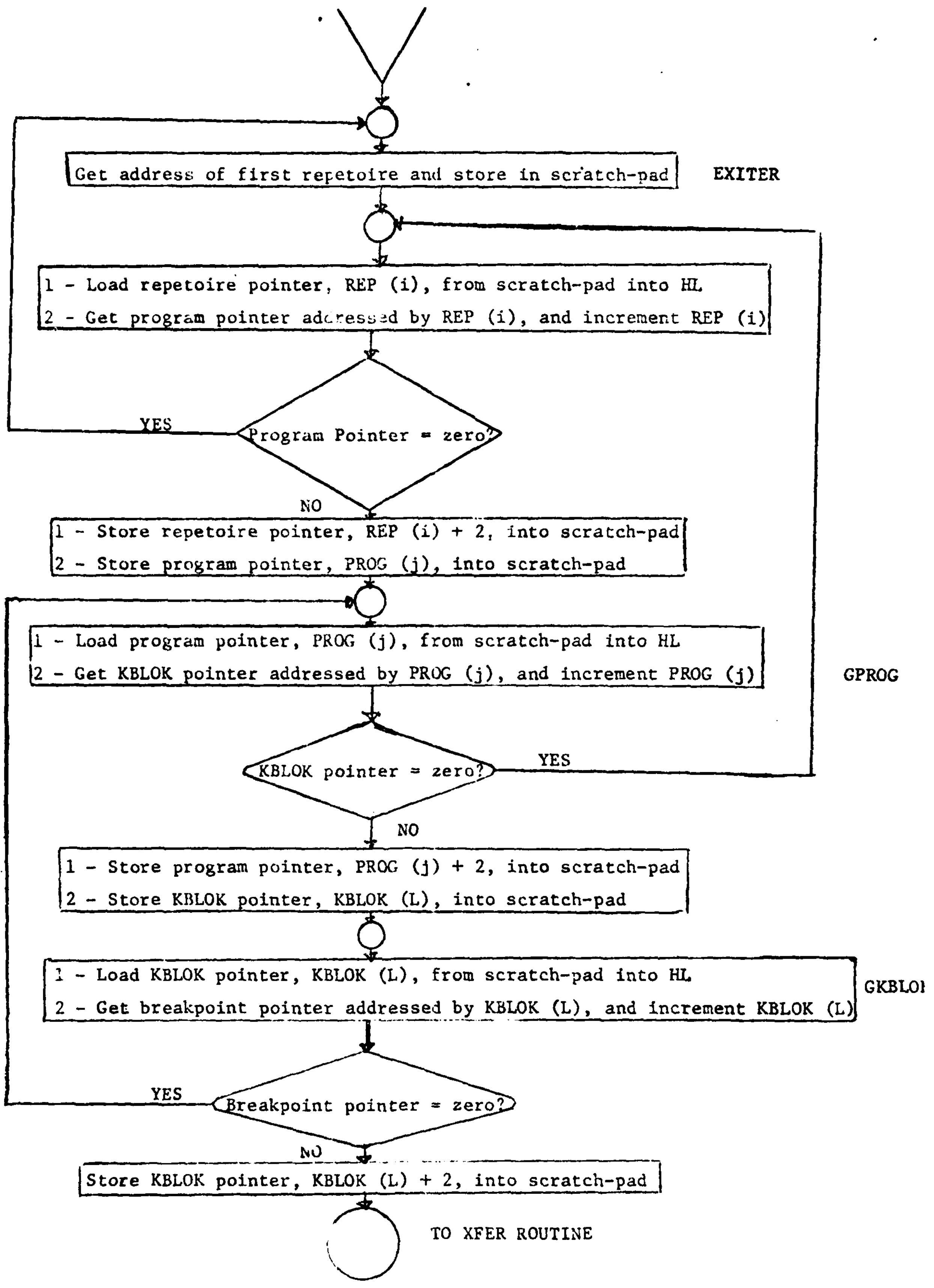
GREP loads its level pointer which is the repertoire pointer from scratch pad into HL registers. And it uses the pointer to obtain the next lower pointer which is the program pointer via LDE subroutine. If the program pointer is equal to zero, which is the mark for the end of the level pointers list, then LDE sets the Z-flag for GREP to use as a conditional branch to the start of the EXITER, and repeates processing the archive.

Similarly GPROG operates on program pointer to obtain a KBLOK pointer. If the latter is zero, a branch is made to GREP for processing a repertoire in sequence. Otherwise both pointers, program pointer and KBLOK pointer, are stored in scratch pad. Following GPROG routine is GKBLOK routine which operates on KBLOK pointer to get a breakpoint pointer from the library. The latter is tested for zero which is the conditional jump to GPROG. When the condition is false, GKBLOK stores its level pointer into scratch pad and relinquishes the 8085A control to XFER routine.

3.7.2. Control

As shown in Figure 3-12, EXITER subroutine uses the control routines, transfer parameter (XFER) routine and output parameter (OUTPAR) routine.

XFER routine transfers all ten parameters pointed to by the BKPOINT



Get address of first repetoire and store in scratch-pad
EXITER
1 - Load repetoire pointer, REP (i), from scratch-pad into HL
2 - Get program pointer addressed by REP (i), and increment REP (i)
Program Pointer = zero?
YES
NO
1 - Store repetoire pointer, REP (i) + 2, into scratch-pad
2 - Store program pointer, PROG (j), into scratch-pad
1 - Load program pointer, PROG (j), from scratch-pad into HL
2 - Get KBLOK pointer addressed by PROG (j), and increment PROG (j)
GPROG
KBLOK pointer = zero?
YES
NO
1 - Store program pointer, PROG (j) + 2, into scratch-pad
2 - Store KBLOK pointer, KBLOK (L), into scratch-pad
1 - Load KBLOK pointer, KBLOK (L), from scratch-pad into HL
2 - Get breakpoint pointer addressed by KBLOK (L), and increment KBLOK (L)
GKBLOI
Breakpoint pointer = zero?
YES
NO
Store KBLOK pointer, KBLOK (L) + 2, into scratch-pad
TO XFER ROUTINE

Figure 3 - 11. Flow-Chart diagram of EXITER library management routines.

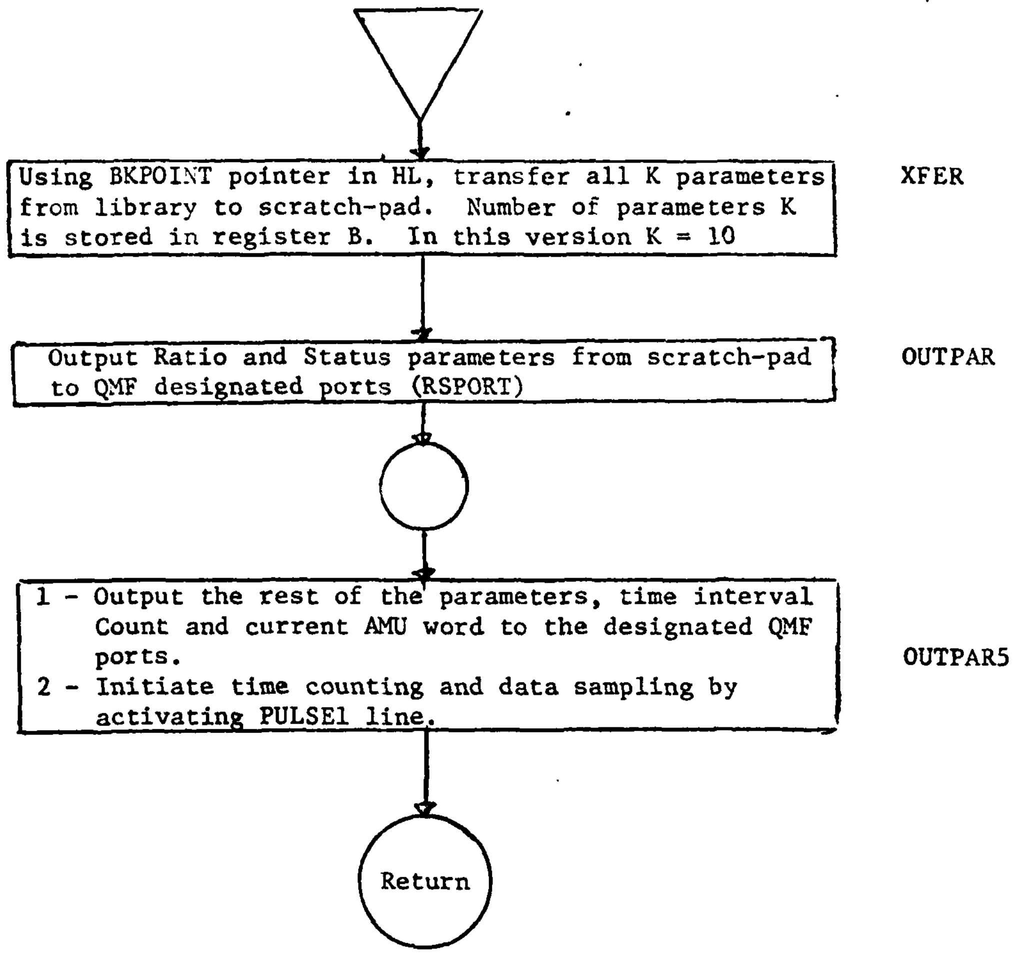

Using BKPOINT pointer in HL, transfer all K parameters from library to scratch-pad. Number of parameters K is stored in register B. In this version K = 10
XFER
Output Ratio and Status parameters from scratch-pad to QMF designated ports (RSPORT)
OUTPAR
1 - Output the rest of the parameters, time interval Count and current AMU word to the designated QMF ports.
2 - Initiate time counting and data sampling by activating PULSE1 line.
OUTPAR5
Return

Figure 3 - 12. Flow-Chart diagram of EXITER control routines.

pointer in HL registers from the library to the start of the scratch pad. This is accomplished to provide the capability for other routines to reference these parameters and to change their values.

OUTPAR routine updates the QMF designated ports with the control parameters found in the scratch pad, whenever a new breakpoint is to be initiated. Since the status and ratio control parameters remain unchanged during a breakpoint, OUTPAR5 routine which outputs the rest of the parameters to the QMF ports, is usually called by Task 2 whenever a new steppoint is to be initiated. At the end of the routine OUTPAR, and therefore OUTPAR5 activates PULSE1/line. This allows data pulses to be accumulated in the Pulse Counter, and at the same time, the time interval count to start decrementing. When the memory buffer area becomes full, the count is decremented to zero which will activate RST 6.5 interrupt line. After interrupts are enabled by software, then Task 3 routine starts transferring data from buffer area to CRT screen.

Task 2 routine terminates its functions by restoring the 8085A registers, enabling interrupts and relinquishing the control of the 8085A to the idle loop.

3.7.3. Task 2 Execution Time

Execution time for Task 2, Δt_2, is defined to be the time the routine starts execution until sampling pulse (PULSE1/) is activated for the new steppoint. The value of Δt_2 is calculated by adding all the clock cycles required to execute each instruction in the Task 2 routine, and multiplying the results by the clock period. When a new steppoint is initiated Δt_2 = 541 (clock cycles) X 0.326 (μs/clock cycle) = 176.366 μs. This value is increased to 362.2 Δsec. if a new breakpoint is initiated. Thus Δt_2 is a variable quantity depending on the library level to which Task 2 has reached.

3.8 Task 3 (RST 6.5 Interrupts)

Task 3 routine transfers a block of data from memory to CRT screen at a baud rate of 9600. A file header is attached to the start of the block for identification of data. Before surrendering control of the processor to the idle loop, the routine initiates control operation of the QMF from the beginning of the current breakpoint. Figure 3-13 shows the display format that appears on the screen.

```
This Is The PCU First Run.
Date:  3/28/1978
File#:  (value)
Repertoire           Program          KBLOK
(value)              (value)          (value)

Data      Time      Data      Time      Data      Time      Data      Time
(value)  (value)   (value)
(value)  (value)    etc.
(value)  (value)    etc.
```

Figure 3-13. File Header and Block of Data Display Format.

3.8.1. Transfer

As shown in Figure 3-14, TRANX routine after disabling interrupts it displays file header using MSGOUT and an EPROM-based table of the required file characters. At the appropriate places, it inserts the value of the file number for that block of data, and the three level pointers, repertoire, program and KBLOK pointers. Each one of these level pointers addresses its next operational level. Therefore, the data displayed on CRT are the results of processing the parameters pointed to in the library by the last block level pointers and up to the present block level pointers.

Data are displayed columnwise in hexadecimal format such that alternate columns represent the same type of data. The first alternate columns represent the actual time interval count. Each row shows data collected in

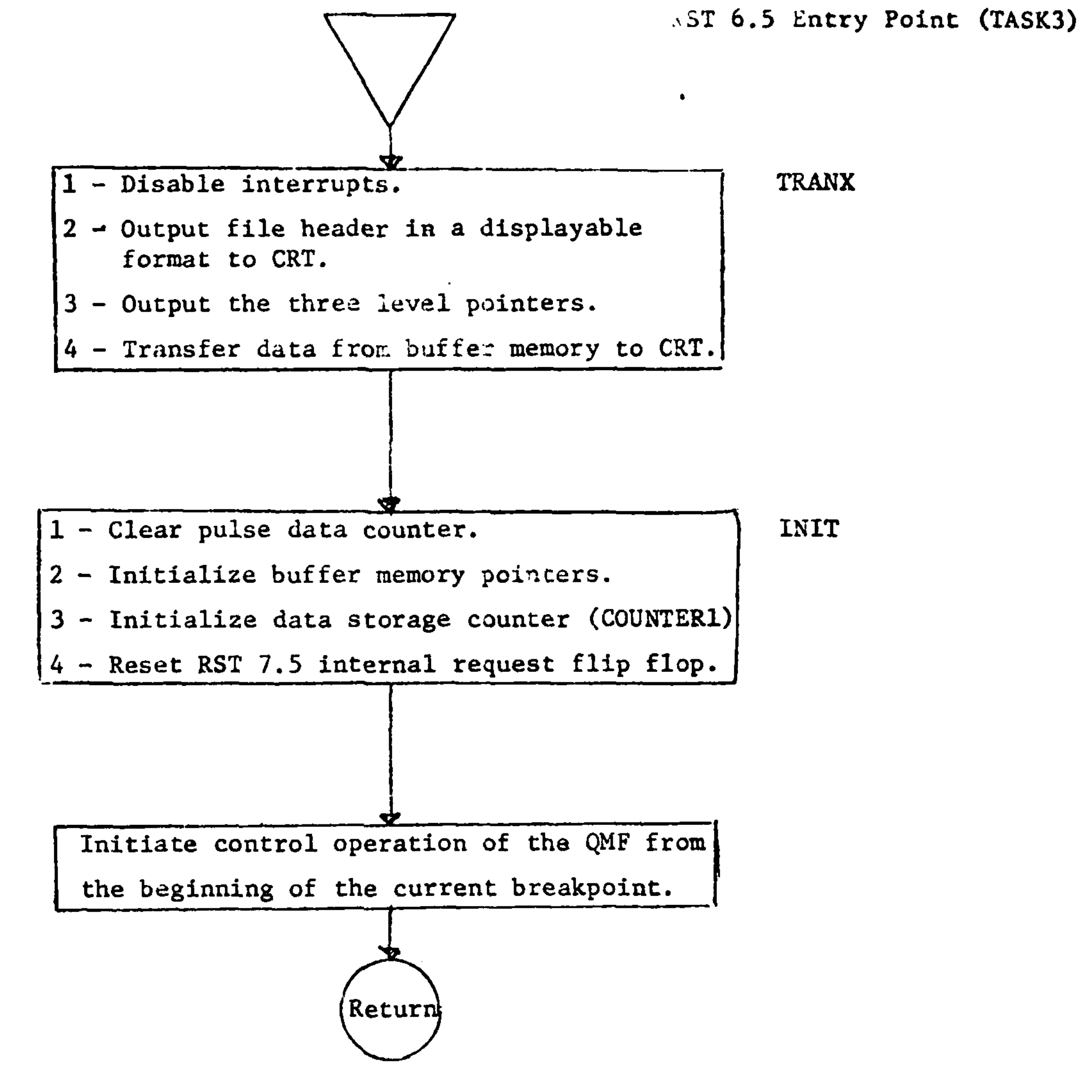

RST 6.5 Entry Point (TASK3)
1 - Disable interrupts.
2 - Output file header in a displayable format to CRT.
3 - Output the three level pointers.
4 - Transfer data from buffer memory to CRT.
TRANX
1 - Clear pulse data counter.
2 - Initialize buffer memory pointers.
3 - Initialize data storage counter (COUNTER1)
4 - Reset RST 7.5 internal request flip flop.
INIT
Initiate control operation of the QMF from the beginning of the current breakpoint.
Return

Figure 3-14. Flow Diagram of Task 3 Routine.

sequential order of first-in-first-out (FIFO) structure.

