Design and development of an interface board between a minicomputer and a CDC printer with a memory buffer and a programmable vertical format throw

by

P T F ORMAN

Thesis submitted in partial compliance with the requirements for the

Masters Diploma in Technology
Electrical Engineering (Light Current)

at the Cape Technikon

BROWN DAVIS and McCORQUODALE
Cape Town
South Africa
June 1988

ACKNOWLEDGEMENTS

Without the guidance and assistance of the following people,this thesis may have never materialised.

1 Brown Davis and McCorquodale,in particular Mr Bill Crotty

2 Mr D Bensch of the Cape Technikon

3 Mr J Human of the Cape Technikon

Abreviations

CDC	Control Data Corporation
CPU	Z80 Central Processing Unit
DTL	Diode Transistor Logic
I/O	Input / Output
MICR	Magnetic Ink Character Recognition
PC	Personal Computer
PCB	Printed Circuit Board
RAM	Random Access Memory
ROM	Read Only Memory
TTL	Transistor Transistor Logic
VFU	Vertical Format Unit
VFU CPU	Vertical Format Unit Computer

Contents

Contents (continued) Page

List of Figures

List of Tables

Introduction

1.1 Preface

Brown Davis and McCorquodale is one of the major suppliers of cheques to the banking industry. To produce these cheques they use a number of different print systems, one of which comprises of a minicomputer, an industry standard tape deck and two printers, a Diablo daisywheel and a Control Data Corporation (CDC) printer which was extensively modified to cater for the requirements of the cheque printing industry.

The CDC printer is used to print the code line on the cheques using magnetic ink. After each line is printed the computer sends a form feed command which causes the printer to throw paper. This throw is controlled by a paper tape, known as a Vertical Format Unit tape, or rather a VFU tape. This tape has holes punched into it at specific places which determine the amount of paper throw also known as vertical feed. The holes are sensed by brushes which are pulled up to 5 volt when they pass over a hole and touch a roller connected to the 5 volt line.

This system, being of an electro-mechanical nature, is prone to faults and causes much down time due to mechanical wear on the brushes and dirt on the roller. This means that the brushes have to be adjusted and therefore also means that the timing has to be readjusted each time. The timing relationships are discussed in Section 2.8

1.2 Motivation for this project

To save time and to speed up the printer it was decided to design and build a microprocessor controlled VFU with the added feature of a one line buffer. This would allow the computer to program the VFU when the print program is selected as well as allow data to be down loaded to the printer during paper movement. The advantages of this system are:

1 This allows the computer a bit more time to prepare the next line, instead of waiting for the printer to be ready, and then only preparing the line

2 This saves time as the operator does not have to search for and change the VFU tape between each job with a different throw.

3 This saves down time as the time required for maintenance will be minimal (no mechanical parts) and there are no timing constraints, therefore no setting up of the timing relationships

Background

How the CDC printers VFU works.

2.1 Tractors

The tractors are the devices that physically hold the paper and pull it through the printer. These tractors comprise of four separate units, top and bottom, left and right. The tractors are driven by two shafts, top and bottom, connected to a timing belt. The timing belt is driven by a pulley connected to the drive and brake clutch assembly.

2.2 Drive pulley

The drive pulley consists of a spring loaded knob (Vernier Adjustment Knob) connected to the shaft and driven by a very fine gear. See Figure 2.1. This allows an operator to do vertical fine adjustments to the print position.

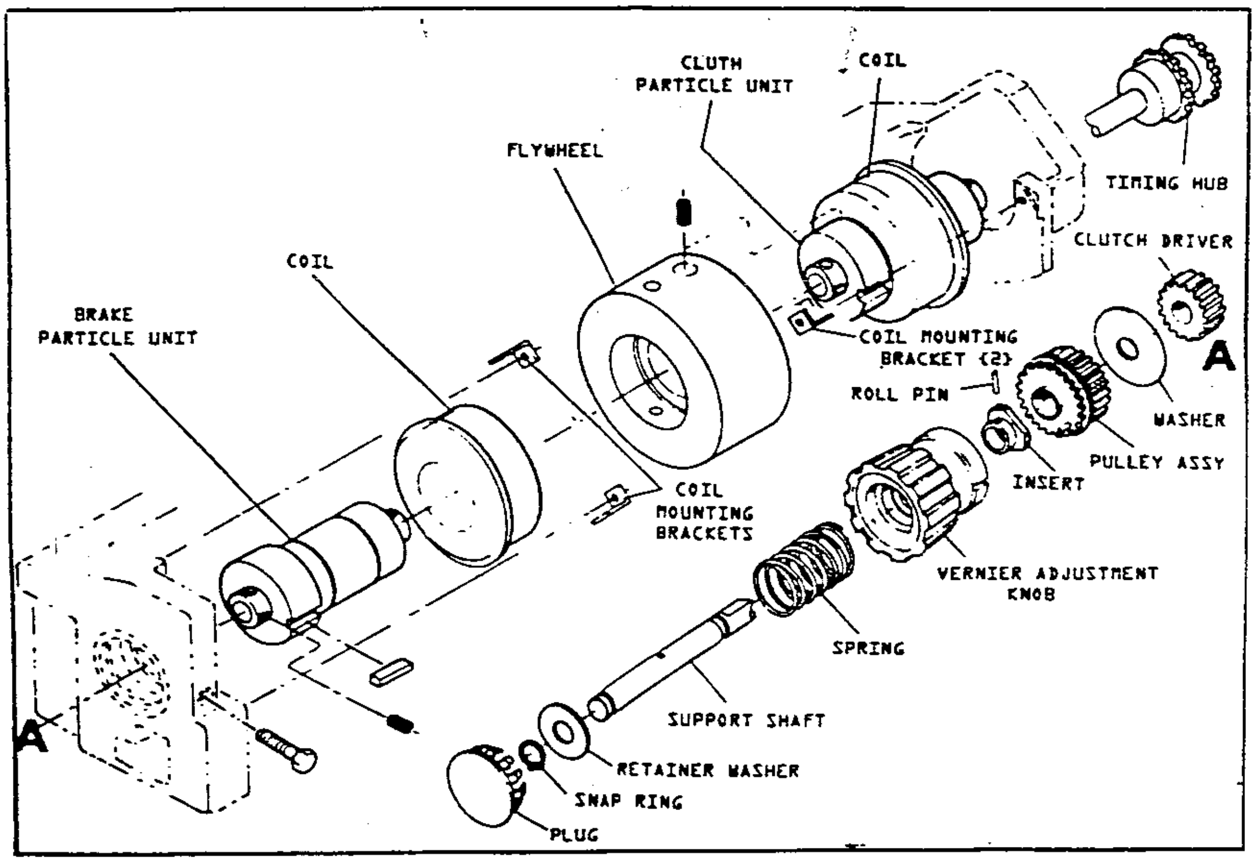


Figure 2.1 Paper Drive Unit Assembly

2.3 Shaft

The drive shaft carries the drive and brake clutches and drives the paper tractors via a timing belt. The brake clutch is fixed to the chassis of the printer while the drive clutch is fixed to a flywheel. The flywheel is driven by an AC motor at a constant speed (Figure 2.2).The shaft also drives a sprocket roller via a smaller timing belt (Clutch Driver) as well as two timing gears, one set at eighths of an inch, the other at sixths.(Timing Hub)

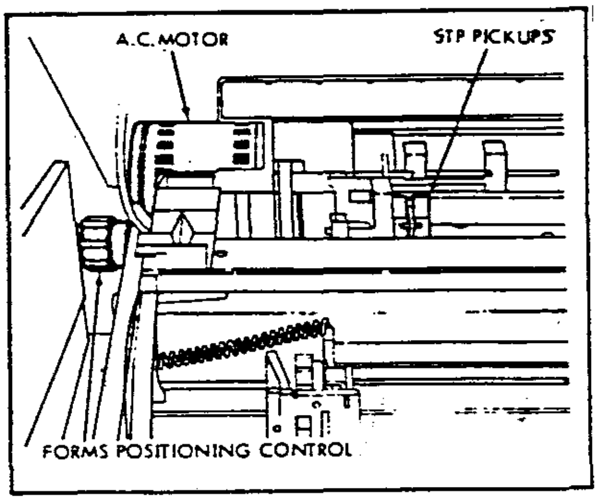


Figure 2.2 Forms Advance Assembly

2.4 Clutches

Both the brake and the drive clutch are identical. They are particle clutches and are used to start and stop paper movement Refer to Figure 2.1

2.5 Sprocket roller

This roller (Figure 2.4) is responsible for pulling the paper tape with the punched holes through the vertical format unit. The roller has a one way bearing and can only be driven in the direction of normal paper movement. The sprockets are set at one sixth of an inch. The paper tape is tensioned by a fixed roller, adjustable by the operator. The roller is marked every half inch by a red dot.(Figure 2.3)This is to allow operators to line up the paper tape top of form hole with a coincident pulse.

A coincident pulse is regarded as the pulse generated whenever the six and eight lines per inch pulse coincides. This occurs every half inch, ie every three pulses on six lines per inch and every four pulses at eight lines per inch. This relationship can be seen in figure 2.7.

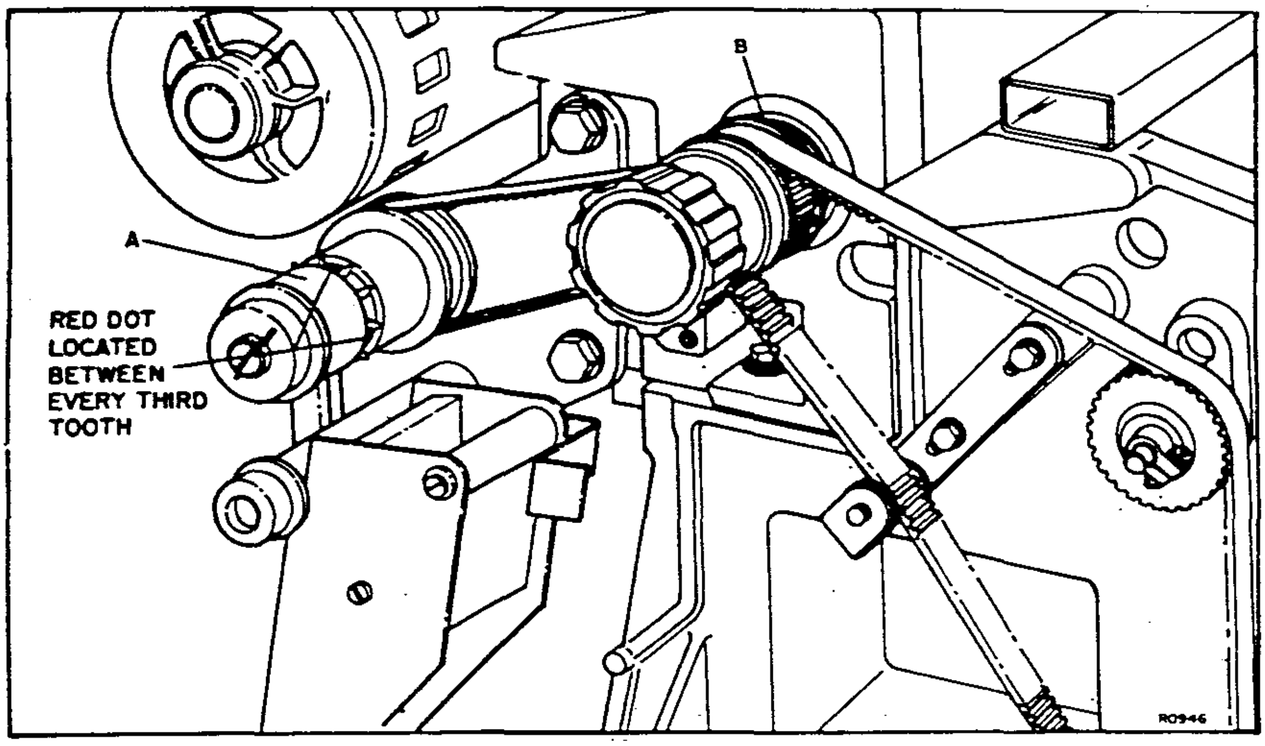


Figure 2.3 Format Reader Drive Sprocket Assembly

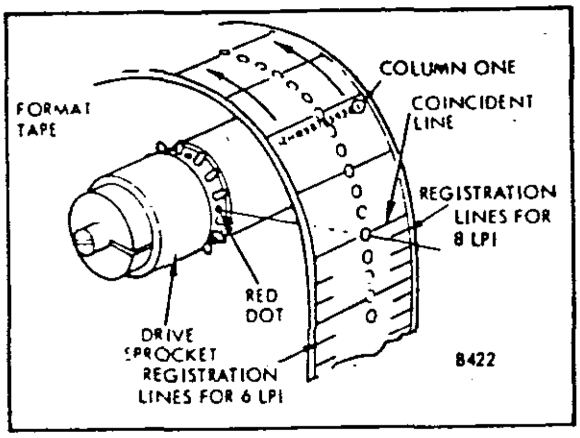


Figure 2.4 Format Tape Installation

2.6 Vertical Format Unit

The Vertical Format Unit consists of a cage which holds a brush block.(Figure 2.5 and Figure 2.6). The brush block holds the brushes in position. When the cage is closed and locked the brushes make contact with a roller held at a potential of 5 volts. One of the brushes is connected to the 5 volt power supply and is never interrupted by the paper tape. The paper tape is drawn through the vertical format unit by the sprocket roller. Control of vertical line spacing is accomplished by holes punched in a format tape.(Refer to figure 2.7). The format tape reader drive sprocket is driven by the forms drive (clutch/brake) shaft so that each single line movement of the shaft and tractors corresponds to a single frame movement of the format tape. Where there is a hole punched in the paper tape the brush touches the 5 volt roller and pulls it to a logical high. This causes a pulse to be generated. The 5 volt roller is electrically isolated from the chassis by special bearings.