The size of the block of data is determined by the size of the buffer memory. The last data to be displayed on the CRT is bounded by the breakpoint data storage pointer (BDPOINT) which points to the first data not to be transferred. Thus BDPOINT helps in terminating the display of data on the CRT.

3.8.2. Control

To resume QMF control operation and data acquisition INIT routine executes the following steps:

1. Clear the data pulse counter by activating PULSE0/line.
2. Pull down PULSE3/line to reset RST 6.5 interrupt request flip flop and to initiate COUNTER1 counting operation.
3. Initialize buffer memory pointers to their values.
4. Reset RST 7.5 internal request flip flop.
5. Initiate the start of the current breakpoint control by calling GKBLOK routine.
6. Enable interrupts and relinquish control of the processor to the idle loop.

3.9 Programming the PCU-85

The number of breakpoints and level pointers required for a balloon flight are limited only by the size of the memory assigned to the library. The primary sources of this memory are Intel 2716 EPROMS. Although the starting address is at 800 hexadecimal, the rest of the library can be virtually anywhere in the memory space available.

Programming the library involves the following steps:

1. Once the number of breakpoints are determined and assigned to certain memory locations, then the rest of the available memory space is distributed among the three hierarchy level pointers. This process is termed space management.
2. Calculate the hexadecimal values of all breakpoint parameters that are required to generate the desired control signals.

3. Using the development software monitor DSM-85 and a CRT, the operator can program the library in an Intel 2716 EPROM.

An example memory space distribution and parameter calculation is given below.

3.9.1. Space Management

A proposed memory space organization for a hundred breakpoints library is given in Table 3-1. A breakpoint has processing capabilities of more than just one AMU number. Therefore, one Intel 2716 EPROM may be sufficient for the present balloon borne ion mass spectrometer (BBIMS) specification.

Any number of pointers can be assigned to one level as long as a zero is inserted in the last pointer memory location to mark the end of that level.

Breakpoint parameters could be allocated to the higher memory space in the EPROM, and the higher hierarchy levels could be allocated to the lower memory space. This memory space allocation provides a unified approach to library construction for debugging and ease of referencing only.

Level	Number of levels	Number of Bytes in each level	Number of Pointers in each level	Total Bytes
Breakpoint	100	10	None	1000
KBLOK	15	50	24	750
Program	12	22	10	264
Repertoire	1	13	12	26
Archive	1	1	1	2
Total bytes required				2042
Total bytes available				2048

Table 3-1. Memory Space Distribution for Intel 2716 EPROM-based Library.

3.9.2. Parameters Calculations

It is easy to construct tables for each control parameter in hexadecimal numerals and their corresponding physical values such as voltage or time.

Then for any desired physical value, a hexadecimal number is selected from the table and is written in an Intel 2716 EPROM.

Table 3-2 gives the hexadecimal values of each time interval count and its corresponding real time values in microseconds when 1.536 MHz clock is used, and in milliseconds when 2 kHz clock is used.

Table 3-3 gives the hexadecimal values of each voltage sweep control and its corresponding voltage value in volts. This value is dependent on the DAC being used and the reference voltage V_{REF} applied to it.

Similar tables could be constructed for the rest of the control parameters.

Time Interval Count (HEX)	Time Value (μs) using 0.65 μs clock	Time Value (ms) using 0.50 ms clock
0000	42598.40	32768.00
0001	0.65	0.50
0002	1.30	1.00
⋮	⋮	⋮
FFFF	42597.75	32767.50

Table 3-2. Time Interval Count.

Table 3-3. Voltage Sweep Control.

Voltage Sweep Control (HEX)	Analog Value (volts)
0FFF	$-V_{REF}(1 - 2^{-12})$
0801	$-V_{REF}(1/2 + 2^{-12})$
0800	$-V_{REF}(1/2)$
07FF	$-V_{REF}(1/2 - 2^{-12})$
0001	$-V_{REF}(2^{-12})$
0000	0

3.10 Conclusion

The PCU-85 system was designed and developed for two purposes. The primary target is to control a balloon-borne quadrupole ion mass spectrometer (BBQIMS) and to acquire data from the instrument for intermediate storage in RAM with possible intercommunication between the system and ground station computer. Due to the lack of a microprocessor development system at the time, additional hardware components and a development software package were added to the system. Therefore, the PCU-85 achieves its second purpose as a development system. The PCU-85 monitor and diagnostic routines were developed on the PCU-85.

An advanced version of the PCU-85 is proposed in Figure 3-15. In the new system, the PCU-85A, most of the circuits that are used for development purposes, such as the Debug Circuit and Advance Status Decoder, are removed. To provide for more processing power and an increase in the capability of the PCU-85A additional I/O devices and RAM are added.

The RAM is increased to 16K bytes with separate address, data and control buffers that interface it with the 8085A bus system. This increase of storage will enable more data to be stored, and at the same time part of the RAM can be used to store control parameters which are either transmitted from the ground computer or retrieved from a magnetic storage in the balloon.

The new I/O devices that are added to the system are:

1. Am 9511 Arithmetic Processing Unit (APU). The Am 9511 APU is an arithmetic processor that provides fixed and floating point arithmetic and a variety of floating point trigonometric and mathematical operations. It will be used to enhance the computational capability of the 8085A and to provide data processing

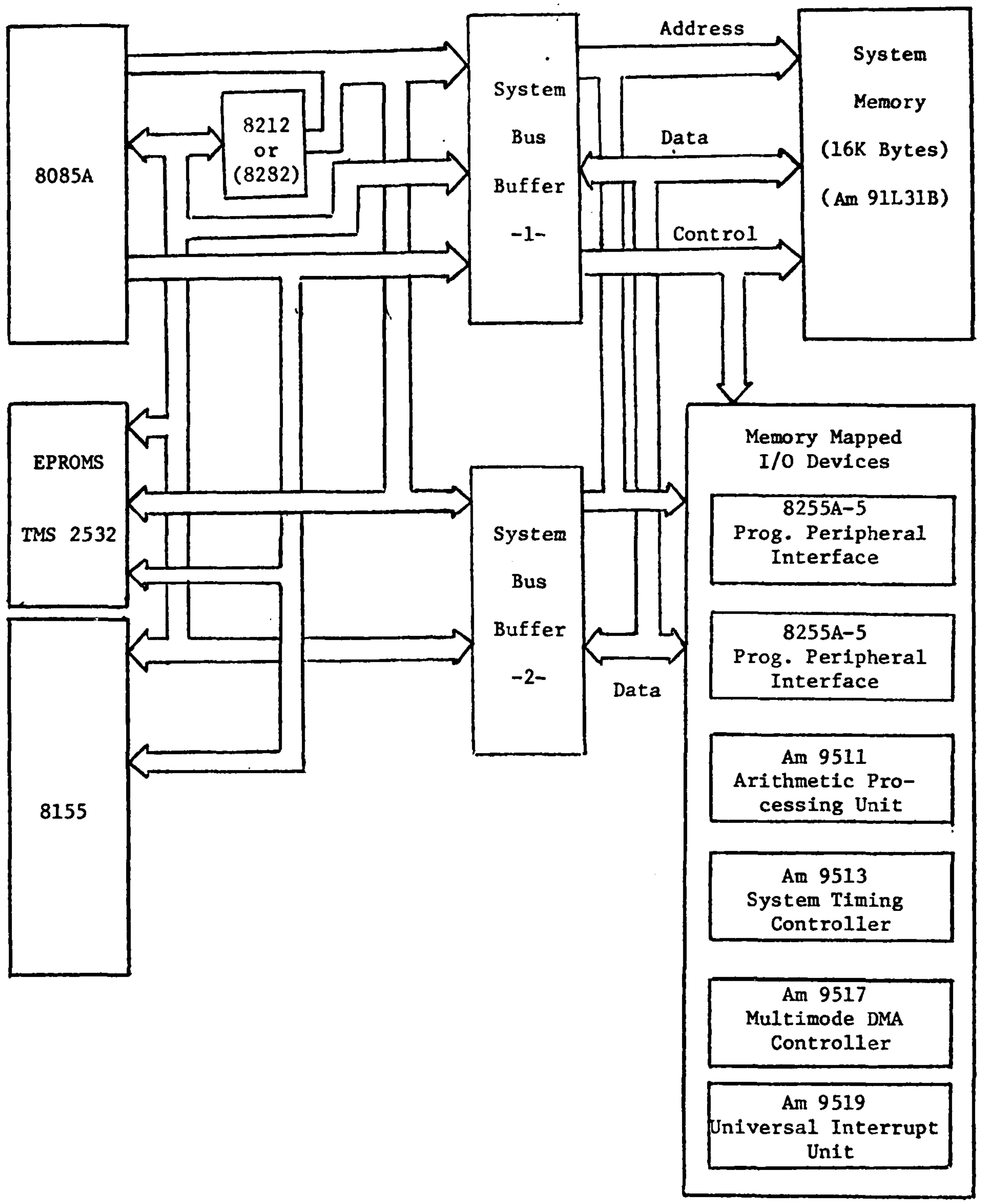


Figure 3-15. PCU-85A System.

on the balloon. Data processes anticipated ar: division of the pulse count by the time interval count and smoothing the results; computation of the balloon altitude and coordinate and some environmental data such as pressure, humidity and temperature.

2. Am 9513 System Timing Controller (STC). The Am 9513 STC is an advanced version of the 8253 Programmable Interval Timer (PIT) with five independent 16-bit counters. Two of the counters may be used for time-of-day counters, and the other counter are to be used in the same manner the counters of the 8253 PIT are used.

3. Am 9517 Multimode Direct Memory Access (DMA) Controller. The Am 9517 DMA allows external devices, such as the memory mapped I/O devices, to directly transfer information to or from the system memory. This improves system performance. In addition, the 8085A can be doing control processing since it is isolated from the system memory and the memory-mapped I/O devices by system bus buffers one and two. The system should be designed such that the 8085A has a higher priority than the Am 9517 DMA to access the system bus buffers when contention arises.

4. Am 9519 Universal Interrupt Control (UIC). The Am 9519 UIC is added to enhance the 8085A interrupt handling capability and to increase the number of tasks to be performed by the PCU-85A system. Some of the housekeeping functions can be assigned to these additional tasks. These functions are Digital-to-Analog, and Analog-to-Digital conversions of environmental data (e.g., pressure, temperature, etc.). Also, a communication link can be assigned to one of the tasks.

Appendix A

Appendix A contains the PCU-85 software assembly listing. General descriptions of these programs are found in Chapter 3.

1 - Most of the RESTART entry points are used for branching to their respective interrupt service routines.

2 - RESET entry point is reserved for system initialization.

3 - Task 2 routine uses RST 7.5 entry point for its executive functions. It supervises control operation of the QMF and the data acquisition resulting from such an operation.