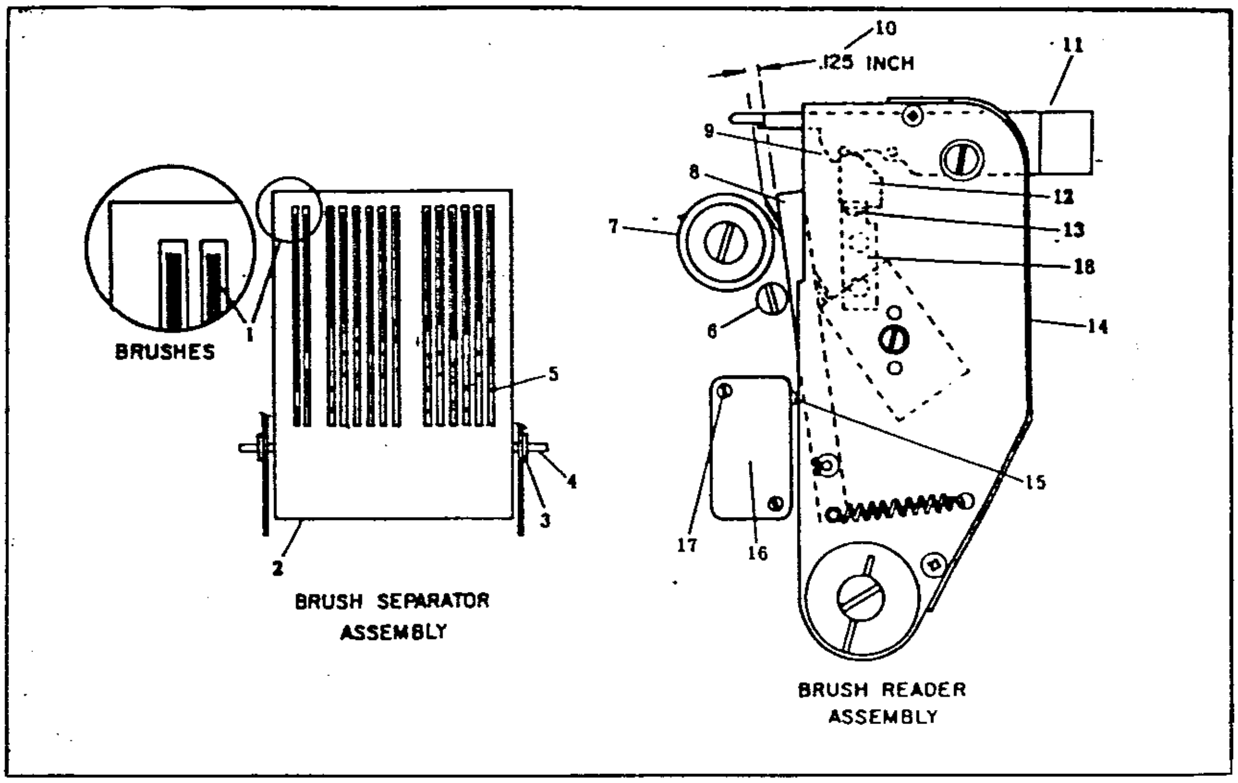


Figure 2.5 Brush Reader

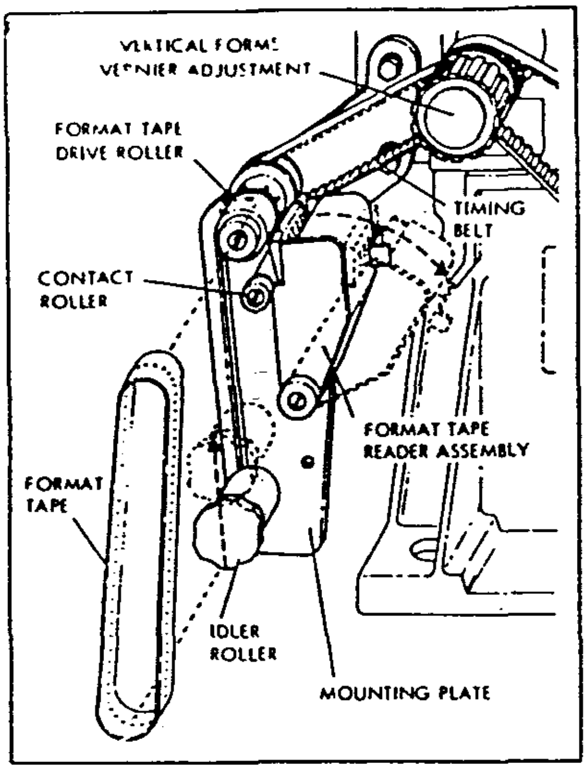


Figure 2.6 Format Tape Reader Assembly

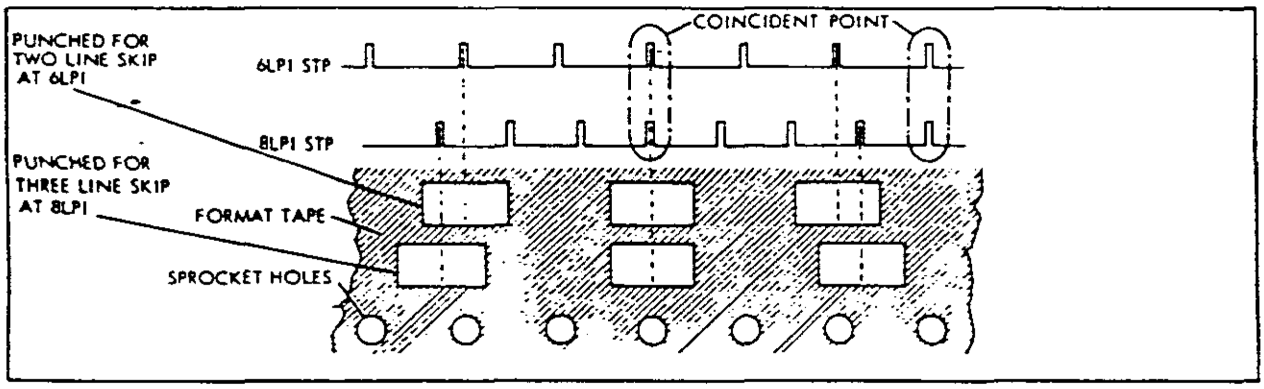

PUNCHED FOR TWO LINE SKIP AT 6LPI
COINCIDENT POINT
6LPI STP
8LPI STP
PUNCHED FOR THREE LINE SKIP AT 8LPI
FORMAT TAPE
SPROCKET HOLES

Figure 2.7 Stop Pulse and Format Reader Alignment

2.7 Timing gears

The function of the timing gears is to generate a pulse at intervals of one sixth or one eighth of an inch of paper throw. The timing gears consist of small teeth, set at one sixth or one eighth of an inch. These gears are magnetized by a small solid magnet fixed closely to the shaft.(See figure 2.8) Above the teeth is a small coil of wire on a metal former When a tooth passes under the coil of wire a current is induced in it by the magnetic fluxlines cutting it. This causes a pulse of approximately 2.7 micro seconds. Both the six and eight lines per inch pulses are generated, but only one is allowed to the printer electronics. This is selected by the operator by pressing a switch. The timing gears are set so that every half inch there is a coincident pulse, ie both the six and eight lines pulse fall at the same time.

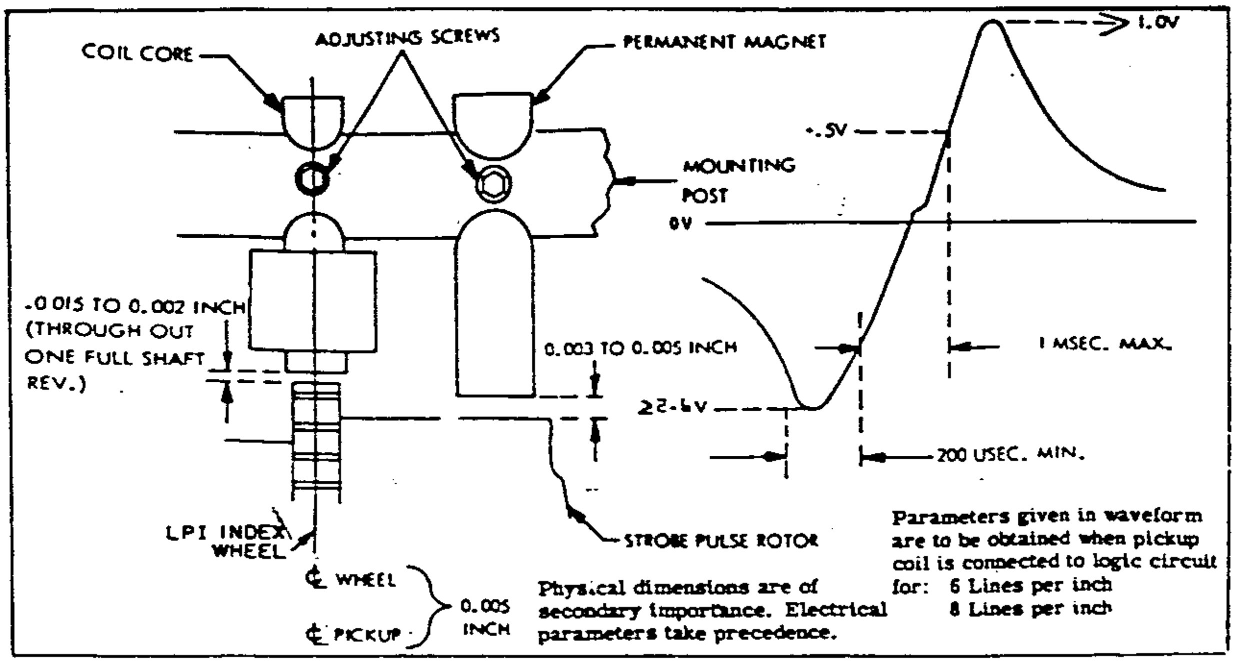

COIL CORE
ADJUSTING SCREWS
PERMANENT MAGNET
MOUNTING POST
-0.015 TO 0.002 INCH (THROUGH OUT ONE FULL SHAFT REV.)
0.003 TO 0.005 INCH
LPI INDEX WHEEL
STROBE PULSE ROTOR
℄ WHEEL
℄ PICKUP
0.005 INCH
1.0V
+.5V
0V
≥2-6V
1 MSEC. MAX.
200 USEC. MIN.
Physical dimensions are of secondary importance. Electrical parameters take precedence.
Parameters given in waveform are to be obtained when pickup coil is connected to logic circuit for: 6 Lines per inch 8 Lines per inch

Figure 2.8 Stop Pulse Generator and Waveform

2.8 Operation

The printer electronics will stop advancing paper and print a line when the six or eight lines per inch pulse falls into a window pulse generated by the vertical format unit (Figure 2.9) This means that the sprocket roller must be set such that there will be a window pulse generated to allow a six/eight lines per inch pulse to fall into the window. The window pulse must at the same time be set such that the coincident pulses of the six/eight lines per inch signal fall on a half inch interval (Figure 2.10) This means that a paper tape top of form hole will be positioned on a red spot, (Figure 2.3 and 2.4) which indicates the coincident pulse. This allows an operator to switch between six or eight lines per inch without calling a technician to set the timing again. A paper tape punched in multiples of a half inch can be run at either six or eight lines per inch without changing the timing or the print position.

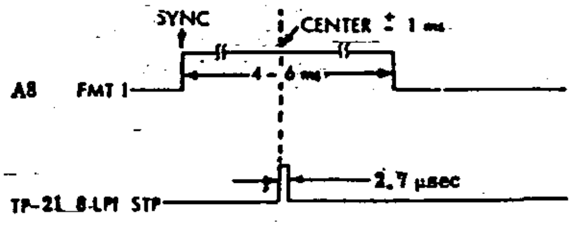


Figure 2.9 Format Reader Synchronization

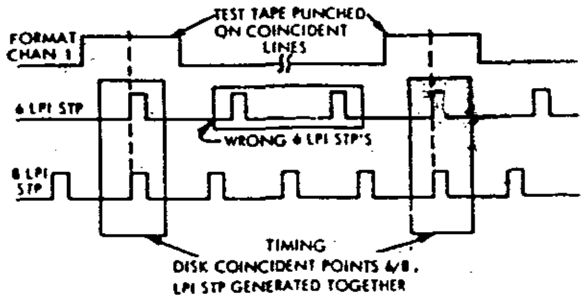

TEST TAPE PUNCHED ON COINCIDENT LINES
FORMAT CHAN 1
6 LPI STP
WRONG 6 LPI STP'S
8 LPI STP
TIMING
DISK COINCIDENT POINTS 6/8,
LPI STP GENERATED TOGETHER

Figure 2.10 Format Coincidence

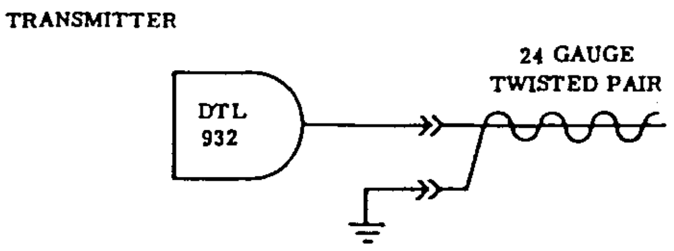
2.9 Data Transfer

The systems data transfer system was modified by the suppliers of the MICR printing system to allow faster data transfer rates. The circuit diagram of the modified interface is shown on page B2.

2.9.1 Transmitter/Receiver

The basic controller has a short line interface (up to 10 feet/3 metres), which is DTL (Diode Transistor Logic) and TTL (Transistor Transistor Logic) compatible. Figure 2.11

shows a typical transmitter and receiver configuration and the recommended cable characteristics.



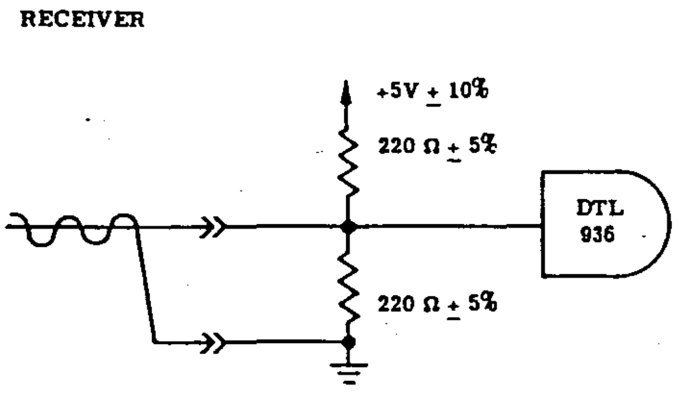


Figure 2.11 Typical Receiver and Transmitter configurations

2.9.2 Interface Voltage levals

When the recommended configuration shown in Figure 2.11 is used the line voltage levels will be as follows:

```
logic "1" 3.25 +- 0.5 volt   (True)
logic "0" 0.20 +- 0.2 volts  (False)
```

2.9.3 Interface Signals

See figure 2.12

2.9.3.1 Ready

The Ready line is an interface line from the printer to the controller which being False (0 Volt) indicates that the printer is fully operational. It means that no alarm conditions exist, and that the control panel start button has been depressed.

2.9.3.2 Line Ready

This is an interface line from the printer which, when False (0 Volt)indicates that the printer is ready to accept a line of data. This line will become False preceding the generation of the first Character Request and will remain False until the Control Bit signal is received. This line will remain True (5 Volt) during the print cycle and during a paper advance indicating that the printer controller cannot and will not accept data.

2.9.3.3 Character Request

An interface line from the printer which, when False indicates that the controller is ready to accept a data character from the data source. The data source must sense this line before a Data Strobe signal is generated. See Figure 2.13 for timing relationships. This line will go True after the data byte has been sampled, and the Data Strobe signal may be removed at this time.

2.9.3.4 Data Strobe

An interface line from the data source which, when False, indicates to the printer controller that the data byte has been stabilized, and the data lines may be sampled.

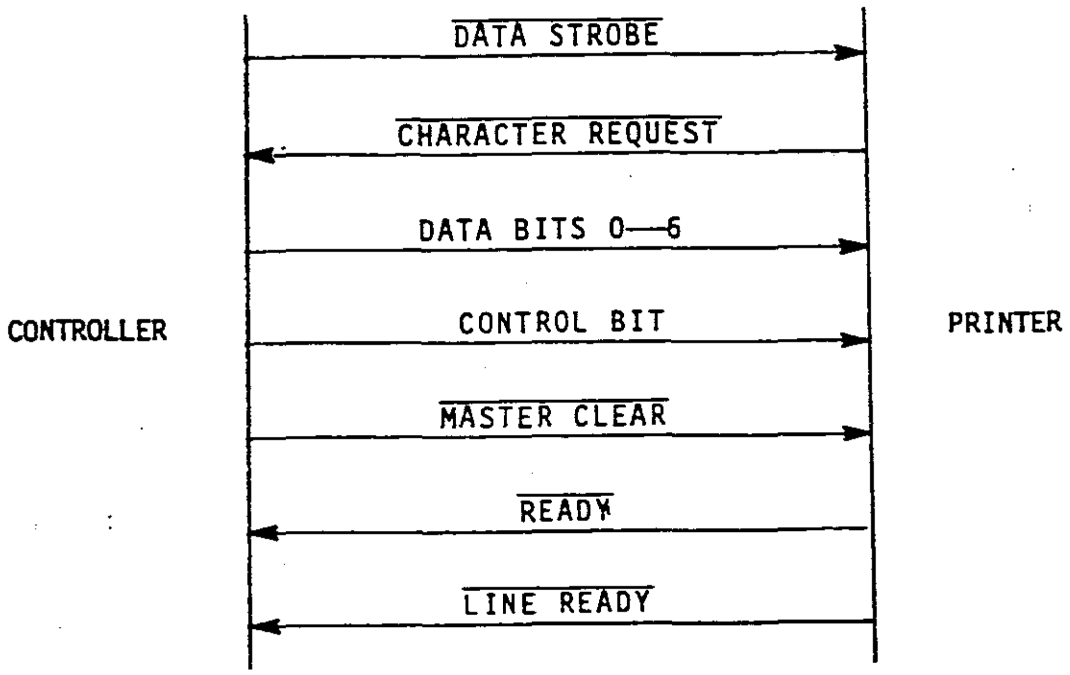

CONTROLLER
PRINTER
$\overline{\text{DATA STROBE}}$
$\overline{\text{CHARACTER REQUEST}}$
DATA BITS 0—6
CONTROL BIT
$\overline{\text{MASTER CLEAR}}$
$\overline{\text{READY}}$
$\overline{\text{LINE READY}}$

Figure 2.12 Control and Data signals between Printer and Mini computer

2.9.3.5 Data Bits

Six data lines carry the information (control codes and data characters) from the mini computer to the line printer controller. The data must be placed on the data lines 1.0 microseconds prior to the generation of the Data Strobe signal, and must not be removed until the Character Request signal goes true. See Figure 2.13

2.9.3.6 Control Bit

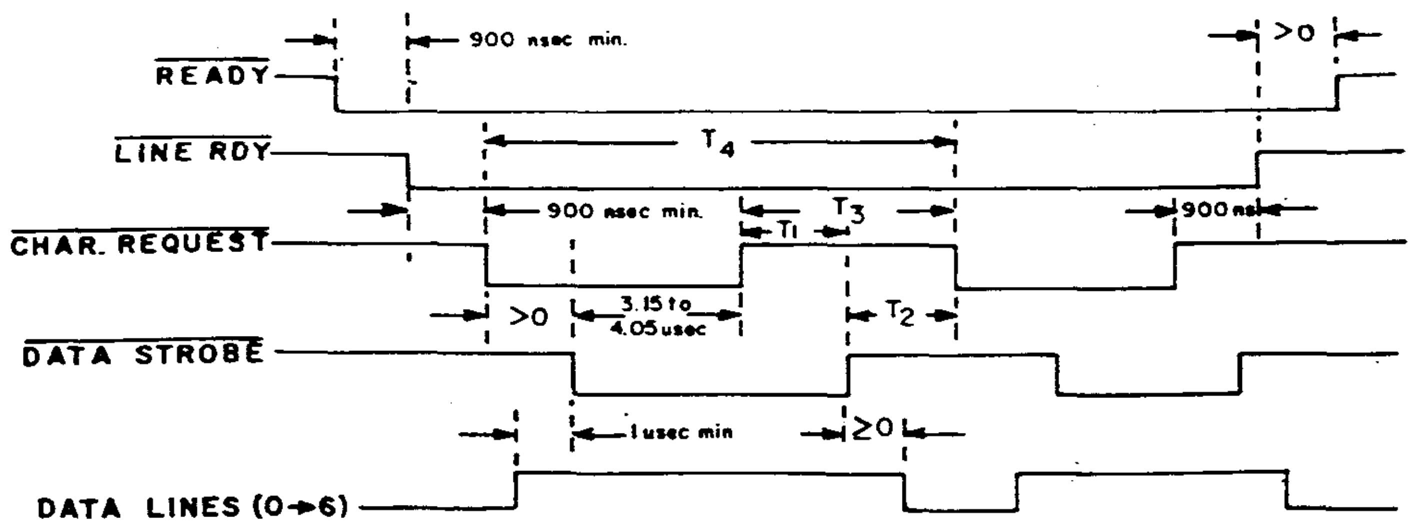
The control bit is an interface signal from the data source which, when true, indicates to the controller that the data input is being truncated, and that the printer may now enter its print cycle. The timing relationship of the Control Bit to other interface signals is the same as that of the six data bits. The code on the data lines when Control Bit is true is ignored. Control Bit must be true in order to truncate the data input.

2.9.3.7 Processor Master Clear

An interface signal from the data source which, when false, indicates to the printer controller that all circuits should be placed in their initial steady state condition. The signal does not, however, remove the printer from the Start mode (Ready line at a logical "0"). The signal must be a minimum of 5 microseconds in duration.

2.9.4 Data Interchange Technique

The basic controller operates on an interlocking handshaking technique. The interlocking feature simply means that there are no stringent timing requirements on the data exchange signals. Transitions from one voltage level to the other rather than pulses are used to exchange data. The printer controller asks for a character by placing the Character Request signal at a logical "0". The only timing requirement is that the data be stable on the data lines for 1.0 microseconds, and that the data source senses Character Request before the Data Strobe signal is generated. The printer will drop the Character Request signal when the data byte has been sampled. When the printer senses the removal of Data Strobe it will generate another Character Request when it is ready to accept another character. (See Figure 2.13) for complete interface timing).

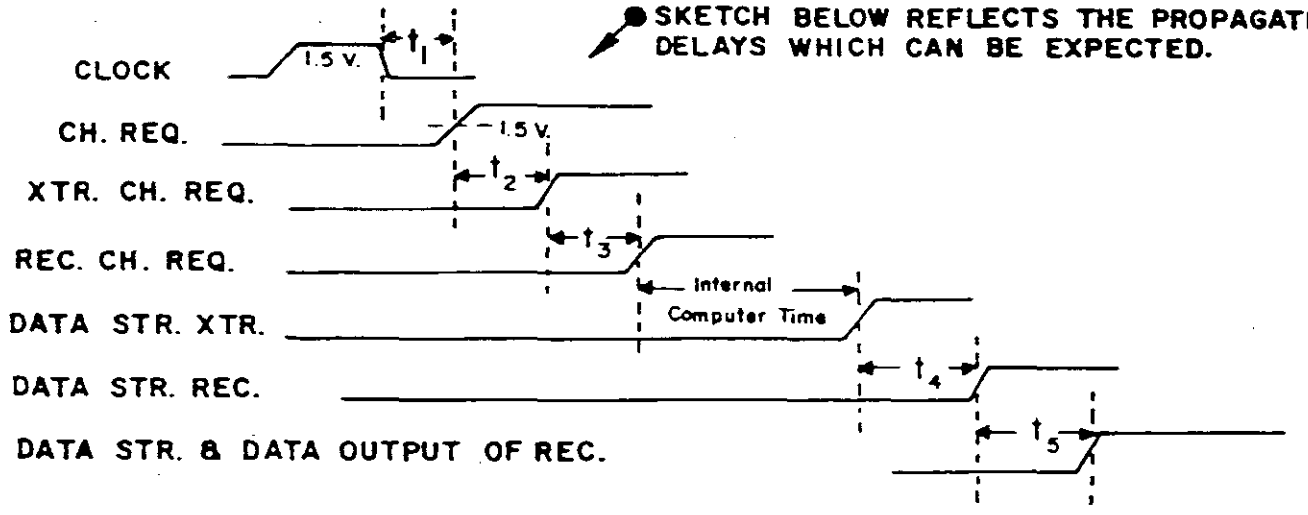


ABOVE INTERFACE LINES ARE SHOWN AT THE INPUT OF THE CDC'S PRINTER RECEIVERS, OR THE OUTPUT OF ITS TRANSMITTERS.

NOTE 1: IF T_1 IS LESS THAN 900 nsec., T_3 ($T_1 + T_2 = T_3$) WILL BE 1.8 usec.. AS T_1 BECOMES LARGER THAN 900 nsec., T_3 WILL INCREASE BY MULTIPLES OF 900 nsec. ($T_3 = 1.8 + n \times 900$ nsec. WHERE $n = 1,2,3\ldots$) T_2 IS ALWAYS ≥ 0.

NOTE 2: IF WE ASSUME DATA IS SET UP DURING T_2, THE OVERALL TRANSFER RATE T_4 WILL BE 4.95 usec. TO 5.85 usec. (1.8 + 3.15 OR 4.05)

● SKETCH BELOW REFLECTS THE PROPAGATION DELAYS WHICH CAN BE EXPECTED.



$t_1 \cong$	65 n sec.	$t_4 \cong$	10 n sec.
$t_2 \cong$	35 n sec.	$t_5 \cong$	50 n sec.
$t_3 \cong$	10 n sec.	TOTAL	170 n sec.

NOTE 3: T_3 AND T_4 ARE FOR APPROXIMATELY 10 FT. OF CABLE. TYPICAL CABLE DELAY IS APPROXIMATELY 1 n sec. PER FT.

Figure 2.13 Interface Timing Relationships

2.9.5 Data Sequencing

The first character transferred is recognized as the format character (Control Bit must be logical "0" at this time). The following characters are recognized as data characters and are printed with the first character in the left most column position. The input may be truncated at any time by the sending of a Control Bit. The code on the lines at this time is neither stored nor printed. A Control Bit must be sent in order to end the line. Characters in excess of the column capacity for that particular printer will be responded to, but will not be stored or printed. If a short line is sent (less than the column capacity for that printer), the remaining positions will be filled with blanks.

2.9.6 Programming

A sample source code of the program used to control the printer is shown in Appendix A

2.9.7 Interface Circuits

The interface circuits used by the Mini Computer are shown in Appendix B. The circuits used by the printer can be seen in Control Data Corporation's Technical Reference Manual number 9352. The relevent pages are Sheet 4A03, 4B01, 4B02, 4B08 and 4B10. These sheets were not included due to the extremely bad quality of the photocopies.

3.1 Requirements and Constraints of the new system

The modification to the printer should be done in such a way that the printer could still use the old vertical format unit tapes. This consideration is due to the fact that every program that uses the printer has to be modified, a task that would take a considerable time. The modification involves inserting a section of code into the programs (See Appendix Sheet A6). This is done manually, in machine language, by setting switches.

This was done in such a way that all the vertical format unit parts were kept on the printer. The signals from the six/eight lines per inch paper throw sensor were interrupted. They were fed into the new interface, from now on referred to as the vertical format unit computer (VFU CPU), and processed. The processed signal was then sent to the printer to the same place where it would have normally gone. The system was designed such that the default setting on switch-on would be to use a vertical format tape. This means that every pulse received from the paper throw sensor was simply transferred to the printer. Should the program being run have been modified, it will send a string of set-up commands to the VFU CPU. Another consideration is that the existing interface could still be used should the microprocessor interface (VFU CPU) fail. This was done by modifying the plugs such that should the interface be removed, the system would need no modification at all. All that would be required would be to unplug the new interface and reconnect the old cables.

4.1 Description of the microprocessor board (VFU CPU)

The microprocessor board is based on the Z80 microprocessor manufactured by Zilog. The system comprises of the microprocessor, a buffer circuit, a clock generator, an interrupt circuit, a reset circuit, memory, input/output circuits and control circuits. The block diagram of the VFU CPU is shown in Figure 4.1

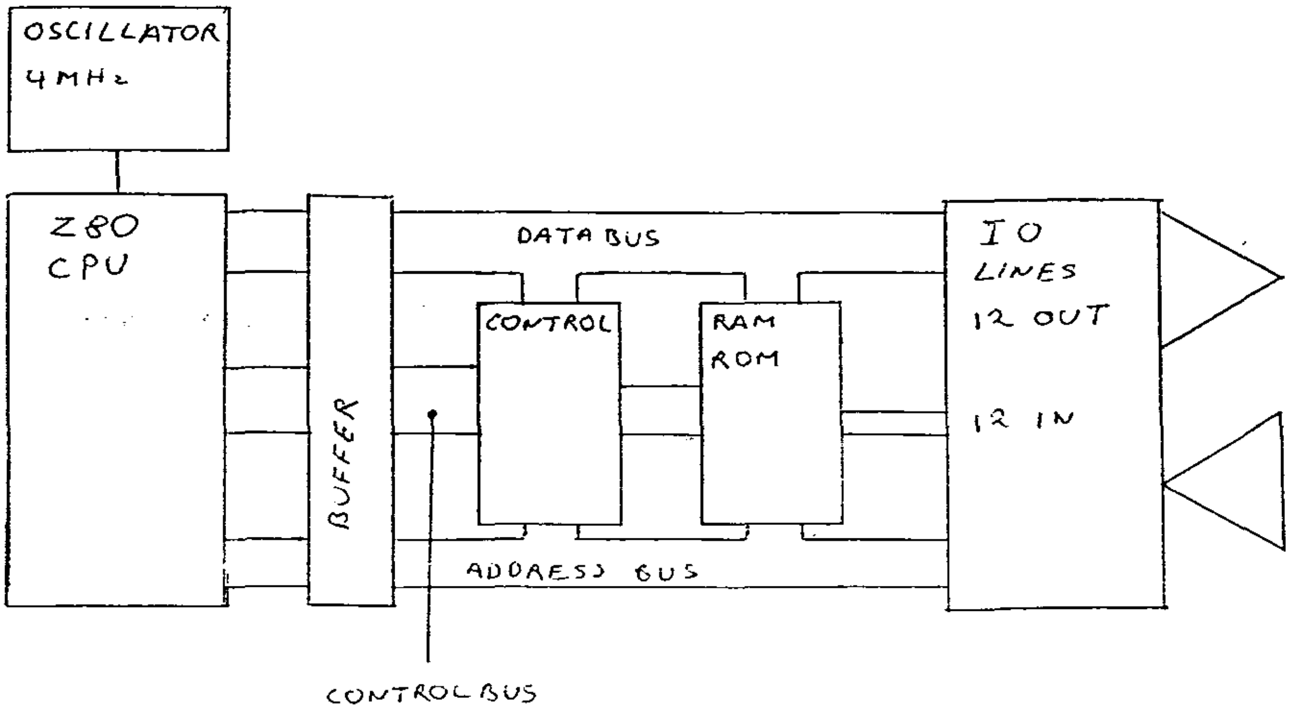


Figure 4.1 Block Diagram of the VFU CPU

4.1.1 Microprocessor

The microprocessor is the well known Z80A microprocessor chip. The functional block diagram and commands are shown in appendix E. Refer to Appendix C Sheet 4 of the circuit diagram. Two of the control signals are not used. These are the NOT MASKEBLE INTERRUPT and BUSREQ lines. However circuitry to allow future use of the bus request signal was included

4.1.2 Buffer Circuit

To allow the microprocessor to run at a speed of 4 mega hertz, buffer and line drivers had to be used. Refer to Appendix C Sheet 4 of the circuit diagram. The data bus is buffered with a Octal Bus Tranceiver (Type 74LS245 chip 5) The address bus is buffered with two Octal Buffers/Line Drivers/Line Receivers (Type 74LS244 chips 2 and 3) The control lines of the microprocessor are buffered with another Octal Buffers/Line Drivers/Line Receivers chip (Type 74LS244 chip 4). The direction of data flow on the data bus is controlled by the RD, IORQ and M1 signals. When the microprocessor is in a read cycle, either from memory or external I/O, the direction of data flow is toward the microprocessor. At all other times the data flow is set away from the microprocessor.

4.1.3 Clock Generator Circuit

See Appendix C Sheet 1. The clock generator is based on a TTL inverter oscillator controlled by a 4 mega hertz crystal. The output of the clock generator is buffered by the recommended circuitry shown in the Mostek manual. (See list of references). This circuitry has an active pull up to allow fast rise times of the clock. This allows maximum frequency operation.

4.1.4 Reset Circuit

See Appendix C Sheet 2. The Reset Circuit consists of a manual and a power on reset circuit. The Z80 CPU has the characteristic that if the Reset input goes low during T3 of an M1 cycle that the MREQ signal will go to an intermediate state for one T-state approximately 10 T-states later. If there are dynamic memories in the system this action could cause an aborted or short access of the dynamic RAM which could cause destruction of data within the RAM. If RAM contents must be preserved, then the falling edge of the RESET input must be synchronized by the falling edge of T1. The circuitry shown in sheet 2 does this synchronization as well as providing a one-shot to limit the duration of the CPU RESET pulse. This circuit was used to allow the future use of dynamic RAM even though none was used in this application.

4.1.5 Interrupt Circuit

See Appendix C Sheet 3. The Interrupt Circuit uses the six/eight lines per inch pulse from the printer circuitry to interrupt the VFU CPU. The Z80 microprocessor acknowledges a interrupt at the end of the current instruction cycle by setting IORQ and M1 lines to a logical "0". These lines are used to reset the interrupt flip-flop to allow the next pulse from the six/eight lines per inch generator to interrupt .

4.1.6 Wait Circuit

See Appendix C Sheet 5. The Wait circuit was included to allow the use of slower memory such as the 2716 ROM. The original VFU CPU had some problems which were thought to have been due to too fast a clock speed. This was later found to be not the case. The Wait circuitry could be easily removed by removing chip 23(7474)and jumping pin 9 of the removed chip to pin 7(0 volt).

4.1.7 Control and Address Decoder Circuit

See Appendix C Sheet 6. The whole VFU CPU is memory mapped. Refer to Figure 4.2. The circuit controls the memory access with CSROM for the 2 kilo bytes of ROM and CSRAM controls the 2 kilobytes of RAM. The RAM is further decoded for a READ or a WRITE situation by RAMWR.
All Input/Output , except for two exceptions, is done through the Intell 8255 Programmable Peripheral interface and is selected by CS8255. The exceptions are CSL and CSTOF which are decoded seperately. These signals are used to output the processed six/eight lines per inch pulse to the printer.

Memory locations	Function
0000 - 07FF	Read Only Memory
0800 - 0FFF	Random Access Memory
8000 - 8FFF	Vertical Format Unit Signals
9000 - 9FFF	Input / Output Ports(8255)

The undefined memory is free

Table 4.1 Memory Map of the VFU CPU

4.1.8 Memory Circuit

See Appendix C Sheet 7. The memory consists of one 2 kilo byte ROM (Type 2716) and one 2 kilo byte RAM (Type 6116). The ROM is decoded as the first two kilo byte, the RAM as the next two kilo byte.

4.1.9 Interface Circuit

The main interface of the VFU CPU is the 8255 programmable peripheral interface.The 8255 is designed to operate in mode 0, using port A and the lower half of port C as output, and port B and the upper half of port C as input. This gives twelve input and twelve output lines. The input lines are impedence matched with 220 ohm resistors to +5 volt and 330 ohm resistors to 0 volt. The outputs are all buffered by open collector buffers (Type 7417). The output of each open collector is pulled to a logical "1" by a 240 ohm resistor.

4.1 10 Output and Display Circuit

See Appendix C Sheet 9. The output signal CSL (Appendix C Sheet 6) is used to trigger a monostable. The output of the monostable is buffered by a open collector buffer and is then connected to the six/eight lines per inch line where it was interrupted.
The CSTOF line (Appendix C Sheet 6) is similarly buffered. Its use is however different. The output can be used to trigger a guillotine to cut one sheet. Extra circuitry would be needed to electrically isolate the guillotine from the printer, but this could be easily done using a relay. A further feature was the inclusion of a circuit to indicate to the operator when the paper was in the Top Of Form position using a light emitting diode.

4.1.11 Connector Cables

See Appendix C Sheet 10. The VFU CPU is connected to the printer and computer via an edge connector. This edge connector is itself connected to the existing male and female plugs. The plugs were modified to include the six/eight lines per inch lines as well as allowing the system to be reconfigured to its original by just removing the VFU CPU.

4.1.12 Construction Details

The PCB layout wad designed at the Cape Technikon using their SMARTWORK software package.The PCB was etched and drilled there as well.