Labels	Address	HEX Inst	D8 OR A16	Mnemonics	No. States	Comments
	0000	F3		DI		RST 0 Entry Point
	0001	21		LXI H, SPOINT		Load HL with Stack Pointer
	0002		A1			
	0003		20			
	0004	F9		SPHL		Load SP with contents of HL.
	0005	C3		JMP INITZ		Jump to initialization routine.
	0006		94			
	0007		00			
	0008	C3		JMP		RST 1 Entry Point
	0009					
	000A					
	000B					
	000C					
	000D					
	000E					
	000F					
	0010	C3		JMP		RST 2 Entry Point
	0011					
	0012					
	0013					
	0014					
	0015					
	0016					
	0017					
	0018	C3		JMP		RST 3 Entry Point
	0019					
	001A					
	001B					
	001C					
	001D					
	001E					
	001F					
	0020	C3		JMP		RST 4 Entry Point
	0021					
	0022					
	0023					
	0024	C3		JMP PFR		Trap RST (4.5) Entry Point
	0025					Reserved for power failure.
	0026					

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	0027					
	0028	C3		JMP		RST 5 Entry Point
	0029					
	002A					
	002B					
	002C	C3		JMP MON		RST (5.5) Int. Entry Point.
	002D		BC			
	002E		00			
	002F					
	0030	C3				RST 6 Entry Point.
	0031					
	0032					
	0033					
	0034	C3		JMP TASK2		RST (6.5) Int. Entry Point.
	0035		48			JMP to data transfer routine.
	0036		01			
	0037					
	0038	C3				RST (7) Entry Point.
	0039					
	003A					
	003B					
TASK2	003C	F3		DI	04	RST (7.5) Int. Entry Point.
	003D	C5		PUSH B	12	Save 8085 Registers.
	003E	D5		PUSH D	12	
	003F	E5		PUSH H	12	
	0040	F5		PUSH PSW	12	
DFETCH	0041	2A		LHLD ACC	16	Get data count and load it into HL.
	0042		08			
	0043		21			
	0044	EB		XCHG	04	Put data count into DE.
	0045	2A		LHLD CDPOINT	16	Get data pointer in HL.
	0046		F8			
	0047		20			
	0048	CD		CALL SDE	18	Call SDE routine.
	0049		40			
	004A		02			
	004B	7A		MOV A, D	4	Move high byte of data count into A.
	004C	EB		XCHG	4	Put data pointer into DE.
	004D	2A		LHLD TIME	16	Load time interval count into HL.
	004E		08			
	004F		20			
	0050	EB		XCHG	4	Exchange data contents of HL and DE.
	0051	A7		ANA A	4	Check overflow of data count.
	0052	F2		JP DFETCH 5	7/10	If not jump to DFETCH 5.
	0053		60			If yes. Adjust time interval count.
	0054		00			
	0055	E5		PUSH H	12	Save DPOINT.
	0056	21		LXI H, TIMPORT	10	Get remainder of time interval count.
	0057		0C			
	0058		21			

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	0059	7B		MOV A, E	4	Subtract (TI-TIME Residue), and store
	005A	96		SUB M	7	Results in D. (actual time interval)
	005B	5F		MOV E, A	4	
	005C	7A		MOV A, D	4	
	005D	9E		SBB M	7	
	005E	57		MOV D, A	4	
	005F	E1		POP H	10	Restore the data pointer.
DFETCH5	0060	CD		CALL SDE	18	Store (DE) in location pointed toby HL
	0061		40			
	0062		02			
	0063	22		SHLD CD POINT	16	Store current data pointer.
	0064		F8			
	0065		20			
	0066	D3		OUT PULSEO	10	Clear data count accumulator.
	0067		80			
CONTROL	0068	2A		LHLD AMU	16	Check if sweep reached AMUEND.
	0069		02			
	006A		20			
	006B	EB		XCHG	4	
	006C	2A		LHLD AMUEND	16	Load AMUEND into HL.
	006D		04			
	006E		20			
COMPARE	006F	7D		MOV A, L	4	Check if (HL) - (DE) 0
	0070	93		SUB E	4	
	0071	7C		MOV A, H	4	
	0072	9B		SBB D	4	
	0073	D2		JNC INCR	7/10	CY = 1 if (DE) ≥ (HL)
	0074		82			CY = 0 if (DE) < (HL)
	0075		C0			
	0076	CD		CALL GKBLOK	18	Get next breakpoint and output its
	0077		C7			parameters to QMF
	0078		C2			
	0079	2A		LHLD CD POINT	16	Update data storage pointer since a
	007A		F8			new breakpoint is reached.
	007B		20			
	007C	22		SHLD BD POINT	16	This pointer will be used in INT 6.5
	007D		F6			service routine.
	007E		20			
	007F	C3		JMP RESTORE	10	Restore the 8085 register.
	0080		8C			
	0081		00			
INCR	0082	2A		LHLD AMUINC	16	GET AMUINC into HL.
	0083		C6			
	0084		20			
	0085	19		DAD D	10	Add AMUINC, to the current AMU.
	0086	22		SHLD AMU	16	Store in scratch pad.
	0087		02			
	0088		20			
	0089	CD		CALL OUTPAR5	18	Output parameters.
	008A		27			
	008B		02			

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
RESTORE	008C	D3		OUT PULSE2	10	Decrement storage indicator.
	008D		82			
	008E	F1		POP PSW	10	restore 8085 registers and return
	008F	E1		POP H	10	to calling program.
	0090	D1		POP D	10	
	0091	C1		POP B	10	
	0092	FB		EI	4	
	0093	C9		RET	10	

After power-up sequence the Monitor jumps to INITZ rout... This routine initializes the integrated system (hardware and software). In addition, it initiates mass spectrometer control, and data acquisition. Finally, it enters the Idle routine.

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
INITZ	0094	22		SHLD, 20 BC		Store SP in scratch pad.
	0095		3C			
	0096		20			
	0097	21		LXI, INTABL		Load pointer of initialization table.
	0098		E0			
	0099		03			
	009A	06		MVI B, NRUN		Load B with half the size of the
	009B		0A			table.
	009C	16		MVI D, 21		Load D with most significant bits of
	009D		21			memory-mapped I/O pointer.
INITZ5	009E	5E		MOV E, M		Initialize system ports, by outputting
	009F	23		INX H		control words from table.
	00A0	7E		MOV A, M		
	00A1	12		STAX D		
	00A2	23		INX H		
	00A3	05		DCR B		
	00A4	C2		JNZ, INITZ5		If table is not exhausted repeat,
	00A5		9E			otherwise continue.
	00A6		00			
	00A7	D3		OUT, PULSE0		CLEAR data count accumulator.
	00A8		80			
	00A9	D3		OUT, PULSE3		Initialize COUNTER1 operation for
	00AA		83			RST (6.5) INT.
INITZ10	00AB	21		LXI H, SDPOINT		Initialize current data storage pointer
	00AC		00			to buffer start address.
	00AD		24			
	00AE	22		SHLD H, CDPOINT		Store HL in scratch pad.
	00AF		F8			
	00B0		20			
	00B1	3E		MVI, 18		Unmask all Restart interrupts.
	00B2		18			
	00B3	30		SIM		
BEGIN	00B4	CD		CALL EXITER		Output QMF parameters.
	00B5		E1			
	00B6		01			
	00B7	FB		EI		Enable interrupts.
IDLE	00B8	00		NOP		Get into idle loop, wait for an
	00B9	C3		JMP IDLE		interrupt to occur.

AD-A106 398   NORTHEASTERN UNIV BOSTON MASS ELECTRONICS RESEARCH LAB   F/G 9/2
A PROGRAMMABLE CONTROL UNIT FOR A BALLOON-BORNE MASS SPECTROMET--ETC(U)
SEP 79 V C GEROUSIS   F19628-78-C-0218
UNCLASSIFIED   SCIENTIFIC-1   AFGL-TR-79-0225   NL

2 OF 2
A106398

END
DATE
FILMED
11-81
DTIC

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	00BA		B8			
	00BB		00			

INT: - RST 6.5 (TASK 3)

RST 6.5 interrupt service routine transfers data collected in buffer memory to CRT terminal. First a header is transferred in front of the data for identification purposes. Then a block of data, bounded by the buffer start address and the breakpoint data storage pointer, is transmitted as ASCII characters to the TTY terminal.

At the end of the transfer, control operation of the QMF is initiated, interrupt is enabled, and then Task 3 returns to the Idle Loop.

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
TRANX	0148	F3		DI		Task 3
	0149	D3		OUT PULSE2		Complete clock period for COUNTER1, and
	014A		82			make OUT1 high.
	014B	21		LXI H, M POINT1		Load HL with message pointer of
	014C		00			first part.
	014D		07			
	014E	CD		CALL MSGOUT		Output message on CRT screen.
	014F		C2			
	0150		01			
	0151	21		LXI H, FILEP		Load HL with file number pointer
	0152		F0			
	0153		20			
	0154	7E		MOV A, M		Load A with file number.
	0155	34		INR M		Increment file number for next operatio
	0156	CD		CALL NMOUT		Output file number to screen.
	0157		8A			
	0158		03			
	0159	21		LXI H, MPOINT2		Load HL with message pointer of second
	015A		33			part.
	015B		07			
	015C	CD		CALL MSGOUT		Output message on CRT screen.
	015D		C2			
	015E		01			
TRANX5	015F	2A		LHLD REPBUF		Load HL with the content of the repetoire pointer
	0160		FE			
	0161		20			
	0162	CD		CALL DHL		Display contents of HL in HEX
	0163		D8			
	0164		01			
	0165	06		MVI B, 06		Load B with number of spaces to be
	0166		06			displayed on screen
	0167	CD		CALL SPACE		Display 6 spaces on CRT screen.
	0168		CE			
	0169		01			
	016A	2A		LHLD PROGBUF		Load contents of program pointer into
	016B		FC			HL
	016C		20			
	016D	CD		CALL DHL		Display contents of HL in HEX.
	016E		D8			

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	016F		01			
	0170	06		MVI B, 06		Load B with number of spaces to be
	0171		06			displayed on screen.
	0172	CD		CALL SPACE		Display 6 space characters on screen.
	0173		CE			
	0174		01			
	0175	2A		LHLD KBLOK		Load KBLOK pointer into HL.
	0176		FA			
	0177		20			
	0178	2B		DCX H		Decrement KBLOK pointer so as to
	0179	2B		DCX H		point to the current KBLOK pointer.
	017A	22		SHLD KBLOK		Store the adjusted KBLOK pointer
	017B		FA			into scratch pad.
	017C		20			
	017D	CD		CALL DHL		Display KBLOK pointer on screen.
	017E		D8			
	017F		01			
	0180	21		LXI H, MPOINT3		Load HL with message pointer of third
	0181		50			part.
	0182		07			
	0183	CD		CALL MSG		Display third part of message.
	0184		C2			
	0185		01			
	0186	2A		LHLD BDPOINT		Load HL with contents of breakpoint da
	0187		F6			pointer which points to the last data
	0188		20			collected in the previous KBLOKexecuti
	0189	EB		XCHG		Load HL into DE.
	018A	21		LXI H, SDPOINT		Load HL with the start of data buffer
	018B		00			pointer.
	018C		24			
	018D	22		SHLD, CDPOINT		Store SDPOINT in current data
	018E		F8			buffer pointer.
	018F		20			
NLINE	0190	CD		CALL CROUT		Output a new line on CRT screen
	0191		B1			
	0192		02			
SAME	0193	OE		MVI C, 20		Load C with ASCII code for space.
	0194		20			
	0195	CD		CALL ECHO		ECHO space character on screen.
	0196		BE			
	0197		02			
	0198	7E		MOV A, M		Get high data byte
	0199	F5		PUSH PSW		Save data in memory.
	019A	23		INX H		Increment HL.
	019B	7E		MOV A,M		Get low data byte.
	019C	CD		CALL NMOUT		Display low data byte.
	019D		8A			
	019E		03			
	019F	F1		POP PSW		Load high data byte.
	01A0	CD		CALL NMOUT		Display high data byte.

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	01A1		8A			
	01A2		03			
	01A3	CD		CALL HILO		Check if HL $\geq$ DE
	01A4		63			
	01A5		03			
	01A6	D2		JNC CONTIN		If HL $<$ DE go to CONTIN
	01A7		B8			to display more data
	01A8		01			
	01A9	CD		CALL CROUT		Otherwise, terminate display, and issue
	01AA		B1			carriage return (CR)
	01AB		02			
INIT	01AC	D3		OUT PULSE0		Clear pulse counter
	01AD		80			
	01AE	D3		OUT PULSE3		Initialize COUNTER1 for next
	01AF		83			counting operation.
	01B0	3E		MVI A, 18		Load A with interrupt mask word
	01B1		18			to reset RST 7.5 interrupt
	01B2	30		SIM		Set Interrupt Masks
	01B3	CD		CALL GKBLOK		Reactivate system with execution
	01B4		07			of the current KBLOK
	01B5		02			
	01B6	FB		EI		Enable Interrupts.
	01B7	C9		RET		Return to the
CONTIN	01B8	23		INX H		Point to next data byte
	01B9	7D		MOV A, L		Load low byte of HL into A
	01BA	E6		ANI OF		See if last HEX digit of pointer
	01BB		OF			denotes'start of new line
	01BC	C2		JNZ SAME		No - not at end of line of display.
	01BD		93			
	01BE		01			
	01BF	C3		JMP NLINE		Yes - start new line of display.
	01C0		90			
	01C1		01			
MSG	01C2	4E		MOV C, M		Load a byte from memory into C.
	01C3	CD		CALL CO		Output byte to screen
	01C4		31			
	01C5		02			
	01C6	23		INX H		Point to next byte in memory.
	01C7	79		MOV A,C		Load byte in C into A.
	01C8	FE		CPI 00		See end of message indicator.
	01C9		00			
	01CA	C2		JNZ MSG		No - Output next byte to CRT.
	01CB		C2			
	01CC		01			
	01CD	C9				Yes - Return to calling routine.
SPACE	01CE	0E		MVI C, 20		Load C with ASCII code for space.
	01CF		20			
SPACE +1	01D0	CD		CALL CO		Output one character to screen.
	01D1		81			
	01D2		02			

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	01D3	05		DCR B		Decrement number of spaces
	01D4	C2		JNZ SPACE+1		If count in B is not zero output
	01D5		D0			another space character.
	01D6		01			
	01D7	C9		RET		Otherwise, return to calling routine.
DHL	01D8	7C		MOV A, H		Load H into A.
	01D9	CD		CALL NMOUT		Output content of A as a HEX digit.
	01DA		8A			
	01DB		03			
	01DC	7D		MOV A, L		Load L into A.
	01DD	CD		CALL NMOUT		Output content of A as a HEX digit.
	01DE		8A			
	01DF		03			Return to calling routine.
	01E0	C9		RET		

EXITER:

The EXITER subroutine is the manager subroutine that keeps track and updates all parameter pointers in the library. It transfers the current breakpoint parameters to scratch-pad memory through XFER routine. Then it outputs all the parameters from the scratch-pad memory to the QMF ports and counters via OUTPAR. Finally, it initiates steppoint control, data sampling and time interval counting by activating PULSE1/ line.

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
EXITER	01E1	2A		LHLD 0800	16	Get pointer for archive
	01E2		00			(first repetoire pointer)
	01E3		08			
	01E4	22		SHLD REPBUF	16	Store in scratch-pad.
	01E5		FE			
	01E6		20			
GREP	01E7	2A		LHLD REPBUF	16	Get REP pointer.
	01E8		FE			
	01E9		20			
	01EA	CD		CALL LDE	18	Get program pointer and store
	01EB		39			in DE.
	01EC		02			
	01ED	CA		JZ EXITER	7/10	If all repetoires are executed, repeat
	01EE		E1			execution from the start. Otherwise
	01EF		01			continue.
	01F0	22		SHLD REPBUF	16	Store the new REP pointer into
	01F1		FE			scratch-pad.
	01F2		20			
	01F3	EB		XCHG	4	
	01F4	22		SHLD PROGBUF	16	Store programs pointer into
	01F5		FC			scratch-pad.
	01F6		20			
GPROG	01F7	2A		LHLD PROGBUF	16	Load program pointer into HL.
	01F8		FC			
	01F9		20			
	01FA	CD		CALL LDE	18	Load KBLOK pointer in DE.
	01FB		39			
	01FC		02			
	01FD	CA		JZ GREP	7/10	If zero, get the next repetoire,
	01FE		E7			otherwise continue.
	01FF		01			
	0200	22		SHLD PROGBUF	16	Store program pointer into
	0201		FC			scratch-pad.
	0202		20			
	0203	EB		XCHG	4	
	0204	22		SHLD KBLOK	16	Store KBLOK pointer into
	0205		FA			scratch-pad
	0206		20			
GKBLOK	0207	2A		LHLD KBLOK	16	Load KBLOK pointer into HL.