4.2 Description of modifications done to the printer

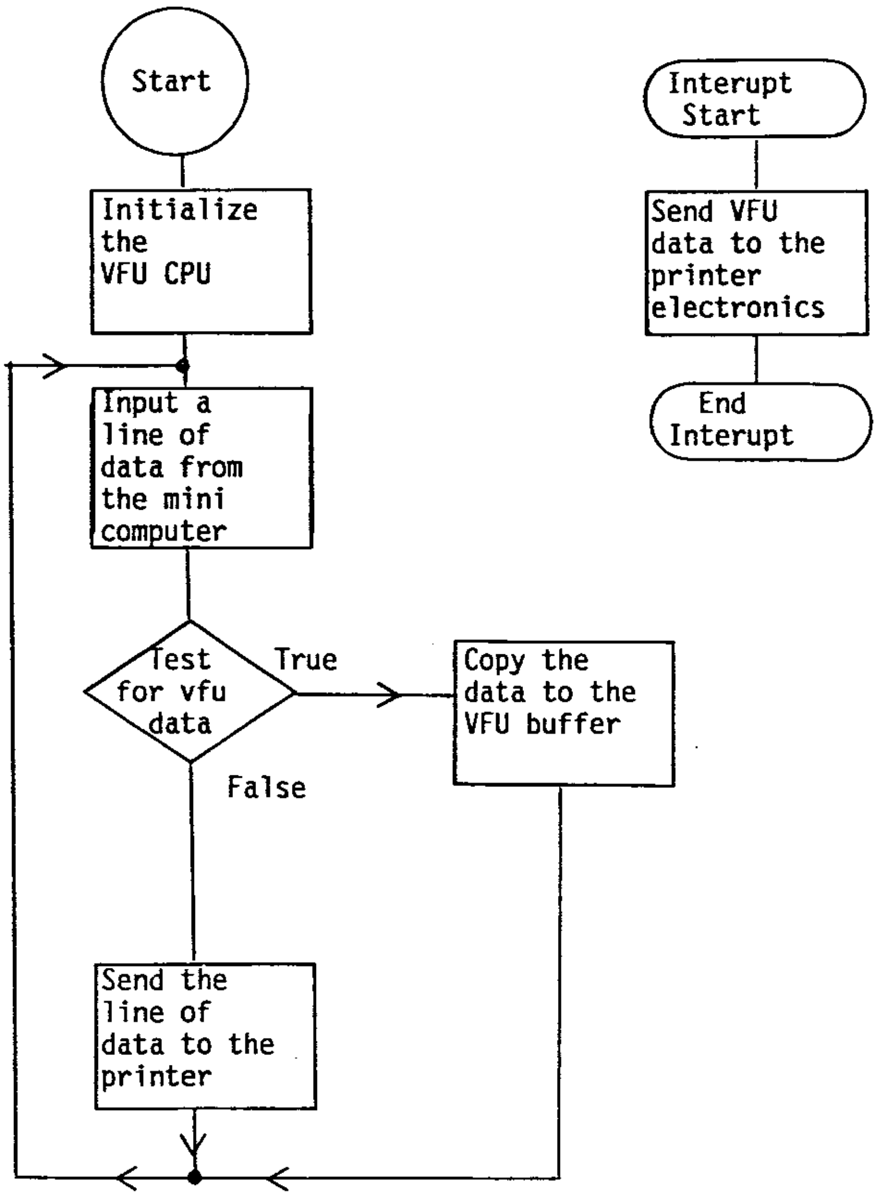
See Appendix C Sheet 11. This sheet describes the modifications done to the printer interface plugs to allow the system to work. These modifications include the replacing of the +5volt line (Appendix C Sheet 10) from the printer back plane to the interface plug with a thicker wire as the exsisting wire could not handle the current drawn by the VFU CPU (Approximately 1 Ampere). Another modification was the interrupting of the existing 6/8 lines per inch signal between boards 4A03 and 4B08. See Appendix B Sheet 5 and 6 . The output signal from the 4A03 board at pin A16 was cut and connected to the male interface plug to pin BB (Appendix C Sheet 10). The other side of the cut line now comes from pin DD (Appendix C Sheet 10) and connect to pin A20 of board 4B08 (Appendix B Sheet 6). Pins CC and EE were used to connect the twisted pair return wires for the 6/8 lines per inch signal. The interface plug from the mini computer was modified to cater for the new route of the 6/8 lines per inch signal by jumping pin DD to pin BB and pin CC to pin EE. A special piece of test gear used to exercize the printer by simulating. A mini computer had to be modified as well by connecting the pins DD to BB and CC to EE to its connector plug. The Printer Reset signal was also tapped at pin B10 on board 4B10 (Control Data Corporation Technical Reference Manual 9352) and connected to pin Z of the printer interface plug to force a reset of the VFU CPU whenever the operator resets the printer.

5.1 Simple Outline of the Program

A flowchart of the program is shown in Figure 5.1. A Key to the symbols used is shown in Appendix D Page 7. The program, obviously being dependant on the hardware, is written such that it continually scans the communications ports, ie the output signals of the host computer and the printer. Any signal received is transferred to the equivalent output provided the signal is ready to be accepted. If the host computer is ready to send valid data, but the printer is still busy, the program will store the data in a first in first out (FIFO) buffer. When the printer is ready to accept data it will be transferred from the first in first out buffer, provided it has data. Should this be the case the program will simply wait for the host computer. The data being received from the host computer is also checked for control codes specific to the VFU CPU. These codes are simply to inform the VFU CPU that the next section of data is for use of the VFU CPU. This data is the look up table for the paper throw.

The above program is of secondry importance to the VFU CPU. The main function of the system is to control the paper throw. This is done by interrupting the above program section at any time with a six or eight lines per inch pulse. When the system is interrupted the program saves the return address and outputs the data found at the VFU buffer pointer to the control port. The program then increments the pointer and checks weather the next data byte is a top of form. Should this be the case, the pointer gets reset to the first location of the buffer and the interrupted program is resumed. Should the next byte be just a normal linefeed or print byte nothing else exiting happens. The program is resumed where it was interrupted.

On initialization the program sets up the VFU buffer to print a line every time the program gets interrupted. This is done so that if the host computers program was not modified to use the VFU CPU the printer could still be used by using a punched format tape.



Start
Initialize the VFU CPU
Input a line of data from the mini computer
Test for vfu data
True
False
Copy the data to the VFU buffer
Send the line of data to the printer
Interupt Start
Send VFU data to the printer electronics
End Interupt

Figure 5.1 The System Flowchart

5.2 Initialization Routine

Refer to Figure 5.2 and Appendix D page 1. On power on the Z80CPU gets reset by the reset circuit. (Appendix C Sheet 2). This forces the program counter to zero and initializes the CPU. The CPU initialization includes:

1) Disable the interrupt enable flip-flop
2) Set Register I = 00
3) Set Register R = 00
4) Set Interupt Mode 0

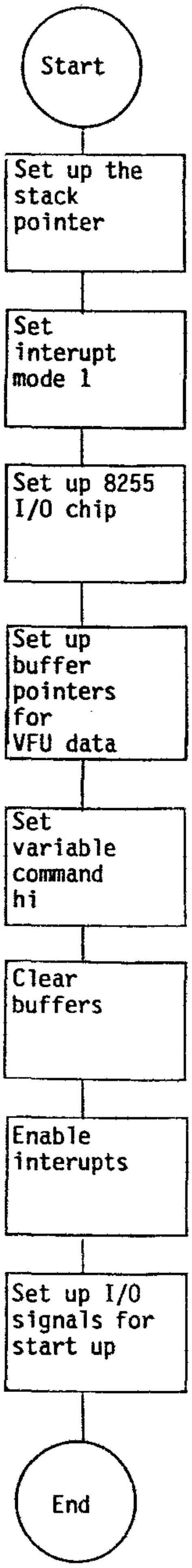


Figure 5.2 The Initialization Flowchart

The CPU now starts at memory location 00. The program now sets the Stack Pointer to 0FFF Hex, sets interupt Mode 1 and programs the programmable I/O chip, the Intel 8255 (Appendix C Sheet 8) to set port A and bits 0 to 3 of port C to outputs and port B and bits 4 to 7 of port C to inputs. The next sequence is to set up the data buffer pointers and the VFU table pointers. This is done by using the registers of the Z80 CPU. The Z80 CPU has the feature where there are two full sets of registers, however only one set is available at a time. See figure 5.3.The command to swop the registers is 'EXX'. The program uses the normal set of registers for normal data and the alternate set for VFU data.

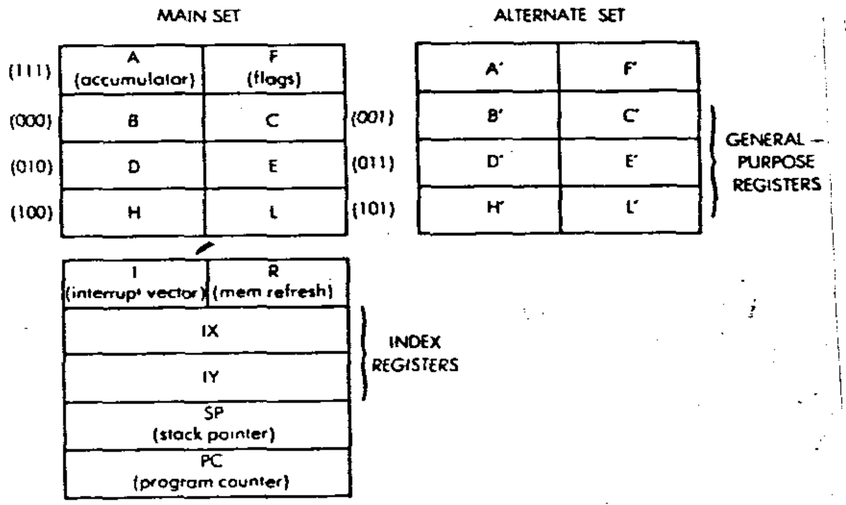


Figure 5.3 The Z80 Registers

The next section clears the Memory Buffer with 'FF Hex' and the VFU Buffer with '00'. After this the interrupt is enabled and the ports set to the initial state. This is done by setting bits 0 to 3 of port C to a logic '1'.
The last statement in the initialization phase is to start the main loop of the program by jumping to VSTART.

5.3 Interupt Routine

Refer to Figure 5.4 and Appendix D page 2. When Interrupt Mode 1 has been selected by the program, the CPU will respond to an interrupt by executing a restart to location 0038 Hex.

The first instruction in the interrupt program is to disable any futher interrupts. The AF register is saved by pushing it onto the stack. The alternate register set is called for and the byte found at the location pointed atby the DE Register is loaded into the A Register and immediately outputted to the 6/8 Lines Per Inch circuit (Appendix C Sheet 9). Now that the urgency has been taken care of the program checks if the byte sent out was a Top Of Form byte. If it was a Top Of Form byte the DE Register gets reset to the beginning of the VFU Table. If it was just a normal byte, ie a line throw or a print byte, the DE Register gets incremented. Whatever sort of byte it was the next section restores the normal Register Set and resets the AF register. Following this interrupts are enabled again and the program returns to its place where it was interrupted.

5.4 Clear Routine

Refer to Figure 5.5 and Appendix D page 2. This subroutine is used by the initialization routine to clear the Memory and VFU Buffers. The first section is to clear the Memory Buffer to FF Hex. Initially the registers DE, AF and BC are saved on to the stack. Register A is loaded with FF Hex. This register is then loaded into the memory location pointed to by the DE Register. The DE Register is then incremented and compared to the BC Register in the subroutine Compare. The BC Register was initially loaded in the initialization phase with the Memory Buffer Stop location. This routine loops until the Memory Buffer is cleared. Once this happens the program sets up the BC and DE Registers as start and stop addresses for the VFU Buffer. The A Register is then set to 00 and the VFU Buffer gets cleared in a similar way to the Memory Buffer. The routine resets the BC, AF and DE Registers once all this has happened.

5.5 Compare Routine

Refer to Figure 5.6 and Appendix D page 3. The Compare routine was written as there is no instruction in the Z80 instruction set to compare two 16 Bit registers without corrupting one of them. This routine starts by loading the D section of the DE Register into the A Register. This is then compared nondestructively with the B Register of the BC Register. If there is a difference ie the zero flag is not set, the routine immediately aborts and returns to the calling program. Should the zero flag be set the program checks the E and C Registers in a simmilar way to the D and B Registers. The 'CP' command compares the A Register against any other eight bit register and sets the zero flag if the comparison is true. This routine destroys the contents of the A Register. This is not a great loss as the A Register is used in this program as the work-horse and is constantly changing.

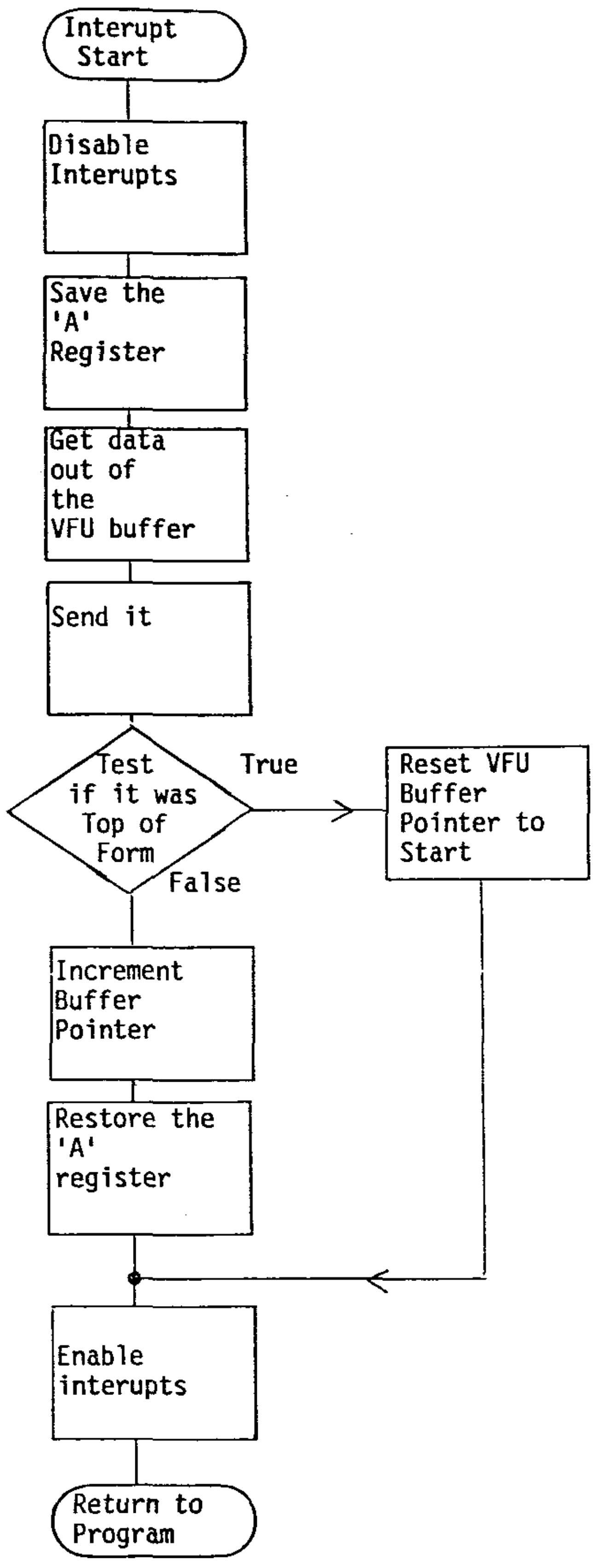


Figure 5.4 The Interrupt Flowchart

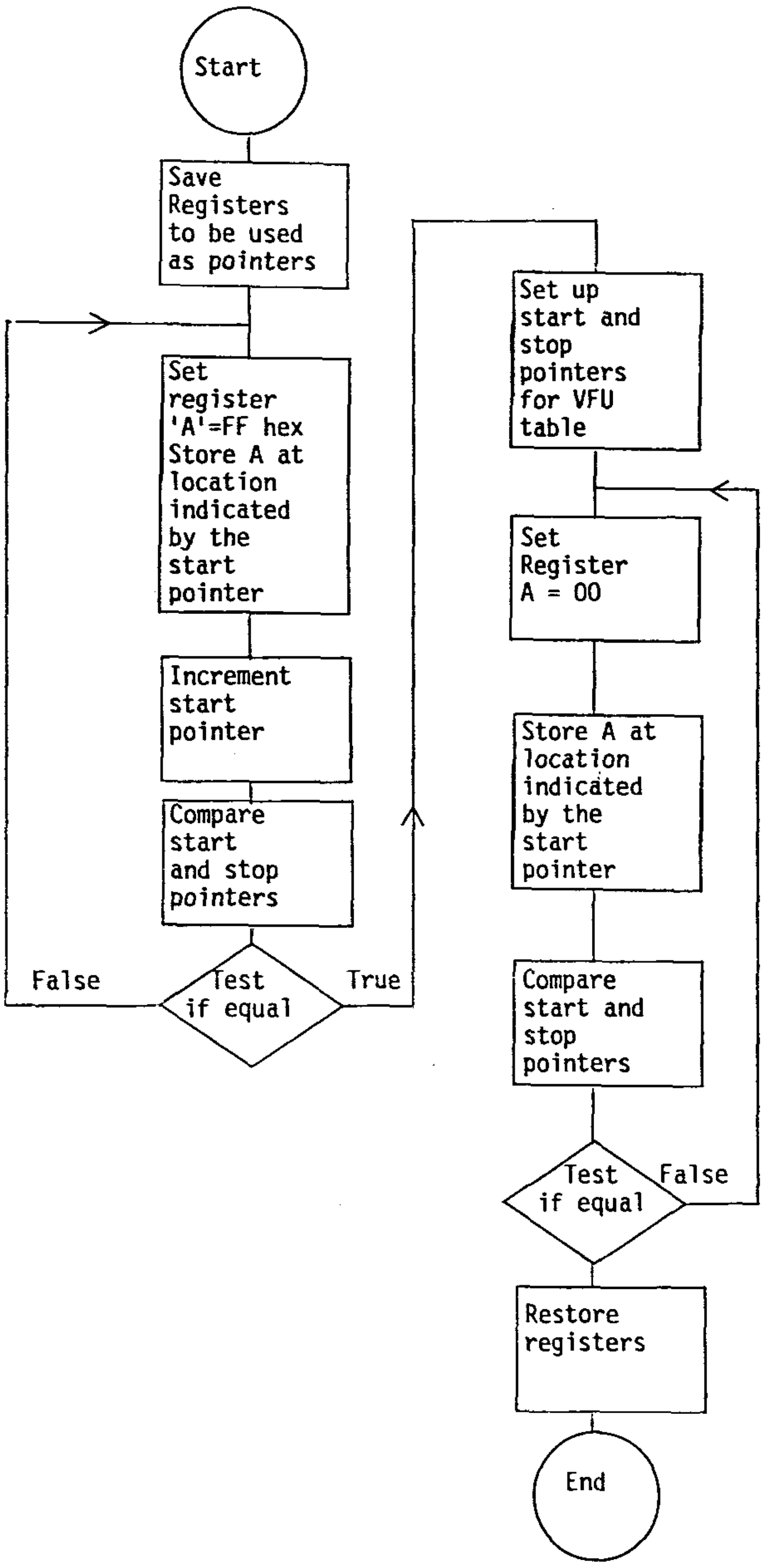

Start
Save Registers to be used as pointers
Set register 'A'=FF hex Store A at location indicated by the start pointer
Increment start pointer
Compare start and stop pointers
Test if equal
False
True
Set up start and stop pointers for VFU table
Set Register A = 00
Store A at location indicated by the start pointer
Compare start and stop pointers
Test if equal
False
Restore registers
End