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	0208		FA			
	0209		20			
GKBLOK+3	020A	CD		CALL LDE	18	Load BKPOINT pointer into DE.
	020B		39			
	020C		02			
	020D	CA		JZ GPROG	7/10	If zero, get the next program
	020E		F7			pointer. Otherwise continue.
	020F		01			
	0210	22		SHLD KBLOK	16	Store KBLOK pointer into
	0211		FA			scratch-pad.
	0212		20			
	0213	EB		XCHG	4	
XFER	0214	11		LXI D, SCADST	10	Transfer control parameters of
	0215		00			BKPOINT to scratch-pad.
	0216		20			
	0217	06		MVI B, 0A	7	Load B with number of bytes to be
	0218		0A			transferred.
XFER5	0219	7E		MOV A, M	7	DO the transfer, until B is
	021A	12		STAX D	7	decremented to zero.
	021B	23		INX H	10	
	021C	13		INX D	10	
	021D	05		DCR B	4	
	021E	C2		JNZ XFER5	7/10	
	021F		19			
	0220		02			
OUTPAR	0221	2A		LHLD RATIO & STATUS	16	Load Ratio and Status control
	0222		00			parameters into HL.
	0223		20			
	0224	22		SHLD RS PORT	16	Output HL to QMF
	0225		01			
	0226		21			
OUTPAR5	0227	2A		LHLD TIM	16	GET time interval count into HL.
	0228		08			
	0229		20			
	022A	EB		XCHG	4	
	022B	21		LXI H, TIMPORT	10	Load HL with time interval port
	022C		0C			address.
	022D		21			
	022E	73		MOV M, E	7	Transfer the time interval.
	022F	72		MOV M, D	7	
	0230	2A		LHLD AMU	16	GET AMU word and store into HL.
	0231		04			
	0232		20			
	0233	22		SHLD AMUPORT	16	Output AMU word to QMF.
	0234		11			
	0235		21			
	0236	D3		OUT PULSE1	10	Initiate sampling data pulses.
	0237		81			
	0238	C9		RET	10	Return to calling routine.
LDE	0239	5E		MOV E, M	7	Load DE with two bytes from

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	023B	56		MOV D, M	7	
	023C	23		INX H	6	
	023D	7B		MOV A, E	4	Check if the two bytes are both
	023E	B2		ORA D	4	zero and set flag Z when true.
	023F	C9		RET	10	Return to calling routine.
SDE	0240	73		MOV M, E	7	Store the contents of DE in
	0241	23		INX H	6	memory at the address found in
	0242	72		MOV M, D	7	HL.
	0243	23		INX H	6	
	0244	C9		RET	10	Return to calling routine.

Labels	Address	HEX Inst	D8 or A16	Mnemonics	No. States	Comments
DIGTB	0245		30	0		Table of ASCII Hex digits
	0246		31	1		
	0247		32	2		
	0248		33	3		
	0249		34	4		
	024A		35	5		
	024B		36	6		
	024C		37	7		
	024D		38	8		
	024E		39	9		
	024F		41	A		
	0250		42	B		
	0251		43	C		
	0252		44	D		
	0253		45	E		
	0254		46	F		

Initialization Table: The table is organized into words of two bytes each where

(i) first byte is the low byte address which points to a control word register, or a counter in a peripheral device.

(ii) second byte is the control word to initialize the target device ports or counters.

Labels	Address	HEX Inst	08 A16	Mnemonics	NO. States	Comments
INTABL	03E0		00			Low byte address of 8155 control
	03E1		CF			register control word for 8155
	03E2		OU			low byte address of 8155 timer
	03E3		FF			low byte count of 8155 timer
	03E4		05			high byte address of 8155 timer
	03E5		10			high byte count of 8155 timer
	03E6		OB			low byte address of 8255A (A20) CR
	03E7		92			Control word for 8255A (A20)
	03E8		13			Low byte address of 8255A (A21) CR
	03E9		80			control word for 8255A (A21)
	03EA		OF			Low byte address of 8253 (A22) CR
	03EB		38			control word for counter 0
	03EC		OF			Low byte address of 8253 (A22) CR
	03ED		7A			control word for counter 1
	03EE		OF			Low byte address of 8253 (A22) CR
	03EF		BA			control word for counter 2
	03F0		OD			Low byte address of counter 1
	03F1		20			low byte count for counter 1 register
	03F2		OD			Low byte address of counter 1
	03F3		00			high byte count for counter 1 register

APPENDIX B

PART 1

ITY Utility Routines

The PCU-85 software monitors can communicate serially at 9600 BAUD rate with any device that has 'RS232C' interface. This is accomplished by using special software subroutines, defined as Teletypewriter (TTY) utility routines. An index and a summary of each routine's function are first described. Then a list of these routines in 8085A assembly mneumonics language and their corresponding hexidecimal codes is given.

Subroutines Index

1 - NAME: DIGTB (Hexadecimal Digit Table)
ADDRESS: 0245
DESCRIPTION: Table of ASCII character representing in an increasing order all the Hex digits.

2 - NAME: CI (Console Input)
ADDRESS: 0256
INPUTS: None
OUTPUTS: A character from TTY in register A
CALLS: Delay subroutine
DESTROYS: A, F/F's
DESCRIPTION: CI waits until a character has been entered by an operator at the TTY and then returns the character via the A register, to the calling routine.

3 - NAME: CO (Console Output)
ADDRESS: 0281
INPUTS: Character to output to TTY in register C
OUTPUTS: Character output to TTY in register C
CALLS: Delay routine
DESTROYS: A, F/F's
DESCRIPTION: CO sends its input argument to the TTY

4 - NAME: CNVBN (convert from ASCII hex to binary equivalent)
ADDRESS: 02A8
INPUTS: ASCII hex character in register C
OUTPUT: Binary hex in register A
CALLS: Nothing
DESTROYS: A, F/F's
DESCRIPTION: CNVBN converts the ASCII code of a hex digit into its corresponding binary value. It does not check the validity of its input.

5 - NAME: CROUT (Carriage Return Output)
ADDRESS: 02B1
INPUTS: None
OUTPUTS: None
CALLS: ECHO routine
DESTROYS: A, B, C, F/F's
DESCRIPTION: CROUT sends a carriage return and a line feed through the ECHO routine to the console.

6 - NAME: DELAY
ADDRESS: 02B7
INPUTS: 16-bit binary integer in DE, denoting number of times to loop.
OUTPUTS: None
CALLS: Nothing
DESTROYS: A, B, E, F/F's
DESCRIPTION: Delay decrements the input argument until it is counted down to zero, and then returns to caller. The delay time in nanosecond is T_{delay} (ns) = (21 (N - 1) + 20) t_{CYC}

where N = decimal value of the binary integer in DE
t_{CYC} = clock cycle of the 8085A.

7 - NAME: ECHO
ADDRESS: 02BE
INPUTS: Character in register C to echo to terminal.
OUTPUTS: Character in register C echoed to terminal.
CALLS: CO
DESTROYS: A, B, F/F's
DESCRIPTION: ECHO takes a single character as input, and via CO, sends that character to the terminal. A carriage return is echoed as a carriage return with a line feed, and an escape character is echoed as $.

8 - NAME: ERROR
ADDRESS: 02D7
INPUTS: None
OUTPUTS: None
CALLS: ECHO, CROUT, MON (RST 5.5 routine)
DESTROYS: A, B, C, F/F's
DESCRIPTION: ERROR prints the error character, an asterisk, on the console, followed by a carriage return-line feed. And then returns control of the processor to the beginning of RST 5.5 routine.

9 - NAME: GETCH (Get character)
ADDRESS: 02E2
INPUTS: None
OUTPUTS: A character from TTY to register C
CALLS: CI
DESTROYS: A, C, F/F's
DESCRIPTION: GETCH returns a character from the TTY to the calling routine.

10 - NAME: GETHX (Get hexadecimal digit)
ADDRESS: 02E9
INPUTS: None
OUTPUTS: BC - 16 bit integer
D - Character which terminated the integer
Carry Flag = 1 if first character not delimiter
= 0 if first character is delimiter
CALLS: GETCH, ECHO, VALDL, VALDG, CNVBN, ERROR
DESTROYS: A, B, C, D, E, F/F's
DESCRIPTION: GETHX accepts a string of hex digits from the TTY and returns their value as a 16 bit binary integer. If more than 4 hex digits are entered, only the last 4 are used. The routine terminates when a valid delimiter is encountered. The delimiter is also returned as an output of the function. Illegal characters (not hex digits or delimiters) cause an error indication. If the first valid character encountered in the input stream is not a delimiter, GETHX will return with carry bit set to 1; otherwise, the carry bit is set to zero and the contents of BC are undefined.

11 - NAME: GETNM (Get hexadecimal numbers)
ADDRESS: 031E
INPUTS: Count of numbers in register C to find in input stream.
OUTPUTS: Numbers are found in reverse order on top of STACK such that last number is topmost.
CALLS: GETHX, HILO, ERROR
DESTROYS: A, B, C, D, E, H, L F/F's
DESCRIPTION: GETNM finds a specified count of numbers, between 1 and 3, inclusive, in the input stream from TTY, and returns their values on the stack. If two or more numbers are requested, then the first must be less than or equal to the second, otherwise, the first and the second number will be set equal. The last number requested must be terminated by a carriage return or an error indication will result.

12 - NAME: HILO (High Low)
ADDRESS: 0363
INPUTS: DE - 16 bit integer
HL - 16 bit integer
OUTPUTS: Carry Flag = 0 if HL < DE
= 1 if HL $\geq$ DE
CALLS: Nothing
DESTROYS: F/F's
DESCRIPTION: HILO compares the two 16 bit unsigned binary integers in HL and DE. The carry bit is set according to the result of the comparision.

13 - NAME: NMOUT (Hexadecimal Number OUTput)
ADDRESS: 038A
INPUTS: 8-bit integer in register A
OUTPUTS: None
CALLS: ECHO, PRVAL
DESTROYS: A, B, C, F/F's
DESCRIPTION: NMOUT converts the 8-bit, unsigned binary integer in the A register into two ASCII characters. These two characters are sent to the console at the current print position of the console.

14 - NAME: PRVAL
ADDRESS: 03A5
INPUTS: A hexadecimal integer, range 0 to F in register C
OUTPUTS: Equivalent ASCII character in register C
CALLS: Nothing
DESTROYS: B, C, H, L, F/F's
DESCRIPTION: PRVAL converts a number in the rage 0 to F to the corresponding ASCII character. PRVAL does not check the validity of its input argument.

15 - NAME: VALDG (Valid DiGit)
ADDRESS: 03AD
INPUTS: ASCII character to be tested in register C
OUTPUTS: Carry=1 if character represents valid hex digit
=0 otherwise
CALLS: Nothing
DESTROYS: A, F/F's
DESCRIPTION: VALDG test its input argument and sets the carry bit if the argument represents a valid hex digit and resets the carry bit if otherwise.

16 - NAME: VALDL (Valid DeLimeter)
ADDRESS: 03C8
INPUTS: ASCII character to be tested in register C
OUTPUTS: Carry = 1 if input argument is valid delimeter
= 0 otherwise
CALLS: Nothing
DESTROYS: A, F/F's
DESCRIPTION: VALDL tests its input argument and sets the carry bit if the argument represents a valid delimiter character (comma, carriage return, and space) and resets the carry bit if otherwise.

17 - NAME: FRET (Failure on RETurn)
ADDRESS: 03DB
INPUTS: None
OUTPUTS: Carry = 0
CALLS: Nothing
DESTROYS: Carry
DESCRIPTION: FRET resets the carry and then returns to the calling routine invoking FRET. This routine is jumped to by any routine that wishes to indicate failure on return.

18 - NAME: SRET (Success on RETurn)
ADDRESS: 03DE
INPUTS: None
OUTPUTS: Carry = 1
CALLS: Nothing
DESTROYS: Carry
DESCRIPTION: SRET sets the carry and then returns to the calling routine invoking FRET. This routine is jumped to by routines wishing to return success.

19 - Table of Address of Utility Routines

ROUTINE	START ADDRESS
DIGTB	0245
CI	0256
CO	0281
CNVBN	02A8
CROUT	02B1
DELAY	02B7
ECHO	02BE
ERROR	02D7
GETCH	02E2
GETHX	02E9
GETNM	031E
HILO	0363
NMOUT	038A
PRVAL	03A5
VALDG	03AD
VALDL	03C8
FRET	03D8
SRET	03DE

PART 2

Console Monitor

Upon the operator request the Console Monitor, command routines plus utility routines, is invoked by activating RST 5.5 interrupt line via switch SW3. The monitor starts executing after the interrupted routine has completed its assigned task. The logic operation of the monitor is shown in the flow diagram of Figure B-1. The monitor receives an input character from the TTY and attempts to locate this character in its command character table. When found, the routine corresponding to this character is selected from a table of command routines addresses, and the CPU control is transferred to this routine. If the character does not match any entries, control is passed to the error handler.

Many command routines can be added to the monitor by adding their corresponding characters to the command character table, and by adding the first instruction's address of each routine to the command routine addresses. The number of commands (NOD) which aids the monitor to scan through the command character table, should be adjusted to equal the number of characters in the table.

The present monitor can execute any of the three existing routines, ALTER, COPY and GO routines.