Figure 5.5 The Clear Subroutine Flowchart

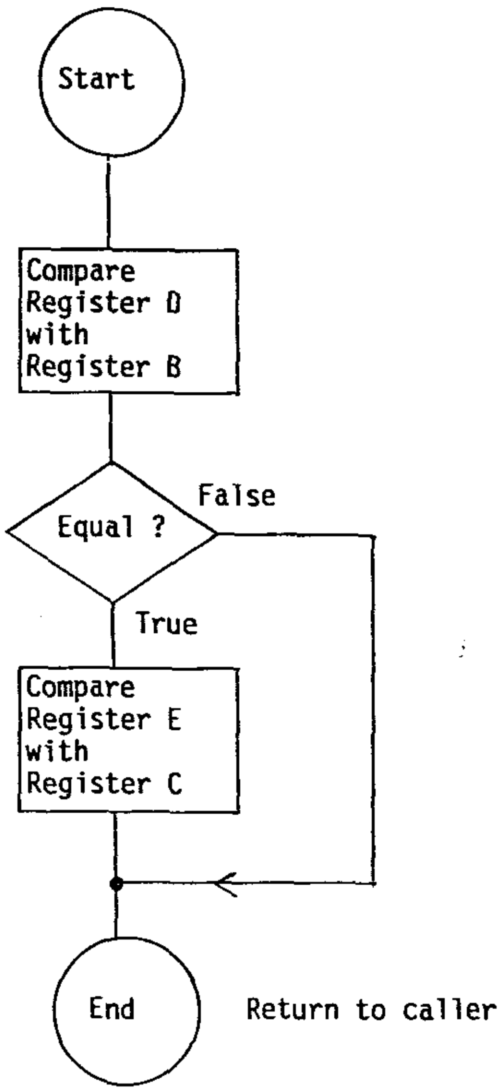

Start
Compare Register D with Register B
Equal ?
False
True
Compare Register E with Register C
End
Return to caller

Figure 5.6 The Compare Subroutine Flowchart

5.6 Main Program

Refer to Figure 5.7A, Figure 5.7B and Appendix D pages 3 to 5. The rest of the program deals with the control of the system ie the nitty gritty of the control signals between the Mini Computer, the CDC Printer and the VFU CPU.

The first thing the main program does is to set up the pointers for the control port and the memory buffer. After that it resets the READY and LINE READY signals. This is the start of the routine INPUTX. The program now enters into a loop which fetches characters from the mini computer. This is done by setting the CHARACTER REQUEST signal first to a logical high immediately followed by a logical low. The data lines are checked if the byte on the lines is a control byte ie bit 6 high. If it is a control byte the program branches to the EOLINE routine. The program has to check the status of the data lines first due to a peculiarity of the mini computer interface and program. If the last byte was sent the mini computers program sets the control bit high with the DOAP command and does not wait for an acknowledgement. When another character request comes along the program has already processed another line and is waiting to send the first byte of data, which is a command to tell the printer which VFU brush to select.

The program now does the normal handshake to get data ie it tests for the strobe signal and waits until it goes low. When it does it saves the data on the data lines into the memory buffer at the location indicated by the buffer pointer, register DE. The buffer pointer is incremented and the program loops back to the start of the INPUTX routine.
The EOLINE routine resets the DE Register to the beginning of the memory buffer and checks if the first byte is a 01 HEX. This 01 HEX is used as an indication byte to inform the program that the following data is for use of the VFUCPU only. If it is the program branches to a routine SETVFU. The above routine is used only once, on initialization

The program now waits for the printer to request data by setting CHARACTER REQUEST low. When it does the program jumps to the routine OUTPUT1. This routine transmits a line of data to the printer using the handshake procedure ie set valid data onto the data lines, then set the strobe low and wait for the printer to respond by setting character request high. When this happens the strobe is set high again and the program checks if the last byte sent is a control byte. If it is then the routine jumps to LOOP, else prepares to send the next byte.

The procedure LOOP is really the real work loop. It first tests the jumps to a procedure SETRO which sets the signals READY, LINE READY and CHARACTER REQUEST going to the mini computer high and then starts again at TESTRI, just below LOOP. If all is ok the program sets the READY line going to the mini computer low and tests LINE READY. Should this signal from the printer be high the program branches to SETLRO, which sets the signals LINE READY and CHARACTER REQUEST going to the mini computer high and then starts again at TESTRI. If all is ok the program now sets LINE READY going to the mini computer low and tests CHARACTER REQUEST from the printer. If the signal is high the program branches to SETCRO which sets the signal CHARACTER REQUEST going to the mini computer high. If all is ok the program sets the line CHARACTER REQUEST signal high at the stars of procedure INPUT. The next command is to set CHARACTER RTEQUEST low and then to test for a control byte. This procedure is almost identical to the procedure INPUTX at the begining of the main program loop. The two routines were not made into a subroutine as the routine INPUTX is used only once and a subroutine call would slow the program down a bit.

When the above routine receives a control byte it branches to OUTPUT4. This routine checks for trailing spaces and moves the control byte accordingly. This was done to speed up the transmission of a line to the printer. The mini computers program does not truncate a line, but space fills it to make a line of 112 bytes long. The printer however can accept lines less than 112 bytes long and space fills the rest automatically.

The program now checks if the first byte is a 01 HEX. If it is it branches to set up the VFU table, routine SETVFU. If the data is normal text data the program continues to output the data, already described above, ie routine OUTPUT1.
The routine SETVFU merely copies the contents of the MEMORY BUFFER to the VFU BUFFER and then jumps to the main routine LOOP.

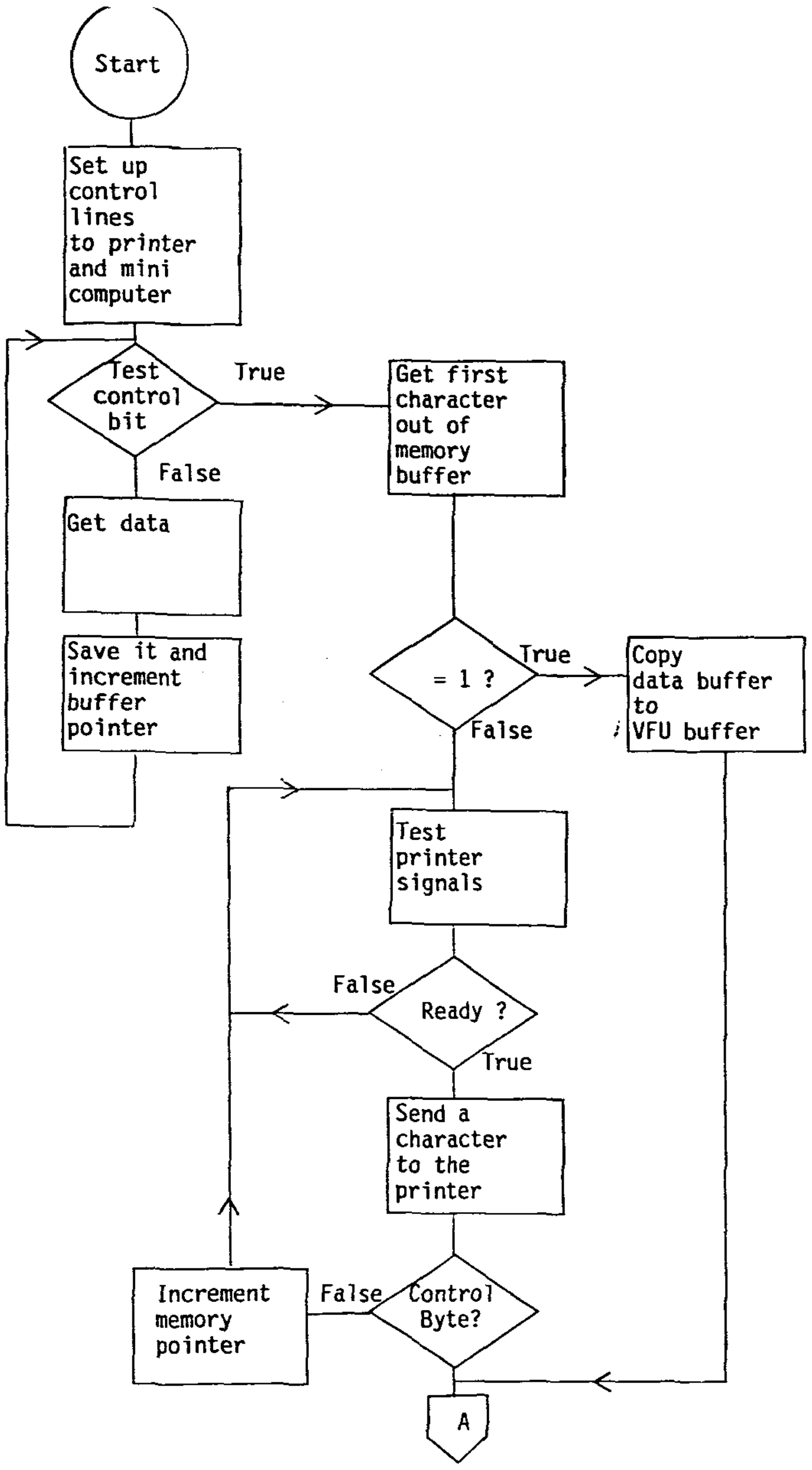

Start
Set up control lines to printer and mini computer
Test control bit
True
False
Get data
Save it and increment buffer pointer
Get first character out of memory buffer
= 1 ?
True
False
Copy data buffer to VFU buffer
Test printer signals
Ready ?
False
True
Send a character to the printer
Control Byte?
False
Increment memory pointer
A

Figure 5.7A The Main Program Flowchart

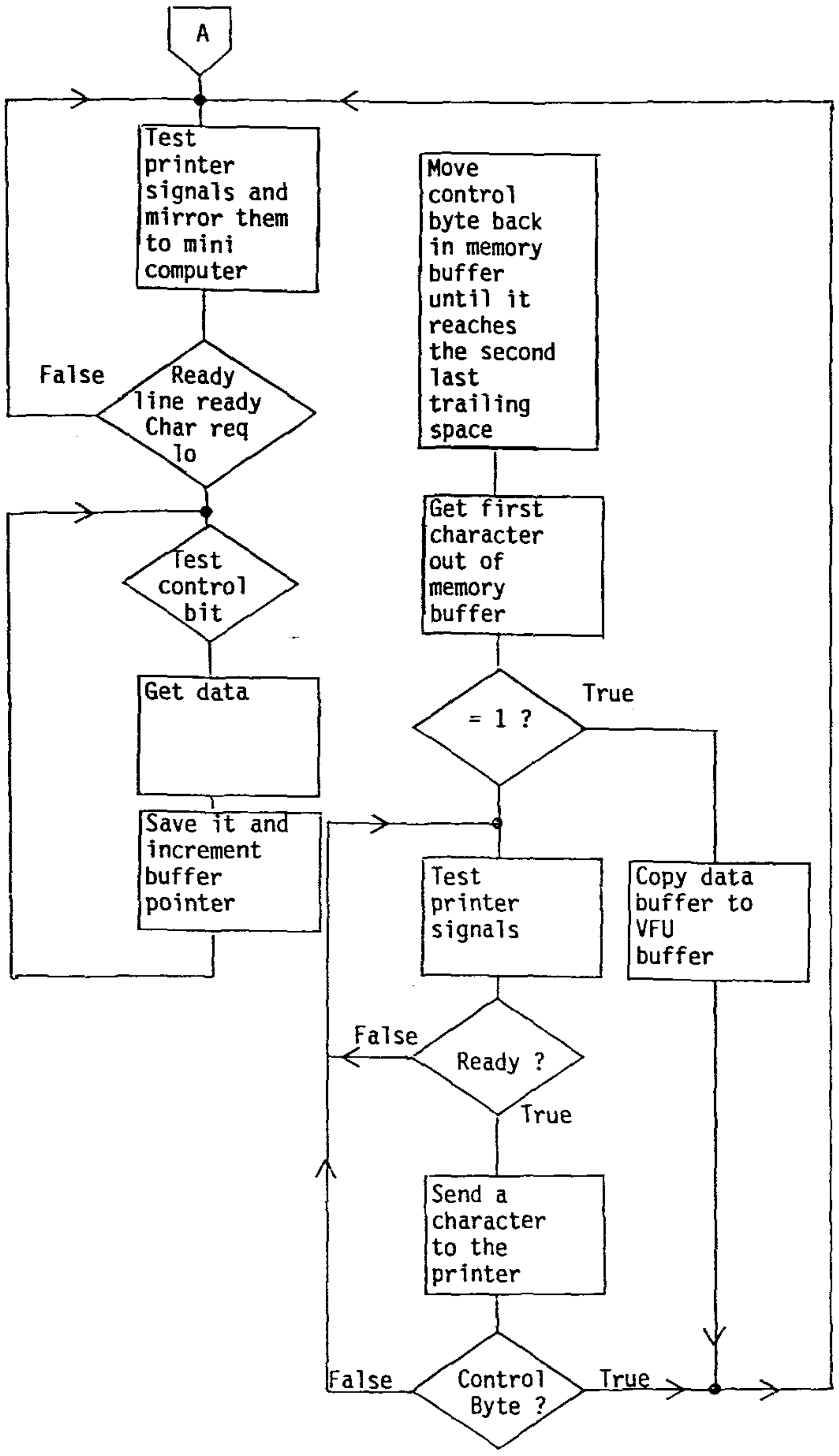

A
Test printer signals and mirror them to mini computer
False
Ready line ready Char req lo
Test control bit
Get data
Save it and increment buffer pointer
Move control byte back in memory buffer until it reaches the second last trailing space
Get first character out of memory buffer
= 1 ?
True
Copy data buffer to VFU buffer
Test printer signals
False
Ready ?
True
Send a character to the printer
False
Control Byte ?
True

Figure 5.7B The Main Program Flowchart (continued)

5.7 Mini Computer Software Modifications

The Mini Computer Programs had to be modified to send the VFU Table to the VFU CPU. The normal start location is location 6 (APPENDIX A Sheet A2) which points to the first routine used. This gets replaced with a pointer to free memory, where the new routine gets inserted. This routine first saves the registers, then sets up the VFU Table pointer and the counter registers. The counter value is defined in a ones complement format as there is no decrement instruction for the mini computer to decrement a register. (Appendix F sheet F1). The routine then gets the data out of the table and sends it to the VFU CPU. Once the data is sent it increments the pointer and counter. If the counter is equal to zero, the routine exits the loop and sends a control bit (ie a DOAP command). If the counter is not zero, the routine loops to send the next word. When the control bit is sent the routine restores the registers and then jumps to the location of normal system startup. The listing shown in the Appendix A Sheet 6 shows the program and a sample VFU Table. The values are all in octal.

5.8 Programing Details

The software was generated on a microcomputer , the SHARPS MZ80K , using an assembler program called 'ZEN'. The resulting code was transferred to a Eprom using a home built Eprom Programer.

6.1 System Operation

The new system worked well with those programs that had been modified to cater for the VFUCPU's abilities. The Programs that had not been modified worked equally well. The new system increased the speed of the daily throughput by a small amount, mainly by reducing the time delay between the printing of the last line of a cheque book and reading the next set of data from the tape deck. Although this time was small, it added up over the course of the working day.

Future developments and Conclusions

7.1 Future Developments

Due to a company policy decision the Mini Computers used to drive the print systems were phased out and replaced with personal computers. This meant a complete rewrite of all our programs and this took priority over anything else, this project included.
The printers themselves are being phased out due to to the lack of available, critical, spare parts.

7.2 Conclusion

The VFU CPU was succesfully interfaced between the Mini Computer and Printer. The VFU CPU was removed when the computer was replaced with a Personal Computer.
The Knowledge gained in designing the VFU CPU was of great benifit in the development of the PC based programs as the printers interface was by then very well known.

References

1 Barden,W (1980) . The Z80 Computer Handbook

2 Control Data Corporation (1975) . Technical Reference
Manual 9352

3 Data Card (1975) . Controller Manual

4 Data Card (1977) . Program Listing of Nedbank Company 100

5 Intel (1981) . Component Data Catalog

6 Mostek . Z80 Microcomputer Devices Technical Manual

7 National Semiconductor Corporation (1984) . Logic
Databook Volume 2

8 Zaks,R (1979) . Programing the Z80,Third Edition

Appendix A

Mini Computer Software

```
10007 .MAIN
                   ;       PAGEZERO.SR                          9/1/77

                   ;       PAGE ZERO MODULE


         000001            .NOCON  1        ;DON'T LIST UNASSEMBLED CODE.
         000000            .LOC    0        ;LOCATIONS 0,2,5 ARE SET FOR RDOS INTERFACE
   00000 000000            0
   00001 000040            ISR              ;START ADDRESS OF INTERRUPTS
         000002            .LOC    2
   00002 000006            PARTS            ; SYSTEM START ADDRESS
         000005            .LOC    5
   00005 000000            0
                   ;
                   ;EOT
```

```
10008 .MAIN
                   ; PART NUMBER MODULE


   00006 002007 PARTS:  JMP      @.+1     ;DISPLAY PART NUMBER
   00007 006433         PARTD




                   ;        BEFORE ASSEMBLING THE PROGRAM CHANGE THE MNEMONICS
                   ;        P1 THROUGH P5 TO THE CORRESPONDING DIGITS OF THE PART
                   ;        NUMBER. CHANGE L2 AND L2 TO THE CORRESPONDING DIGITS
                   ;        OF THE LEVEL NUMBER.
                   ;
                   ;        THE CORRECTION NUMBER IS ONLY CHANGED WHEN PATCHING ON
                   ;        THE MACHINE.
                   ;
                   ;        THE SOFTWARE PREFIX IS NOT CHANGED.


          000010 PART=.
   00010 000011            9.        ;SOFTWARE PREFIX: DO NOT CHANGE
   00011 000001            1.        ;SOFTWARE PREFIX: DO NOT CHANGE

   00012 000000            P1
   00013 000000            P2
   00014 000001            P3
   00015 000003            P4
   00016 000003            P5

   00017 000000            L1
   00020 000001            L2

   00021 000000            0         ;CORRECTION NUMBER: MUST BE CHANGED WHEN PATCHING.

   05642 000736             JMP      SD120             ;CONTINUE HERE
                    ;
         005643 SD500=.
   05643 022415             LDA      0,@.OVF1                   ;FIELD 1 DONE
   05644 101015             MOV#     0,0,SNR
   05645 000403             JMP      SDECX             ;NO.OVERFLOW
                    ;
         005646 SD510=.
   05646 102000             ADC      0,0               ;R0=-1
   05647 000407             JMP      SDERR             ;EXIT HERE
                    ;
         005650 SDECX=.
   05650 022411             LDA      0,@.WRKA                   ;WORK AREA
   05651 026411             LDA      1,@.A12           ;FIELD 1
   05652 030613             LDA      2,WRKL            ;NO. OF DIGITS
                            JSX      CMOVE             ;MOVE IT
   05653 006401             JSR    @.+1
   05654 004600             CMOVE+1
                    ;
   05655 102440             SUBO     0,0               ;CLEAR
                    ;
         005656 SDERR=.
                            .EXIT    2,DECSUB          ;RETURN TO CALLER
   05656 034611             LDA    3,DECSUB
   05657 001403             JMP    2+1,3
                    ;
   05660 005436 .OVF1:  OVF1     ;ADDRESS OF OVF1
   05661 005447 .WRKA:  ADWKA    ;ADDRESS OF ADWKA
   05662 005431 .A12:   A12      ;ADDRESS OF A12



                    ;
                    ;       BYTEPRT.SR
                    ;
                    ;       REVISED TO BE ABLE TO USE BYTES RATHER THAN WORDS
                    ;
                    ;       REVISION:       DATE:           DESCRIPTION:
                    ;
                    ;       00              1/21/76         XFER/A PRINT.SR TO BYTEPRT.SR
                    ;
                    ;       CALLING SEQUENCE:
                    ;       R0=      BYTE ADDRESS OF PRINT BUFFER
                    ;       R1=      NUMBER OF BYTES TO BE PRINTED
                    ;       JSX      BYTEPRT           =MACRO CALL
                    ;
                    ;
                    ;
                    ;      SUBROUTINE BYTEPRT       BYTEPRT.SR
                    ;      -----------------
                    ;

                    ;
                    BYTEPRT:         SBRX              ;SAVE THE RETURN
   05663 000000             0
   05664 054777             STA    3,.-1

   05665 040453             STA      0,PRT1            ;BYTE ADDRESS OF PRINT BUFFER
   05666 044453             STA      1,PRT2            ;# OF CHARACTERS TO BE PRINTED

```

```
0106 .MAIN
      005667 PT110=  .
05667 060417         DIA     0,LPT          ;GET STATUS OF LINE PRINTER
05670 024067         LDA     1,BIT15
05671 106415         SUB#    0,1,SNR        ;WAS STATUS WORD A ONE ?
05672 000414         JMP     PT200
05673 024055         LDA     1,BIT5         ;PRINTER OFF-LINE BIT
05674 067033         DOC     1,SW1          ;TURN PRINTER OFF LINE LIGHT ON
05675 070417 PT100: DIA     2,LPT          ; GET PRINTER READY BIT
05676 151232         MOVZR#  2,2,SZC        ; IS PRINTER READY
05677 000407         JMP     PT200          ; YES - READY
05700 063033         DOC     0,SW1          ; BLINK LIGHT
05701 010025         ISZ     25             ; INCREMENT DELAY COUNTER
05702 000773         JMP     PT100          ; LOOP
05703 123000         ADD     1,0            ;
05704 123400         AND     1,0
05705 000770         JMP     PT100          ; LOOP
05706 102400 PT200: SUB     0,0
05707 063033         DOC     0,SW1          ; TURN OFF-LINE LIGHT OUT
05710 000403         JMP     PT300
05711 102400         SUB     0,0
      005712 PT210=  .
05712 063033         DOC     0,SW1          ;TURN IT OFF AGAIN
             ;
             ;               FIRST TIME THROUGH, BOTH BUSY AND DONE WILL BE ZERO
             ;
05713 063417 PT300: SKPBN   LPT            ;SKIP IF BUSY NOT ZERO
05714 000402         JMP     PT310          ;PRINTER IS DONE WITH LAST LINE, PROCESS NEW
             ;
             ;               HERE WE WAIT UNTIL PRINTER DONE WITH LAST LINE
             ;
05715 000752         JMP     PT110          ; CONTINE
             ;

             ;      TIS TIME TO START THE NEXT LINE
             ;
      005716 PT310=.
05716 024422         LDA     1,PRT1         ;CURRENT BYTE ADDRESS OF PRINT BUFFER
05717 030422         LDA     2,PRT2         ;# OF BYTES TO BE PRINTED THIS LINE
05720 150000         COM     2,2            ;ADJUST CHARACTER COUNT
             ;
             ;       R1=BYTE ADDRESS OF CHARACTER TO BE PRINTED
      005721 PT400=  .
                     JSX     LDBT           ;GET THE FIRST CHARACTER
05721 006401         JSR     @.+1
05722 004655         LDBT+1
             ;       R0=THE CHARACTER TO BE PRINTED

      005723 PT410=  .
05723 063517         SKPBZ   LPT            ;SKIP IF BUSY ZERO
05724 000777         JMP     PT410
05725 061117         DOAS    0,LPT          ;LOAD CHARACTER
05726 125400         INC     1,1            ;BUMP PRINT BUFFFER BYTE ADDRESS
05727 151404         INC     2,2,SZR        ;TALLEY
05730 000771         JMP     PT400
             ;
      005731 PT420=  .
05731 063517         SKPBZ   LPT            ;IS LAST CHARACTER LOADED
05732 000777         JMP     PT420
05733 061317         DOAP    0,LPT          ;YES, INITIALIZE PRINTING
      005734 PT430=  .
```

```
0107 .MAIN
05734 102400          SUB     0,0     ;GO WAIT
05735 006167          JSR     @.SKDA  ;TILL LINE PRINTED.



      005736          .SLPT=.         ;SERVICE LPT (LINE DONE) ENTRY POINT

                      EXIT     0,BYTEPRT        ;RETURN TO CALLER
05736 034725          LDA     3,BYTEPRT
05737 001401          JMP     0+1,3

05740 000000 PRT1:    0                         ;BYTE ADDRESS PRINT BUFFER
05741 000000 PRT2:    0                         ;#NUMBER OF CHARACTERS TO BE PRINTED

             ;*****
```

This program sends the vfu table to the printer
The data table is a demonstration only and is set to two lines at one inch per line with the printer set at 6 lines per inch.

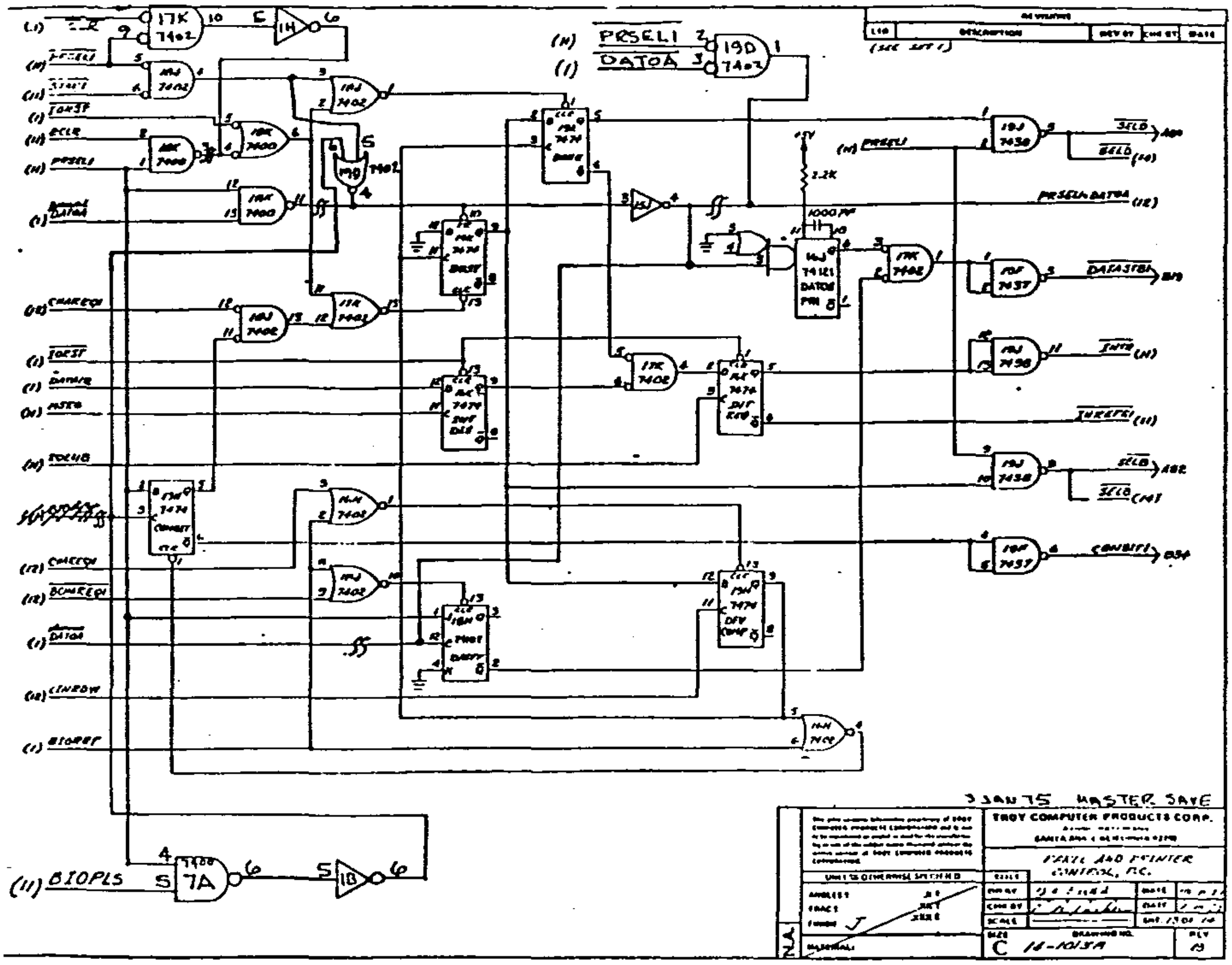
Change location 6 to point to VFU CPU Loader

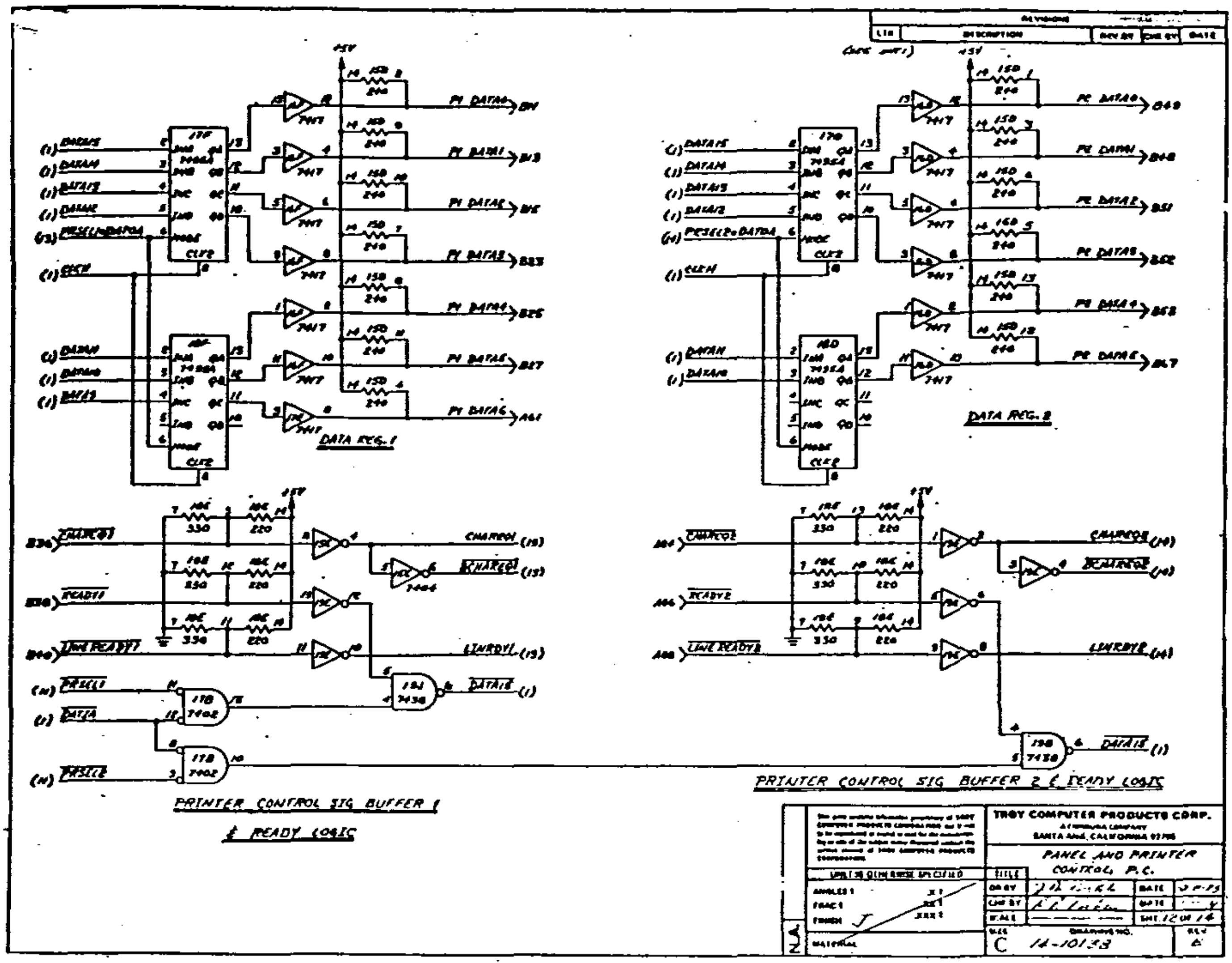
```
    6  11000                                  Start of VFU CPU Loader

11000  054426           STA 3,DUMP3           Save Register 3
11001  050426           STA 2,DUMP2           Save Register 2
11002  044426           STA 1,DUMP1           Save Register 1
11003  040426           STA 0,DUMP0           Save Register 0
11004  024426           LDA 1,COUNT           Get Number Of Words
11005  030426           LDA 2,START           Get Start Of Table
11006  023000  LOOP     LDA 0,@,2             Get Data
11007  063517           SKPBZ LPT             Check Printer Ready
                                              ie VFU CPU
11010  000777           JMP -1                No,Jump -1
11011  061117           DOAS 0,LPT            Send It
10012  151400           INC 2,2               Increment Pointer
10013  165404           INC 1,1 SZR           Increment Counter
10014  000772           JMP LOOP              Do It Again
10015  063517           SKPBZ LPT             Check Printer Ready
10016  000777           JMP -1                No,Jump -1
10017  061317           DOAP 0 LPT            Yes,Send Control Bit
10020  034406           LDA 3,DUMP3           Restore Register 3
10021  030406           LDA 2,DUMP2           Restore Register 2
10022  024406           LDA 1,DUMP1           Restore Register 1
10023  020406           LDA 0,DUMP0           Restore Register 0
10024  002401           JMP @ +1              Jump Indirect +1
10025  006433                                 Start Of Normal Program
10026          DUMP3
10027          DUMP2
10030          DUMP1
10031          DUMP0
10032  177763  COUNT    -15
10033  010034  START
10034  000001  VFU TABLE INDICATOR
10036  000003  LINE THROW
10037  000003
10040  000003
10041  000003
10042  000003
10043  000002  PRINT LINE
10044  000003
10045  000003
10046  000003
10047  000003
10050  000003
10051  000000  TOP OF FORM
```