The ALTER command routine displays the memory address location entered by the operator and its contents on the CRT. Thus the contents of memory may be examined and modified. If it is desired to change the contents of this location, the operator enters the desired value in hexadecimal. Otherwise the contents are left intact. In both cases, the address is incremented automatically, to point to the next location in memory. To exit the ALTER routine any character except delimeters and hexadecimal digits can be

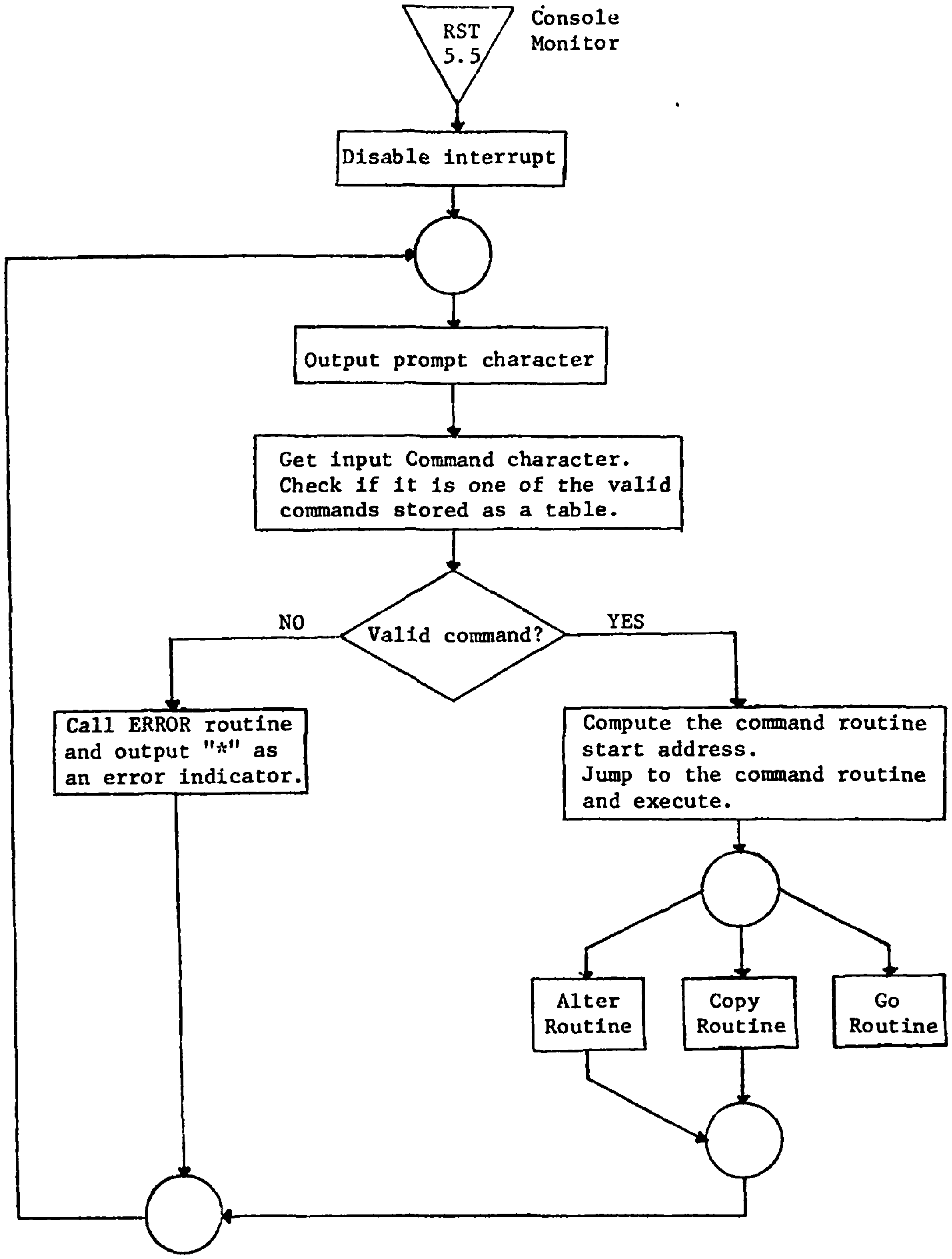

RST 5.5
Console Monitor
Disable interrupt
Output prompt character
Get input Command character. Check if it is one of the valid commands stored as a table.
Valid command?
NO
YES
Call ERROR routine and output "*" as an error indicator.
Compute the command routine start address. Jump to the command routine and execute.
Alter Routine
Copy Routine
Go Routine

Figure B-1. Console Monitor Flowchart.

entered.

The COPY command routine relocates the contents of a block of memory from one area of the memory space to another. The routine requires three operators separated by commas. Each operator is at most four hexadecimal digits wide. The first two operators represent the starting and the ending addresses of the block, while the third operator represents the destination address to which the first byte will be transferred. Once the transfer is completed, the CPU control is relinquished to the beginning of the monitor.

The GO command routine transfers the control of the microprocessor from the console monitor to the PCU-85 monitor. Similar to the COPY routine the GO routine requires three operators - Repertoire pointer, Program pointer and KBLOK pointer. These operators are used to update their respective values in the scratch pad. Then normal control operation of the QMF resumes with the new KBLOK control parameters.

Flowchart diagrams of ALTER, COPY and GO logic operation are shown in Figures B-2, B-3 and B-4.

Any illegal character entered after the command character will be recognized as an error. An asterisk, *, will be issued as error indicator. Then, control of the microprocessor is transferred to the start of the console monitor. Therefore, one of the commands can be repeated with the right parameters required by that command routine.

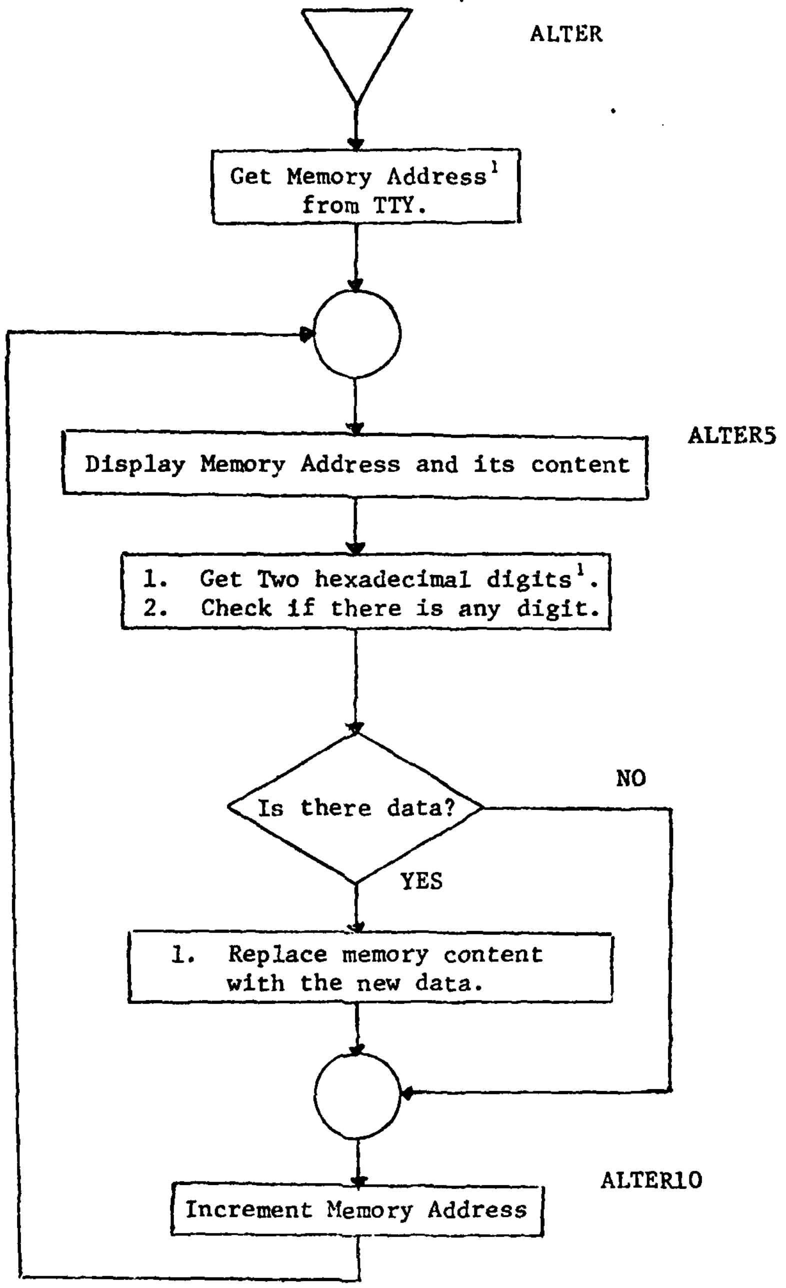

ALTER
Get Memory Address[1] from TTY.
ALTER5
Display Memory Address and its content
1. Get Two hexadecimal digits[1].
2. Check if there is any digit.
Is there data?
NO
YES
1. Replace memory content with the new data.
ALTER10
Increment Memory Address

Figure B-2. ALTER Flowchart.

[1]If a non-hexadecimal digit is entered, the 8085A control is transferred to the ERROR handler and then to the console monitor.

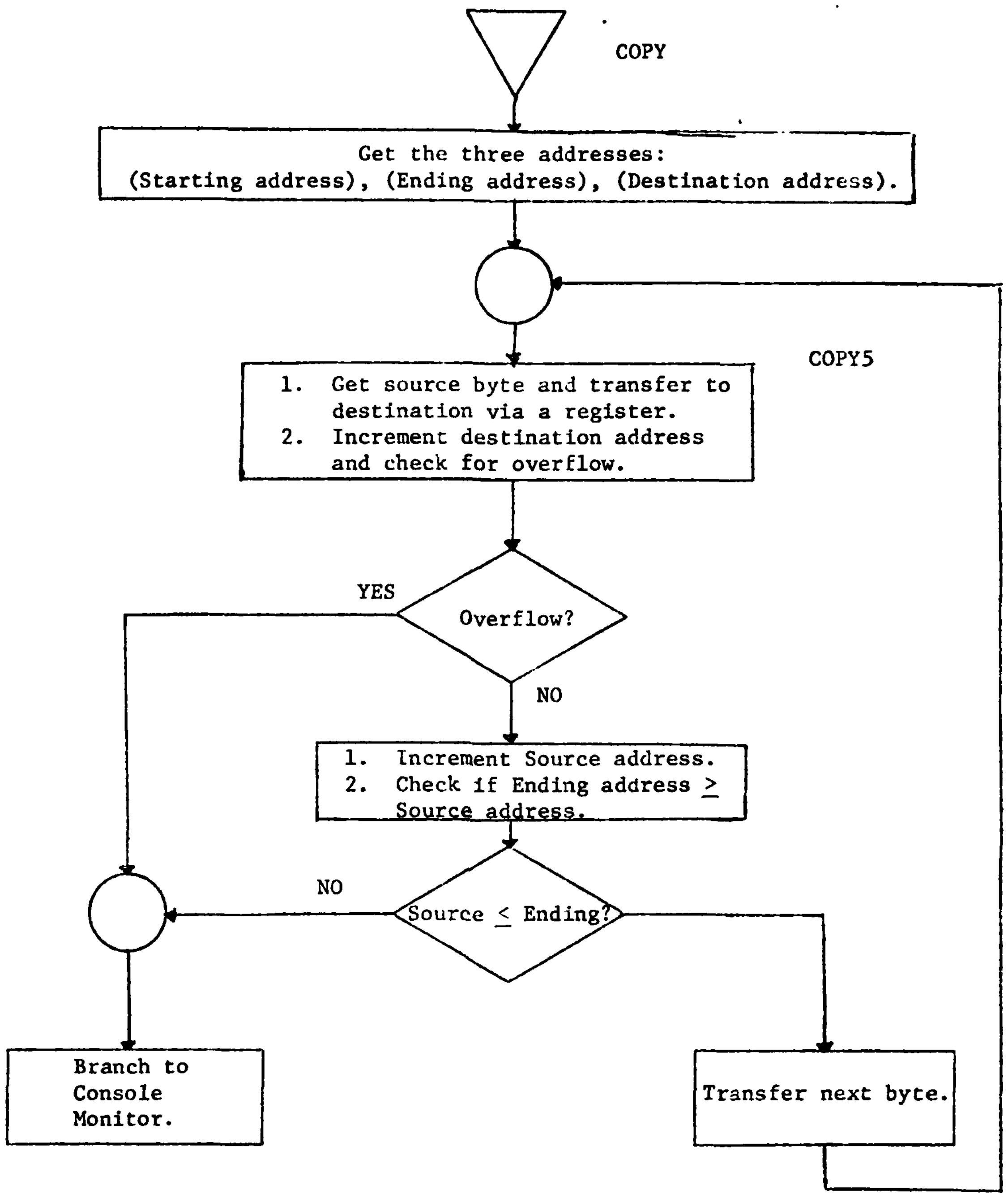

COPY
Get the three addresses: (Starting address), (Ending address), (Destination address).
COPY5
1. Get source byte and transfer to destination via a register.
2. Increment destination address and check for overflow.
Overflow?
YES
NO
1. Increment Source address.
2. Check if Ending address ≥ Source address.
Source ≤ Ending?
NO
Branch to Console Monitor.
Transfer next byte.

Figure B-3. COPY Flowchart.

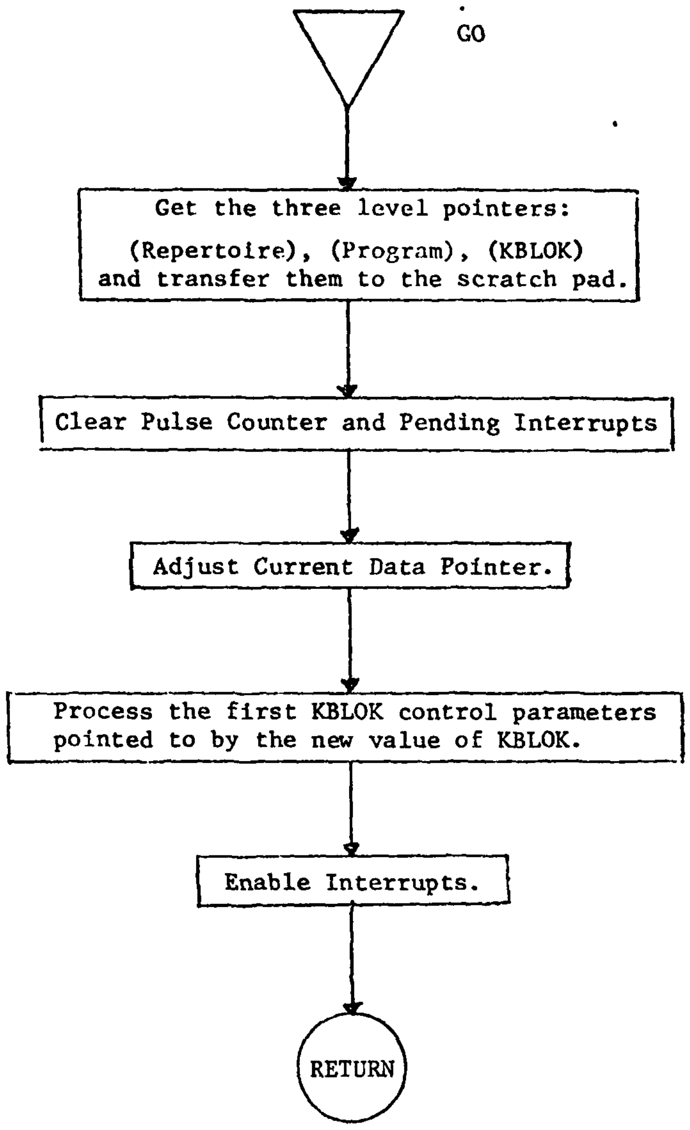

GO
Get the three level pointers:
(Repertoire), (Program), (KBLOK)
and transfer them to the scratch pad.
Clear Pulse Counter and Pending Interrupts
Adjust Current Data Pointer.
Process the first KBLOK control parameters
pointed to by the new value of KBLOK.
Enable Interrupts.
RETURN

Figure B-4. GO Flowchart.

INT: RST 5.5 (Task 4)

Responds to operator's request by suspending control of the mass spectrometer.

Task 4 can execute any of the following commands.

1. ALTER: A(address): Examine and optionally modify memory locations individually, starting at the address following the command letter A.
2. GO: G(REP POINTER), (PROG POINTER), (KBLOK POINTER): Start control of the mass spectrometer with KBLOK parameters defined by the three level pointers.
3. COPY: C(low address), (high address), (destination): Move the contents of memory between (low address) and (high address) inclusive to the area of RAM beginning at (destination).