Appendix B

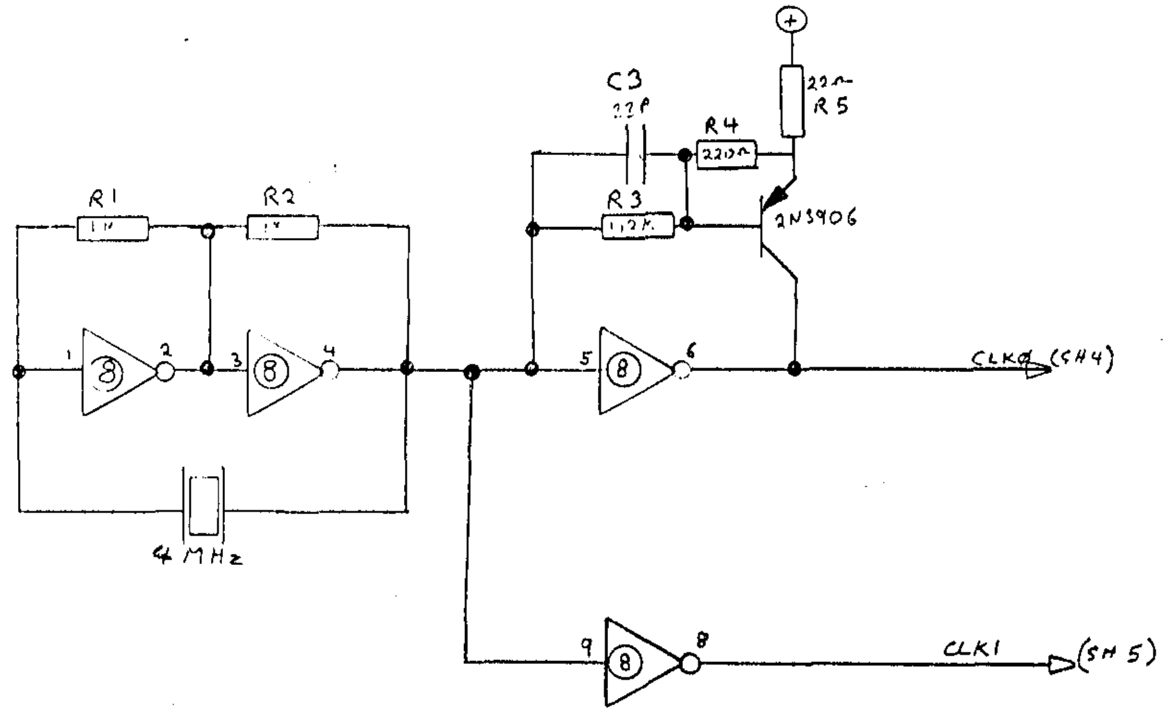
Mini Computer and Printer Interface Circuits


PRSELI
DATOA
19D
7402
17K
7402
14
7A
1B
BIOPLS
PRSELI&DATOA (12)
SELB
DATASTBL
CONBITI
3 JAN 75 MASTER SAVE
TROY COMPUTER PRODUCTS CORP.
SANTA ANA, CALIFORNIA
READER AND PRINTER CONTROL, PC.
C
14-1013A
+5V
2.2K
1000 PF
74121


DATA REG. 1
DATA REG. 2
PRINTER CONTROL SIG BUFFER 1 & READY LOGIC
PRINTER CONTROL SIG BUFFER 2 & READY LOGIC
PI DATA0
PI DATA1
PI DATA2
PI DATA3
PI DATA4
PI DATA5
PI DATA6
P2 DATA0
P2 DATA1
P2 DATA2
P2 DATA3
P2 DATA4
P2 DATA5
CHAREQ1
CHAREQ2
READY1
READY2
LINE READY1
LINE READY2
LINRDY1
LINRDY2
DATA18
DATA15
7495A
7417
7404
7402
7430
+5V
CLCH
TROY COMPUTER PRODUCTS CORP.
SANTA ANA, CALIFORNIA
PANEL AND PRINTER CONTROL P.C.
UNLESS OTHERWISE SPECIFIED
ANGLES
FRAC
FINISH
MATERIAL
TITLE
DR BY
CHK BY
SCALE
DATE
SHT. 12 OF 14
SIZE
C
DRAWING NO.
14-10133
REV