Labels	Address	HEX Inst	DB or A16	Mnemonics	No. States	Comments
MON	00BC	F3		DI		Disable interrupts.
MON1	00BD	0E		MVI C, 3F		Load C with ASCII code for (?)
	00BE		3F			
	00BF	CD		CALL CO		Output prompt character (?) to
	00C0		81			screen.
	00C1		02			
	00C2	CD		CALL GETCH		Get a command character from
	00C3		E2			operator.
	00C4		02			
	00C5	CD		CALL ECHO		Echo command character to screen.
	00C6		BE			
	00C7		02			
	00C8	79		MOV A, C		Put character into "A"
	00C9	01		LXI B, NOD		Initialize BC to number of
	00CA		03			commands.
	00CB		00			
	00CC	21		LXI H, CCT		Initialize HL to point to the
	00CD		F0			command character table.
	00CE		07			
MON5	00CF	BE		CMP M		Compare table entry and character.
	00D0	CA		JZ MON10		Branch if equal.
	00D1		DB			
	00D2		00			
	00D3	23		INX H		Else, increment table pointer.
	00D4	0D		DCR C		Decrement loop count.
	00D5	C2		JNZ MON5		Branch if not at table end.
	00D6		CF			
	00D7		00			
	00D8	C3		JMP ERROR		Else, command character is
	00D9		D7			illegal.
	00DA		02			
MON10	00DB	21		LXI, CAT		Initialize HL to point the
	00DC		80			command address table.
	00DD		07			
	00DE	09		DAD B		Add the remainder of loop count.
	00DF	09		DAD B		Add again.
	00E0	7E		MOV A, M		Get LSP of address to A.
	00E1	23		JNX H		Point to next byte in table.
	00E2	66		MOV H, M		Get MSP of address to H.

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	00E3	6F		MOV L, A		Put LSP of address into 1.
	00E4	E9		PCHL		JMP to the command routine.
ALTER	00E5	CD		CALL GETHX		GET memory address from operator
	00E6		E9			and store in BC.
	00E7		02			
	00E8	C5		PUSH B		Load BC on stack.
	00E9	E1		POP H		Load HL from stack.
ALTER5	00EA	7C		MOV A, H		Move high byte address into A.
	00EB	CD		CALL NMOUT		Output A to TTY
	00EC		8A			
	00ED		03			
	00EE	7D		MOV A, L		Move low byte address into A.
	00EF	CD		CALL NMOUT		Output A to TTY.
	00F0		8A			
	00F1		03			
	00F2	0E		MVI C, 3A		Load C with ASCII code for (!)
	00F3		3A			
	00F4	CD		CALL ECHO		ECHO C to TTY.
	00F5		BE			
	00F6		02			
	00F7	7E		MOV A, M		Get memory byte into A.
	00F8	CD		CALL NMOUT		Output A to TTY.
	00F9		8A			
	00FA		03			
	00FB	0E		MVI C, 2D		Load C with ASCII Code for (-).
	00FC		2D			
	00FD	CD		CALL ECHO		ECHO C to TTY.
	00FE		BE			
	00FF		02			
	0100	CD		CALL GETHX		Get new value for memory location
	0101		E9			if any.
	0102		02			
	0103	D2		JNC ALTER10		If no value present, branch to
	0104		07			get next byte.
	0105		01			
	0106	71		MOV M, C		Else, store new byte into memory.
ALTER10	0107	23		INX H		Point to next byte in memory.
	0108	C3		JMP ALTER5		Branch to get next byte.
	0109		EA			
	010A		00			
GO	010B	0E		MVI C, 03		Get three level pointers from
	010C		03			TTY.
	010D	CD		CALL GETNM		
	010E		1E			
	010F		03			
	0110	E1		POP H		Load KBLOK pointer into HL.
	0111	22		SHLD KBLOK		Store into scratch pad.
	0112		FA			
	0113		20			
	0114	E1		POP H		Load PROG pointer into HL.

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	0115	22		SHLD PROGBUF		Store into scratch pad.
	0116		FC			
	0117		20			
	0118	E1		POP H		Load "REP pointer" into HL.
	0119	22		SHLD REPBUF		Store into scratch pad.
	011A		FE			
	011B		20			
	011C	D3		OUT PULSEO		Clear data counter.
	011D		80			
	011E	3E		MVI A, 18		Reset RST 7.5 interrupt.
	011F		18			
	0120	30		SIM		
	0121	2A		LHLD BDPOINTER		Adjust current data pointer to
	0122		F6			the value found in the data
	0123		20			pointer.
	0124	32		SHLD CDPOINTER		
	0125		F8			
	0126		20			
	0127	CD		CALL GKBLOK		Get breakpoint parameters and
	0128		07			output its parameters to QMF.
	0129		02			
	012A	FB		EI		Enable interrupt and return
	012B	C9		RET		to interrupted routine.
COPY	012C	0E		MVI, 03		Get three addresses from TTY.
	012D		03			
	012E	CD		CALL GETNM		
	012F		1E			
	0130		03			
	0131	C1		POP B		Load destination address into BL.
	0132	E1		POP H		Load Ending address into HL.
	0133	D1		POP D		Load Starting address into DE.
COPY5	0134	1A		LDAX D		Get source byte.
	0135	02		SDAX B		Move byte to destination.
	0136	03		INX B		Increment destination address.
	0137	78		MOV A, B		Test for destination address overflow.
	0138	B1		ORA C		
	0139	CA		JZ MON1		If no, terminate command.
	013A		BD			
	013B		00			
	013C	13		INX D		Else, increment source address.
	013D	CD		CALL HILO		Check if ending address > source
	013E		63			address.
	013F		03			
	0140	D2		JNC MON1		If not, command is done.
	0141		BD			
	0142		00			
	0143	C3		JMP COPY5		Else, move the next byte.
	0144					
	0145		01			
	0146					

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
CAT	0780		00	DUMMY		Table of addresses of command
	0781		00			routines.
	0782		E5	ALTER		
	0783		00			
	0784		0B	GO		
	0785		01			
	0786		2C	COPY		
	0787		01			
CCT	07F0		43	C		Table of valid command characters.
	07F1		47	G		
	07F2		41	A		

These utility subroutines are used by the TTY monitor to execute the commands desired by the operator. A description of each subroutine is summarized in part one at the start of appendix B.

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
CI	0256	F3		DI		
	0257	D5		PUSH D		Save DE
CI05	0258	20		RIM		Get Input Bit
	0259	17		RAL		Into Carry F/F
	025A	DA		JC CI05		Branch If No Start Bit
	025B		58			
	025C		02			
	025D	11		LXI D, WAIT		Wait until middle bit
	025E		05			
	025F		00			
	0260	CD		CALL DELAY		
	0261		B7			
	0262		02			
	0263	C5		PUSH B		Sa e BC
	0264	01		LXI B, 8		Initialize B to zero, and C to
	0265		08			number of data bits.
	0266		00			
CI10	0267	"		LXI D, IB TIM		Wait until middle of next bit
	0268		0A			
	0269		00			
	026A	CD		CALL DELAY		
	026B		B7			
	026C		02			
	026D	20		RIM		Get the bit
	026E	17		RAL		Into carry
	026F	78		MOV A, B		Get partial result
	0270	IF		RAR		Shist in next data bit
	0271	47		MOV B, A		Replace result
	0272	OD		DCR C		Decrement counts of bits to GD
	0273	C2		JNZ CI10		Branch IF more to go
	0274		67			
	0275		02			
	0276	11		LXI D, IBTIM		Else, want to wait out
	0277		0A			stop bit
	0278		00			
	0279	CD		CALL DELAY		
	027A		B7			
	027B		02			
	027C	78		MOV A, B		Get result

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	027D	C1		POP B		Restore saved registers
	027E	D1		POP D		
	027F	FB		EI		
	0280	C9		RET		Return
CO	0281	F3		DI		
	0282	C5		PUSH B		Save BC
	0283	D5		PUSH D		Save DE
	0284	3E		MVI A, STRT	S	Start bit mask
	0285		CD			
	0286	06		MVI B, 8		B will count bits to send
	0287		08			
C005	0288	30		SIM		Send A bit
	0289	11		LXI D, OBTIM		Wait for TTY to handle it
	028A		OA			
	028B		00			
	028C	CD		CALL DELAY		
	028D	B	B7			
	028E		02			
	028F	79		MOV A, C		Pick up bits left to send
	0290	IF		RAR		Low order bit to carry
	0291	4Ff		MOV C, A		Put rest back
	0292	3E		MVI A, SSTRT		Shifted enable bit
	0293		80			
	0294	IF		RAR		Shift in data bit
	0295	EE		XRI 80		Complement data bit
	0296		80			
	0297	05		DCR B		DEC Count
	0298	F2		JP C005		Send if more bits need to be sent
	0299		88			
	029A		02			
	029B	3E		MVI A, STOPB		Else, send stop bit
	029C		40			
	029D	30		SIM		
	029E	"		LXI D, TIM2		Wait out parity bit
	029F		14			
	02A0		00			
	02A1	CD		CALL DELAY		
	02A2		B7			
	02A3		02			
	02A4	D1		POP D		Restore saved registers
	02A5	C1		POP B		
	02A6	F3		EI		
	02A7	C9		RET		Return
CNVBN	02A8	79		MOV A, C		
	02A9	D6		SUI "0"		Subtract code for "0" from argument
	02AA		30			
	02AB	FE		CPI "10"		Want to test for result of 0 to 9
	02AC		0A			
	02AD	F8		RM		If so, then all done
	02AE	D6		SUI '7'		Else, result between 17 and

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	02B0	C9		RET		-Return
CROUT	02B1	0E		MVI C, CR		- Put ASCII CR IN C
	02B2		0D			
	02B3	CD		CALL ECHO		-Output.CR
	02B4		BE			
	02B5		02			
	02B6	C9		RET		-Return
DELAY	02B7	1B		DCX D		-Decrement Input Argument
	02B8	7A		MOV A, D		
	02B9	B7		ORA E		
	02BA	C2		JNZ DELAY		-If argument not '0', keep going
	02BB		F1			
	02BC		05			
	02BD	C9		RET		
ECHO	02BE	41		MOV B, C		-Save argument
	02BF	3E		MVI A, ESC		
	02C0		1B			
	02C1	B8		CMP B		-See if echoing an escape character
	02C2	C2		JNZ ECHO5		-No - Branch
	02C3		C7			-Yes - ECHO A'$' character
	02C4		02			
	02C5	0E		MVI C, '$'		
	02C6		24			
ECHO5	02C7	CD		CALL CO		-Do output through CO
	02C8		81			
	02C9		02			
	02CA	3E		MVI A, CR		
	02CB		0D			
	02CC	B8		JMP B		-See if character was carriage return
	02CD	C2		JNZ ECH10		-No - No need to take special action
	02CE		D5			
	02CF		02			
	02D0	0E		MVI C, LF		-Yes - Want to ECHO line feed, too.
	02D1		0A			
	02D2	CD		CALL CO		
	02D3		81			
	02D4		02			
ECH10	02D5	48		MOV C, B		-Restore argument
	02D6	C9		RET		
ERROR	02D7	0E		MVI C, '*'		-Send '*' to CRT
	02D8		2A			
	02D9	CD		CALL ECHO		
	02DA		BE			
	02DB		02			
	02DC	CD		CALL CROUT		-Skip to beginning of next line
	02DD		B1			
	02DE		02			
	02DF	C3		JMP MONO1		-Get another command through monitor
	02E0		BD			
	02E1		00			

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
GETCH	02E2	CD		CALL CI		-Get character from terminal
	02E3		56			
	02E4		02			
	02E5	E6		ANI PRTYO		-Turn off parity bit in case
	02E6		7F			set by console.
	02E7	4F		MOV C, A		-Put value in C and return.
	02E8	C9		RET		
GETHX	02E9	E5		PUSH H		-Save HL
	02EA	21		LXI H, 0		-Initialize result.
	02EB		00			
	02EC		00			
	02ED	1E		MVI E, 0		-Initialize digit flag to false
	02EE		00			
GHX05	02EF	CD		CALL GETCH		-Get a character
	02F0		E2			
	02F1		02			
	02F2	4F		MOV C, A		
	02F3	CD		CALL ECHO		-Echo the character
	02F4		BE			
	02F5		02			
	02F6	CD		CALL VALDL		-See if delimiter
	02F7		C8			
	02F8		03			
	02F9	D2		JNC GHX10		-No, branch
	02FA		08			
	02FB		03			
	02FC	51		MOV D, C		-Yes, all done, return delimiter
	02FD	E5		PUSH H		
	02FE	C1		POP B		-Move result to BC
	02FF	E1		POP H		-Restore HL
	0300	7B		MOV A, E		-Get Flag
	0301	B7		ORA A		-Set F/F's
	0302	C2		JNZ SRET		-If flag not-zero, a number has
	0303		DE			been found
	0304		03			
	0305	CA		NZ FRET		-Else, delimiter was first character
	0306		DB			
	0307		03			
GHX10	0308	CD		CALL VALDG		-If not delimiter, see if digit
	0309		AD			
	030A		03			
	030B	D2		JNC ERROR		-Error if not a valid digit, either
	030C		D7			
	030D		02			
	030E	CD		CALL CNVBN		-Convert digit to its binary value
	030F		A8			
	0310		02			
	0311	1E		MVI E, FF		-Set digit flag not-zero
	0312		FF			
	0313	29		DAD H		-Multiply HL by 2
	0314	29		DAD H		-Multiply HL by 4

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	0315	29		DAD H		-Multiply HL by 8
	0316	29		DAD H		-Multiply HL by 16
	0317	06		MVI B, D		-Clear upper 8 bits of BC
	0318		00			
	0319	4F		MOV C, A		-Binary value of character into C
	031A	09		DAD B		-Add this value to partial result
	031B	C3		JMP GHX05		-Get next character
	031C		EF			
	031D		02			
GETNM	031E	2E		MVI L, 3		-Put maximum argument count into L
	031F		03			
	0320	79		MOV A, C		-Get the actual argument count
	0321	E6		ANI 3		-Force to maximum of 3
	0322		03			
	0323	C8		RZ		-If zero, don't bother to do anything
	0324	67		MOV H, A		-Else, put actual count into H
GNM05	0325	CD		CALL GETHX		-Get a number from input stream
	0326		E9			
	0327		02			
	0328	D2		JNC ERROR		-Error if not there.
	0329		D7			
	032A		02			
	032B	C5		PUSH B		-Else, save number on stack
	032C	2D		DCR L		-Decrement maximum argument count
	032D	25		DCR H		-Decrement actual argument count
	032E	CA		JZ GNM10		-Branch if no more numbers wanted
	032F		3A			
	0330		03			
	0331	7A		MOV A, D		-Else, get number terminator to A
	0332	FE		CPI CR		-See if carriage return
	0333		0D			
	0334	CA		JZ ERROR		-Error if so
	0335		D7			
	0336		02			
	0337	C3		JMP GNM05		-Else, process next number
	0338		25			
	0339		03			
GNM10	033A	7A		MOV A, D		-When count 0, check last terminator
	033B	FE		CPI CR		
	033C		0D			
	033D	C2		JNZ ERROR		-Error if not carriage return
	033E		D7			
	033F		02			
	0340	01		LXI B, FFFF		-BC gets largest number
	0341		FF			
	0342		FF			
	0343	7D		MOV A, L		-Get remainder of maximum argument coun
	0344	B7		ORA A		-Check for zero
	0345	CA		JZ GNM20		-If yes, 3 numbers were input
	0346		4D			
	0347		03			

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
GNM15	0348	C5		PUSH B		-If not, fill remaining arguments
	0349	2D		DCR L		with FFFF
	034A	C2		JNZ GNM15		
	034B		48			
	034C		03			
GNM20	034D	C1		POP B		-Get the 3 arguments out
	034E	D1		POP D		into the CPU registers
	034F	E1		POP H		
	0350	CD		CALL HILO		-See if first > second
	0351		63			
	0352		03			
	0353	D2		JNC GNM25		-No, branch
	0354		58			
	0355		03			
	0356	54		MOV D, H		-Yes, make second equal to the first
	0357	5D		MOV E, L		
GNM25	0358	E3		XTHL		-Put first on stack, get return address
	0359	D5		PUSH D		-Put second on stack
	035A	C5		PUSH B		-Put third on stack
	035B	E5		PUSH H		-Put return address on stack
GNM30	035C	3D		DCR A		-Decrement residual count
	035D	F8		RM		-If negative, proper results on stack
	035E	E1		POP H		-Else, get return address
	035F	E3		XTHL		-Replace top result with return address
	0360	C3		JMP GNM30		-Try again
	0361		5C			
	0362		03			
HILO	0363	C5		PUSH B		-Save BC
	0364	47		[illegible]		-Save A in B register
	0365	E5		PUSH H		-Save HL
	0366	[illegible]		[illegible]		-Check for DE = 0000
	0367	B3		ORA E		
	0368	CA		JZ HILO5		-If it is, comparison is done
	0369					
	036A					
	[illegible]	23		[illegible]		-Increment HL
	[illegible]	7C		MOV A, H		-Test for zero result
	[illegible]	B5		[illegible]		
	[illegible]	CA		JZ HILO5		-If so, HL must have contained FFFF
	[illegible]					
	[illegible]					
	[illegible]	E1		POP H		-If not, restore original HL
	0372	D5		PUSH D		-Save DE
	0373	3E		MVI A, FF		-Take 2's complement of DE
	0374		FF			
	0375	AA		XRA D		
	0376	57		MOV D, A		
	0377	3E		MVI A, FF		
	0378		FF			
	0379	AB		XRA E		
	037A	5F		MOV E, A		

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	037B	13		INX D		
	037C	7D		MOV A, L		-Add HL and DE
	037D	83		ADD E		
	037E	7C		MOV A, H		
	037F	8A		ADC D		-This operation sets carry properly
	0380	01		POP D		-Restore original DE contents
	0381	78		MOV A, B		-Restore original contents of A
	0382	C1		POP B		-Restore original contents of BC
	0383	C9		RET		-Return with carry set as required
HILO5	0384	E1		POP H		-If HL = FFFF, then carry can only
	0385	78		MOV A, B		be set to 1
	0386	C1		POP B		-Restore original contents of BC
	0387	C3		JMP SRET		-Set carry and return
	0388		DE			
	0389		03			
NMOUT	038A	E5		PUSH H		-Save HL (to be destroyed by PRVAL)
	038B	F5		PUSH PSW		-Save argument
	038C	0F		RRC		-Get upper 4 bits to low 4 bit
	038D	0F		RRC		positions
	038E	0F		RRC		
	038F	0F		RRC		
	0390	E6		ANI HCHAR		-Mask out upper 4 bits
	0391		0F			
	0392	4F		MOV C, A		
	0393	CD		CALL PRVAL		-Convert lower 4 bits to ASCII
	0394		A5			
	0395		03			
	0396	CD		CALL ECHO		-Send to terminal
	0397		BE			
	0398		02			
	0399	F1		POP PSW		-Get back argument
	039A	E6		ANI HCHAR		-Mask out upper 4 bits
	039B		0F			
	039C	4F		MOV C, A		
	039D	CD		CALL PRVAL		-Convert lower 4 bits to ASCII
	039E		A5			
	039F		03			
	03A0	CD		CALL ECHO		-Send to terminal
	03A1		BE			
	03A2		02			
	03A3	E1		POP H		-Restore saved value of HL
	03A4	C9		RET		-Return
PRVAL	03A5	21		LXI H, DIGTB		-Address of ASCII digit table
	03A6		45			
	03A7		02			
	03A8	06		MVI B, 0		-Clear high order bits of BC
	03A9		00			
	03AA	09		DAD B		-Add digit value to HL address
	03A3	4E		MOV C, M		-Fetch ASCII digit from table
	03AC	C9		RET		-Return

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
VALDG	03AD	79		MOV A, C		-Test character against ASCII '0'
	03AE	FE		CPI '0'		
	03AF		30			
	03B0	FA		JMP FRET		-If ASCII code less, cannot be valid
	03B1		DB			digit
	03B2		03			
	03B3	FE		CPI '7'		-Else, see if in range '0' - '9'
	03B4		39			
	03B5	FA		JM SRET		-Code between '0' and '9'
	03B6		DE			
	03B7		03			
	03B8	CA		JZ SRET		-Code equal '9'
	03B9		DE			
	03BA		03			
	03BB	FE		CPI 'A'		-Not a digit, try for a hex letter
	03BC		41			
	03BD	FA		JM FRET		-No, code is between '9' and 'A'
	03BE		DB			
	03BF		03			
	03C0	FE		CPI 'A'		-See if code is between 'A' and 'G'
	03C1		47			
	03C2	F2		JP FRET		-No, code is greater than 'F'
	03C3		DB			
	03C4		03			
	03C5	C3		JMP SRET		-Yes, code is between 'A' and 'F'
	03C6		DE			inclusive
	03C7		03			
VALDL	03C8	79		MOV A, C		-Check for comma
	03C9	FE		CPI ','		
	03CA		2C			
	03CB	CA		JZ SRET		
	03CC		DE			
	03CD		03			
	03CE	FE		CPI CR		-Check for carriage return
	03CF		OD			
	03D0	CA		JZ SRET		
	03D1		DE			
	03D2		03			
	03D3	FE		CPI ' '		-Check for space
	03D4		20			
	03D5	CA		JZ SRET		
	03D6		DE			
	03D7		03			
	03D8	C3		JMP FRET		-Error if not of the above
	03D9		DB			
	03DA		03			
FRET	03DB	37		STC		-First set carry true
	03DC	3F		CMC		-Then complement it to make it false
	03DD	C9		RET		-Return
SRET	03DE	37		STC		-Set carry true
	03DF	C9		RET		-Return

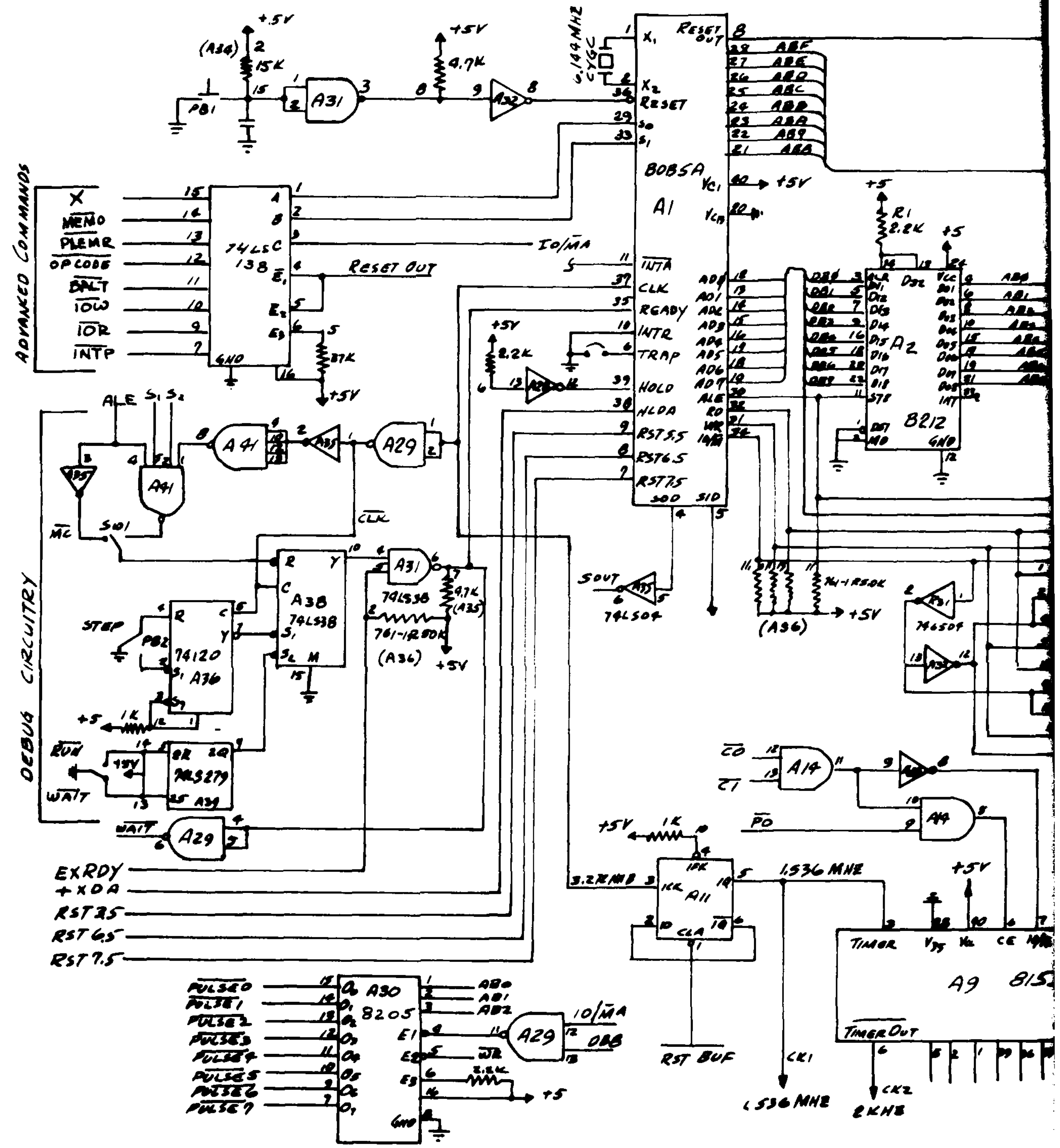
MESSAGE (FILE HEADER):

File header is a message to be displayed on CRT screen at the start of each data transfer from buffer memory to CRT. The following is a table of ASCII characters that spell out the message via Task 3 (RST 6.5 interrupt service) routine.

Labels	Address	HEX Inst	A16	Mnemonics	No. States	Comments
MPOINT1	0700		OD	CR		-New line
	0701		OA	LF		
	0702		09	TAB		
	0703		54	T		-This
	0704		48	H		
	0705		49	I		
	0706		53	S		
	0707		20	SP		-Space
	0708		49	I		-Is
	0709		53	S		
	070A		20	SP		
	070B		50	P		-PCU
	070C		43	C		
	070D		55	U		
	070E		20	SP		
	070F		46	F		-First
	0710		49	I		
	0711		52	R		
	0712		53	S		
	0713		54	T		
	0714		20	SP		
	0715		52	R		-Run.
	0716		55	U		
	0717		4E	N		
	0718		2E	.		
	0719		OA	LF		-New line
	071A		OD	CR		
	071B		44	D		-Date: 3/8/1978.
	071C		41	A		
	071D		54	T		
	071E		45	E		
	071F		3A	:		
	0720		33	3		
	0721		2F	/		
	0722		38	8		
	0723		2F	/		
	0724		31	1		
	0725		39	9		
	0726		37	7		

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	0727		38	8		
	0728		2E	.		
	0729		0A	LF		-New line
	072A		0D	CR		
	072B		46	F		-File #:
	072C		49	I		
	072D		4C	L		
	072E		45	E		
	072F		20	SP		
	0730		23	#		
	0731		3A	:		
	0732		00	NUL		-End of message mark
MPOINT2	0733		0D	CR		-New line
	0734		0A	LF		
	0735		52	R		-Repetoire
	0736		45	E		
	0737		50	P		
	0738		45	E		
	0739		54	T		
	073A		4F	O		
	073B		49	I		
	073C		52	R		
	073D		45	E		
	073E		20	SP		-Space
	073F		50	P		-Program
	0740		52	R		
	0741		4F	O		
	0742		47	G		
	0743		52	R		
	0744		41	A		
	0745		4D	M		
	0746		20	SP		-Space
	0747		4B	K		-KBLOK
	0748		42	B		
	0749		4C	L		
	074A		4F	O		
	074B		4B	K		
	074C		0A	LF		-New line
	074D		0D	CR		
	074E		20	SP		-Space
	074F		00	NUL		-End of message mark
MPOINT3	0750		0A	LF		-New line
	0751		0D	CR		
	0752		0A	LF		-New line
	0753		0D	CR		
	0754		20	SP		-Space
	0755		44	D		-Data
	0756		41	A		
	0757		45	T		
	0758		41	A		

Labels	Address	HEX Inst	D8 A16	Mnemonics	No. States	Comments
	0759		20	SP		
	075A		54	T		-Time
	075B		49	I		
	075C		4D	M		
	075D		45	E		
	075E		20	SP		
	075F		44	D		-Data
	0760		41	A		
	0761		54	T		
	0762		41	A		
	0763		20	SP		
	0764		54	T		-Time
	0765		49	I		
	0766		4D	M		
	0767		45	E		
	0768		20	SP		
	0769		44	D		-Data
	076A		41	A		
	076B		54	T		
	076C		41	A		
	076D		20	SP		
	076E		54	T		-Time
	076F		49	I		
	0770		4D	M		
	0771		45	E		
	0772		20	SP		
	0773		44	D		-Data
	0774		41	A		
	0775		54	T		
	0776		41	A		
	0777		20	SP		
	0778		54	T		-Time
	0779		49	I		
	077A		4D	M		
	077B		45	E		
	077C		20	SP		
	077D		0A	LF		-New line
	077E		0D	CR		
	077F		00	NUL		-End of message mark



ADVANCED COMMANDS
DEBUG CIRCUITRY
8085A
A1
74LS138
8212
A2
74120
A36
74LS38
A38
74LS279
A39
8205
A30
A31
A29
A41
A11
A14
A9
Timer Out
+5V
6.144 MHZ
RESET OUT
IO/MA
EXRDY
+XDA
RST 5.5
RST 6.5
RST 7.5
STEP
RUN
WAIT
SOUT
PULSE0
PULSE1
PULSE2
PULSE3
PULSE4
PULSE5
PULSE6
PULSE7
1.536 MHZ
3.2 MHZ
RST BUF
CK1
CK2
2KHZ

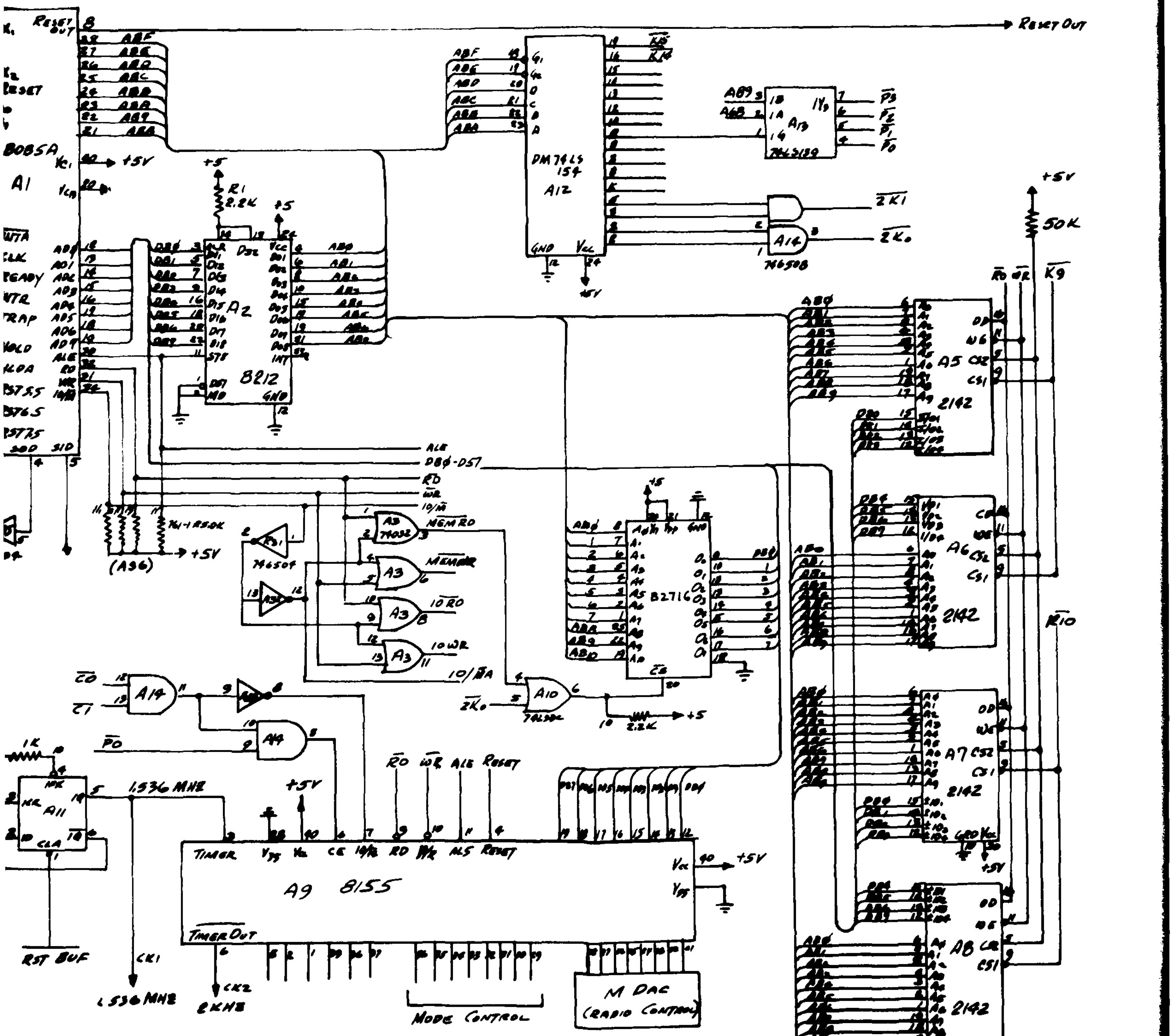


Figure B-5 8085 CPU with scratch-pad memory

2

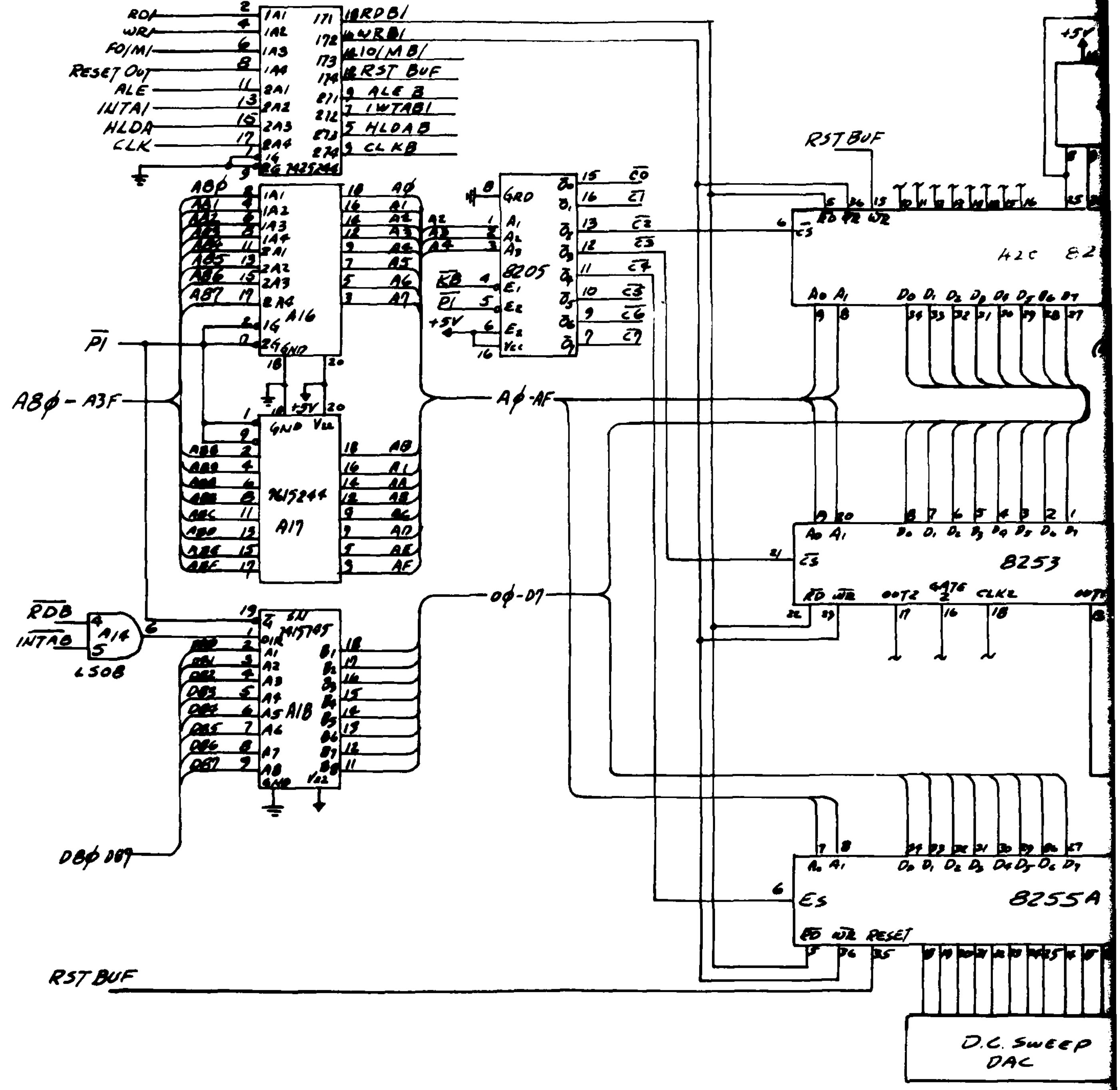
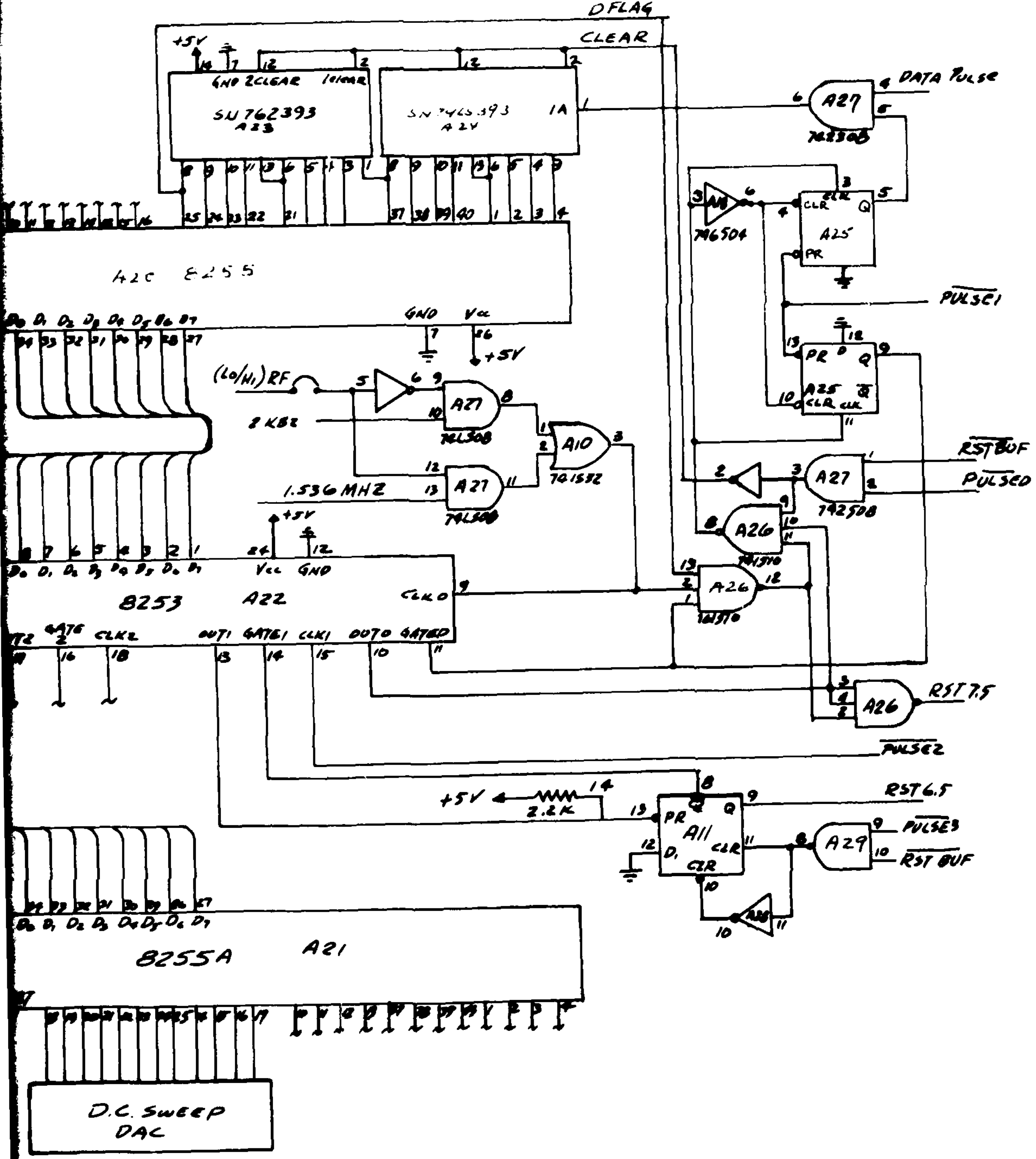


Figure B-6 I/O devices with buffered DATA BUS and ADDRESS BUS


DFLAG
CLEAR
+5V
SN 74LS393 A23
SN 74LS393 A24
DATA PULSE
A27
74LS08
A26 8255
A25
74LS04
PULSE1
RST BUF
PULSED
(LO/HI) RF
2 KHZ
1.536 MHZ
A10
74LS32
8253 A22
CLK0
OUT1 GATE1 CLK1 OUT0 GATE0
RST 7.5
PULSE2
RST 6.5
2.2K
A11
PULSE3
8255A A21
D.C. SWEEP DAC

2

APPENDIX C

Appendix C contains the PCU-85 system schematic diagram.

REFERENCES

(1) Intel, "MCS-85 User's Manual." January 1978.

(2) Texas Instruments Incorporated, "The TTL Data Book for Design Engineers," Second Edition.

(3) Texas Instruments Incorporated, "Low Power Schottky Bus Drivers and Transceivers. The LS240 Series Engineering Guide." 1977.

(4) Advanced Micro Devices, Inc., "The 8080A/9080A MOS Microprocessor Handbook." 1977.

(5) E. Arijs and D. Nevejans, "Programmable Control Unit for a Balloon Borne Quadrupole Mass Spectrometer." Rev. Sci. Instrum., Vol. 46 No. 8, August 1975.

(6) William McFadden, "Technique of Combined Gas Chromatography/Mass Spectrometry: Applications in Organic Analysis." John Wiley and Sons. 1973; Chapter 2.

(7) [1]W. Paul, H.P. Reinhard, and Von Sahn, A. Phys. 152, 143. (1958).

(8) P.H. Dawson, N.R. Whetten, Mass Spectroscopy Using RF Quadrupole Fields; Chap. III, pp 60-185. Advances in Electronics and Electron Physics, Vol. 27 (1969), Academic Press, N.Y. & London.

RELATED CONTRACTS AND PUBLICATIONS

F19628-74-C-0042	1 September 1973 through 31 July 1976
F19628-76-C-0256	1 August 1976 through 31 October 1978
F19628-78-C-0218	18 September 1978 through present

Sukys, R. and Goldberg, S. (1974),"Control Circuits for a Rocket Payload Neutralization and Other Topics", AFCRL-TR-74-0580.

Sukys,R., Rockefort, J.S. and Goldberg, S. (1975), "Bias and Signal Processing Circuits for a Mass Spectrometer in the Project EXCEDE: SWIR Experiment," AFGL-TR-76-200.

Rochefort, J. S. and Sukys, R. (1976), "Instrumentation Systems for Mass Spectrometers", AFGL-TR-76-200.

Rochefort, J. S. and Sukys, R. (1978), "A Digital Control Unit for a Rocket Borne Quadrupole Mass Spectrometer, AFGL-TR-78-0106.

Palasek, T. (1979), "An RF Oscillator for Rocket-Borne and BAlloon-Borne Quadrapole Mass Spectrometer, AFGL-TR-

PERSONNEL

A list of the engineers who contributed to the work reported is given below:

J. Spencer Rochefort, Professor of Electrical Engineering, Department Chairman, Principal Investigator.

Raimundas Sukys, Senior Reasearch Associate, Engineer.

Printed by
United States Air Force
Hanscom AFB, Mass. 01731