Appendix C

VFU CPU Circuit Diagrams

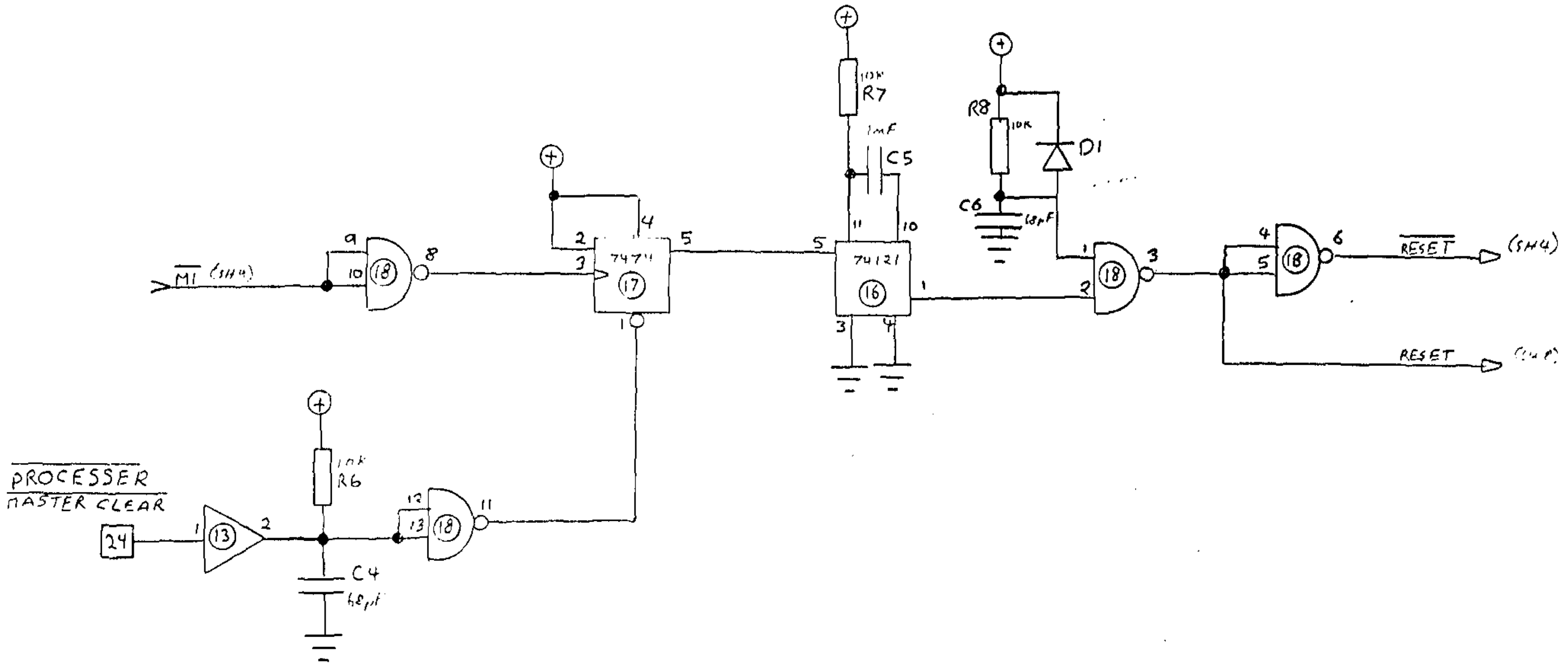


Clock Generator Circuit

Sheet 1 off 12

VFU CPU

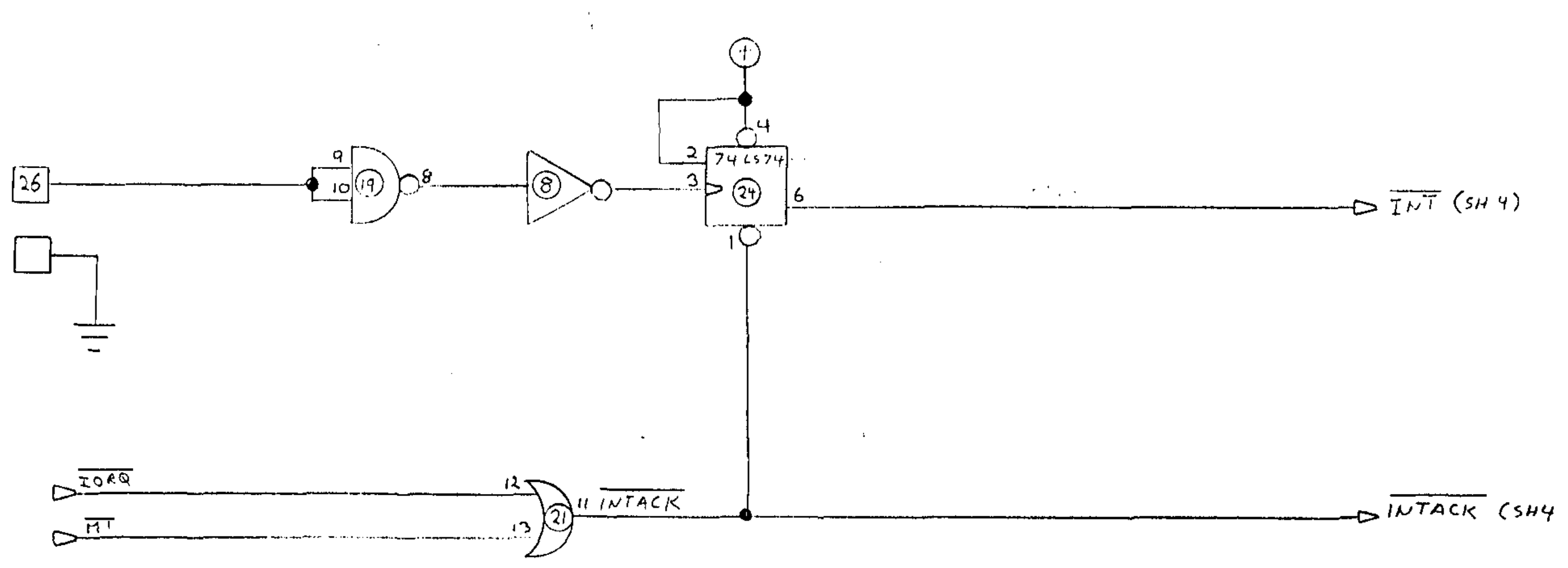
C1



Reset Circuit

Sheet 2 off 12

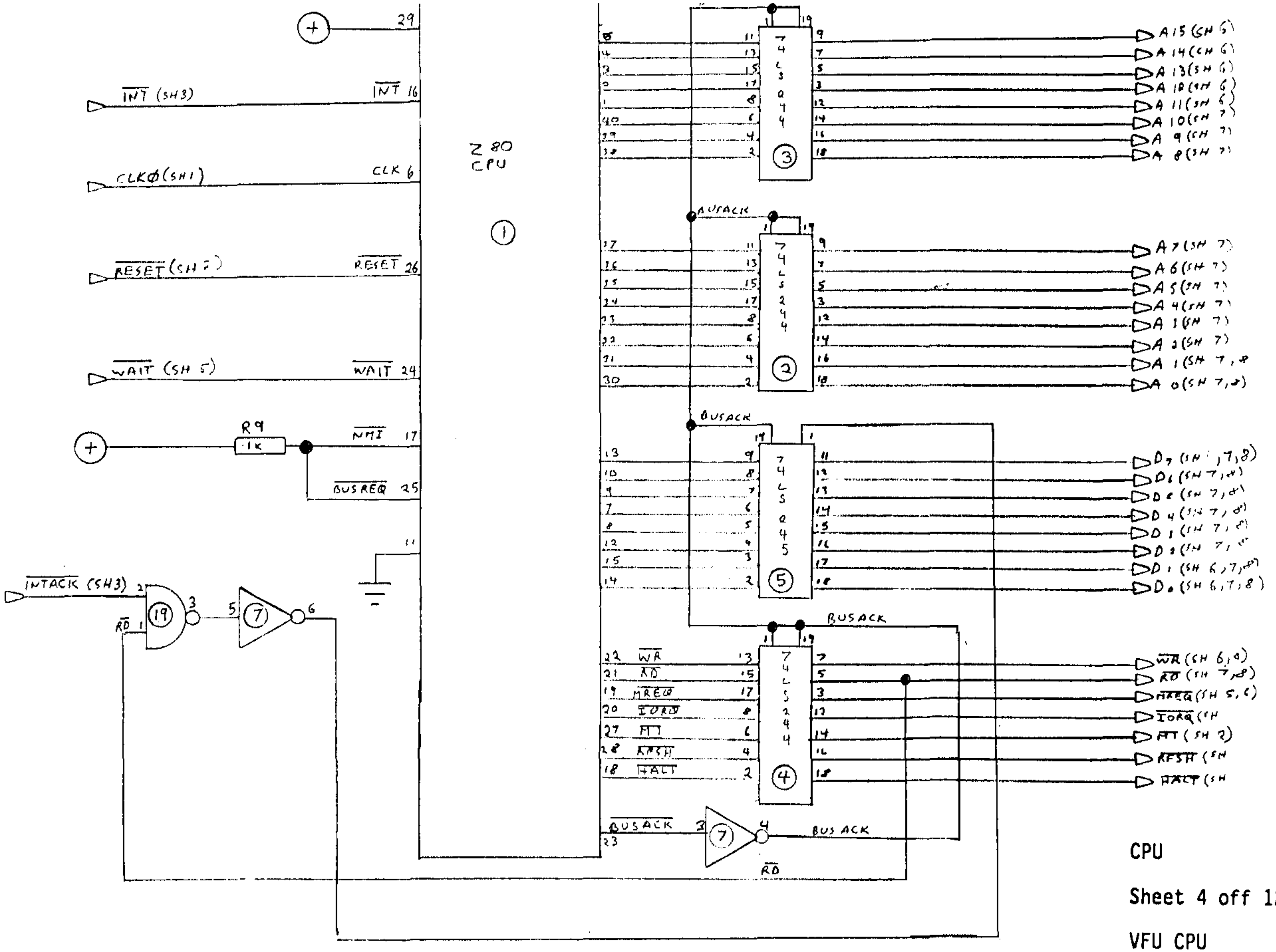
VFU CPU



Interrupt Circuit

Sheet 3 off 12

VFU CPU

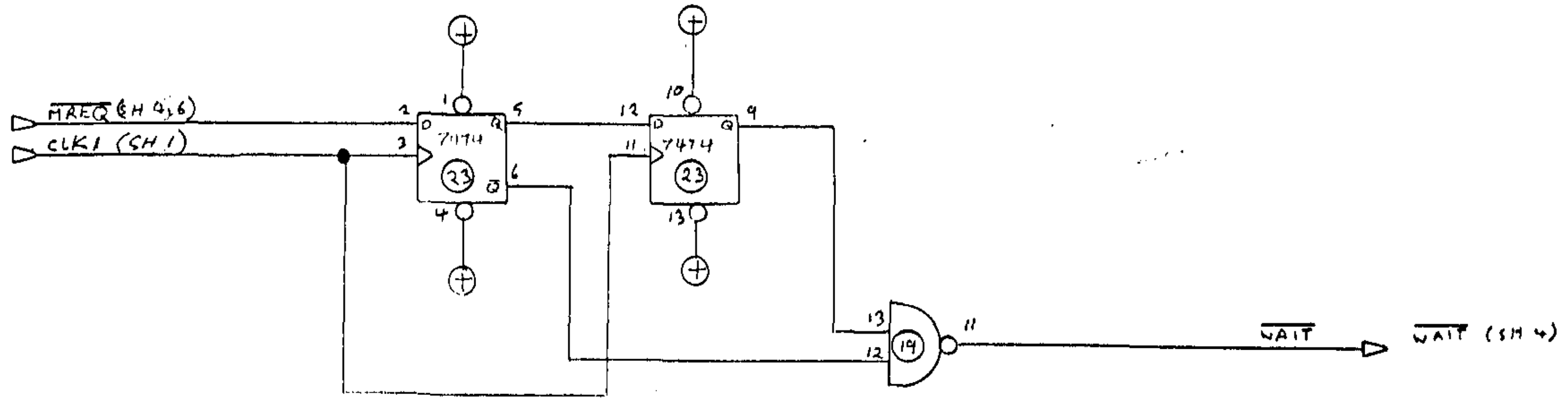


CPU

Sheet 4 off 12

VFU CPU

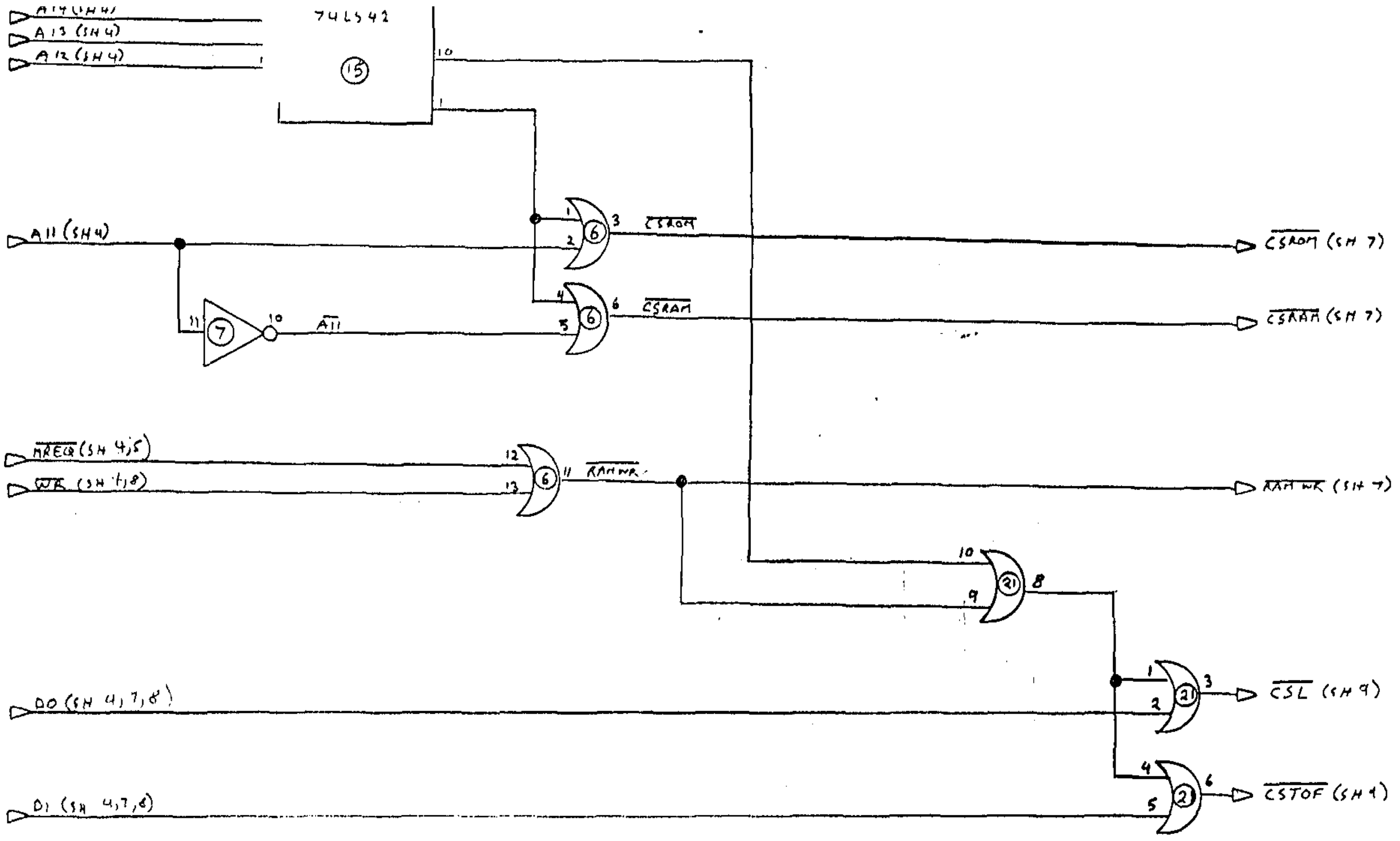
C4



Wait Circuit

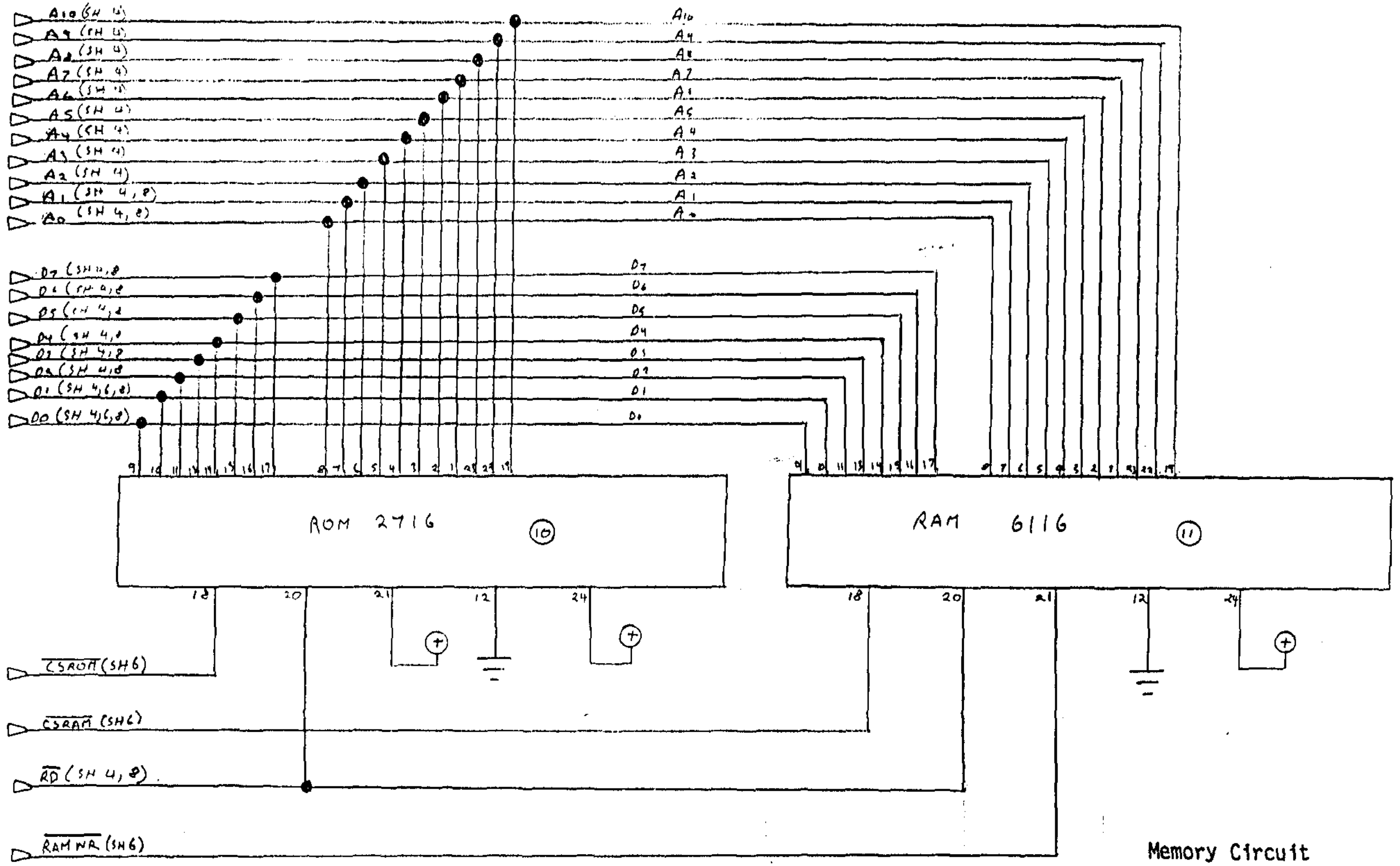
Sheet 5 off 12

VFU CPU



Control and Address Decoder
Sheet 6 of 12

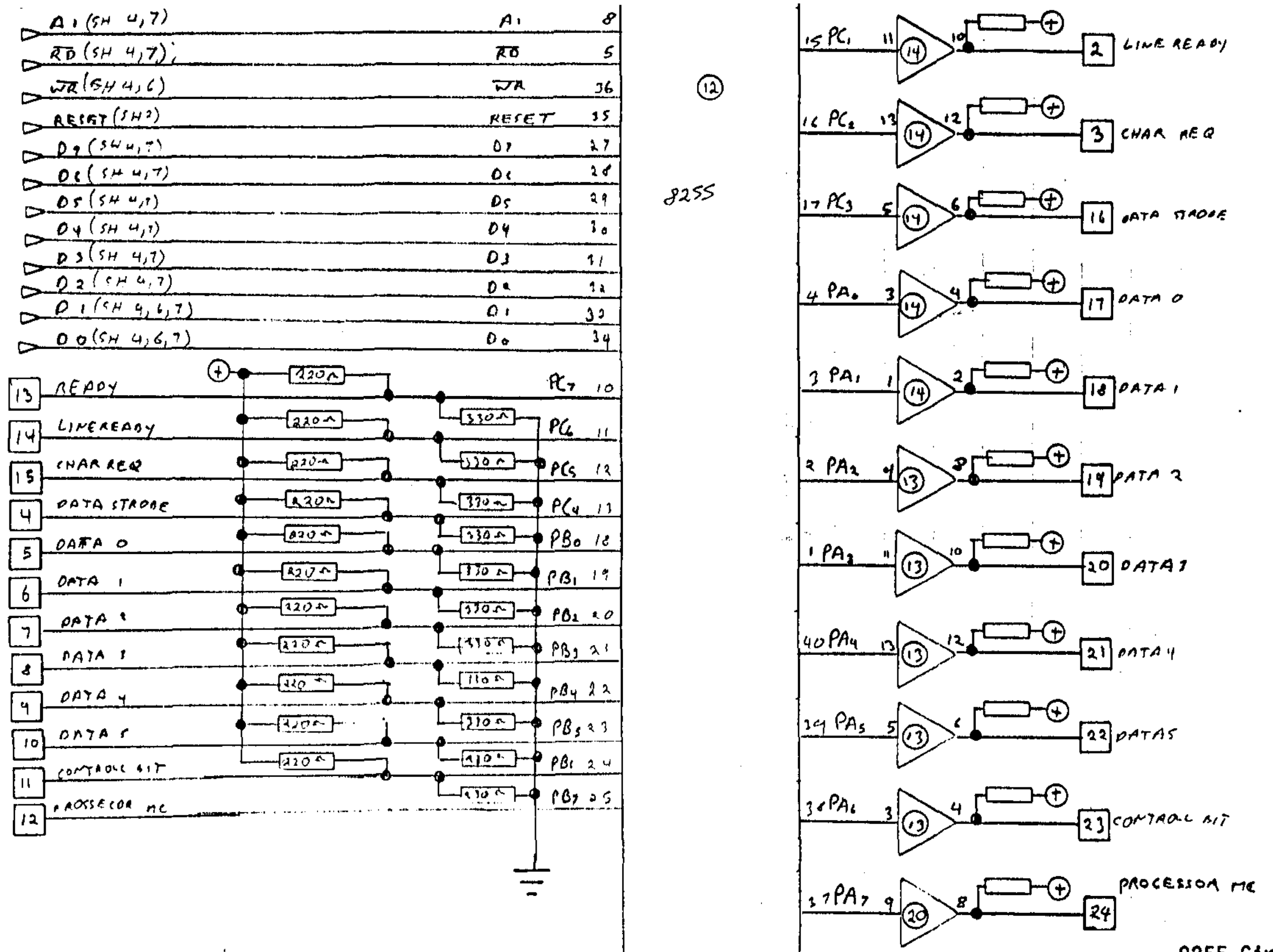
VFU CPU



Memory Circuit

Sheet 7 off 12

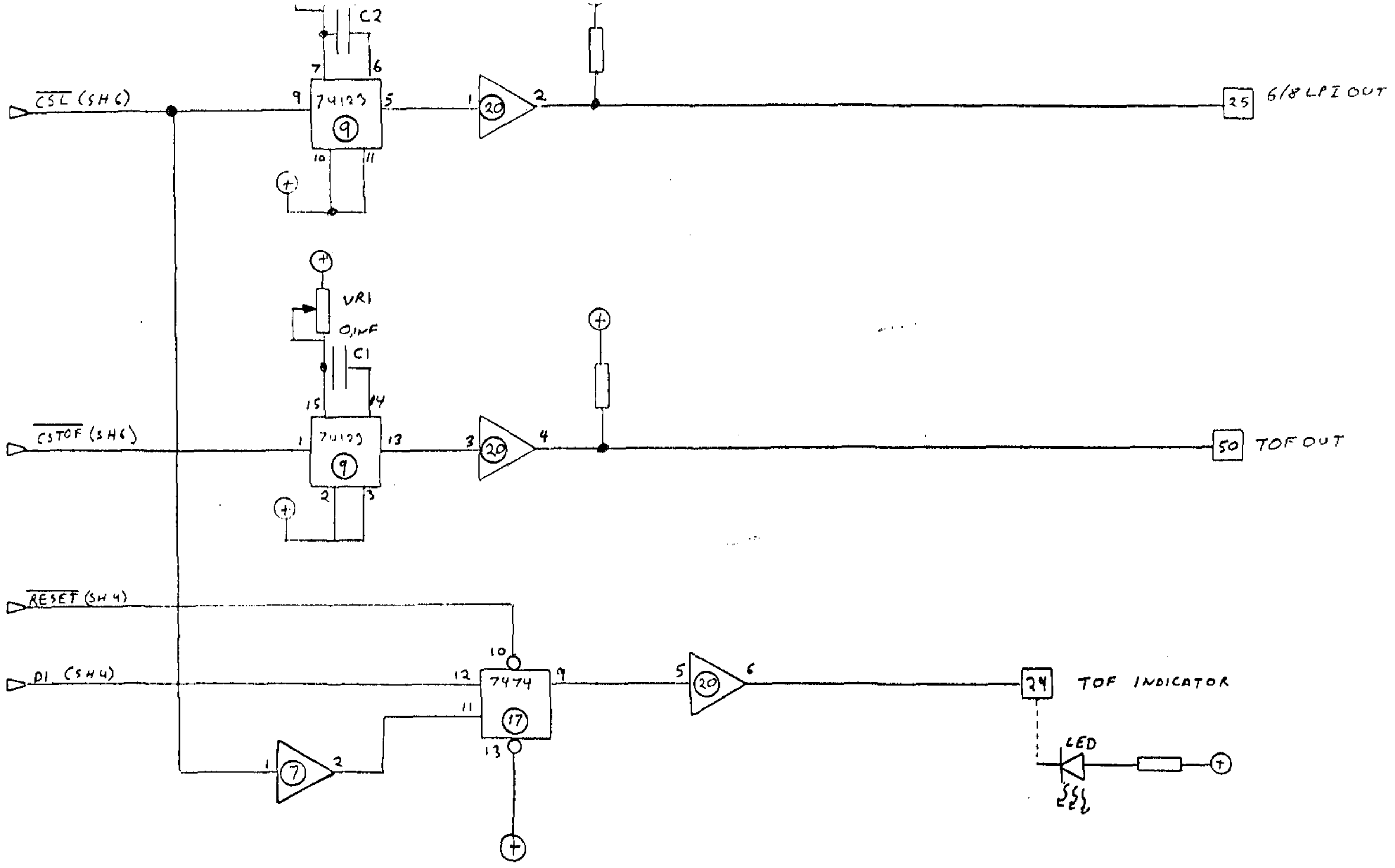
VFU CPU



8255 Circuit

Sheet 8 of 12

VFU CPU



Output and Display

Sheet 9 of 12

VFU CPU

$\boxed{16}$	Ready	▷ M h(j)
$\boxed{3}$	Line Ready	▷ M k(m)
$\boxed{2}$	Char Request	▷ M C(D)
$\boxed{4}$	Data Strobe	▷ M A(B)
$\boxed{5}$	Data 0	▷ M E(F)
$\boxed{6}$	Data 1	▷ M H(J)
$\boxed{7}$	Data 2	▷ M K(L)
$\boxed{8}$	Data 3	▷ M M(N)
$\boxed{9}$	Data 4	▷ M P(R)
$\boxed{10}$	Data 5	▷ M S(T)
$\boxed{51}$	Data 6	▷ M (Not Conected)
$\boxed{11}$	Control bit	▷ M a(b)
$\boxed{12}$	Processor MC	▷ M n(p)
$\boxed{27}$	+ 5 Volt	▷ M FF(HH)

$\boxed{13}$	Ready	▷ F h(j)
$\boxed{14}$	Line Ready	▷ F k(m)
$\boxed{15}$	Char Request	▷ F C(D)
$\boxed{1}$	Data Strobe	▷ F A(B)
$\boxed{17}$	Data 0	▷ F E(F)
$\boxed{18}$	Data 1	▷ F H(J)
$\boxed{19}$	Data 2	▷ F K(L)
$\boxed{20}$	Data 3	▷ F M(N)
$\boxed{21}$	Data 4	▷ F P(R)
$\boxed{22}$	Data 5	▷ F S(T)
$\boxed{52}$	Data 6	▷ F (Not Conected)
$\boxed{23}$	Control bit	▷ F a(b)
$\boxed{24}$	Processor MC	▷ F z(y)
$\boxed{25}$	6 / 8 LPI Out	▷ F DD(EE)
$\boxed{26}$	6 / 8 LPI In	▷ F BB(CC)
$\boxed{28}$	+ 5 Volt	▷ F FF(HH)

Note 1: ☐ Edge Connector

Note 2: ▷ Male (M) or Female (F) Amphenol Plug

Note 3 Characters in brackets denote
Twisted Pair Earth Returns

Connector Cables

Sheet 10 of 12

VFU CPU

Connect jumper into female plug (from computer)
BB to DD , CC to EE
This is to ensure 6/8 LPI continuity should the VFU CPU be removed

Connect jumper into Test Boxes
BB to DD , CC to EE

Connect pin z to Board A4B10 Pin B10 to allow power on reset

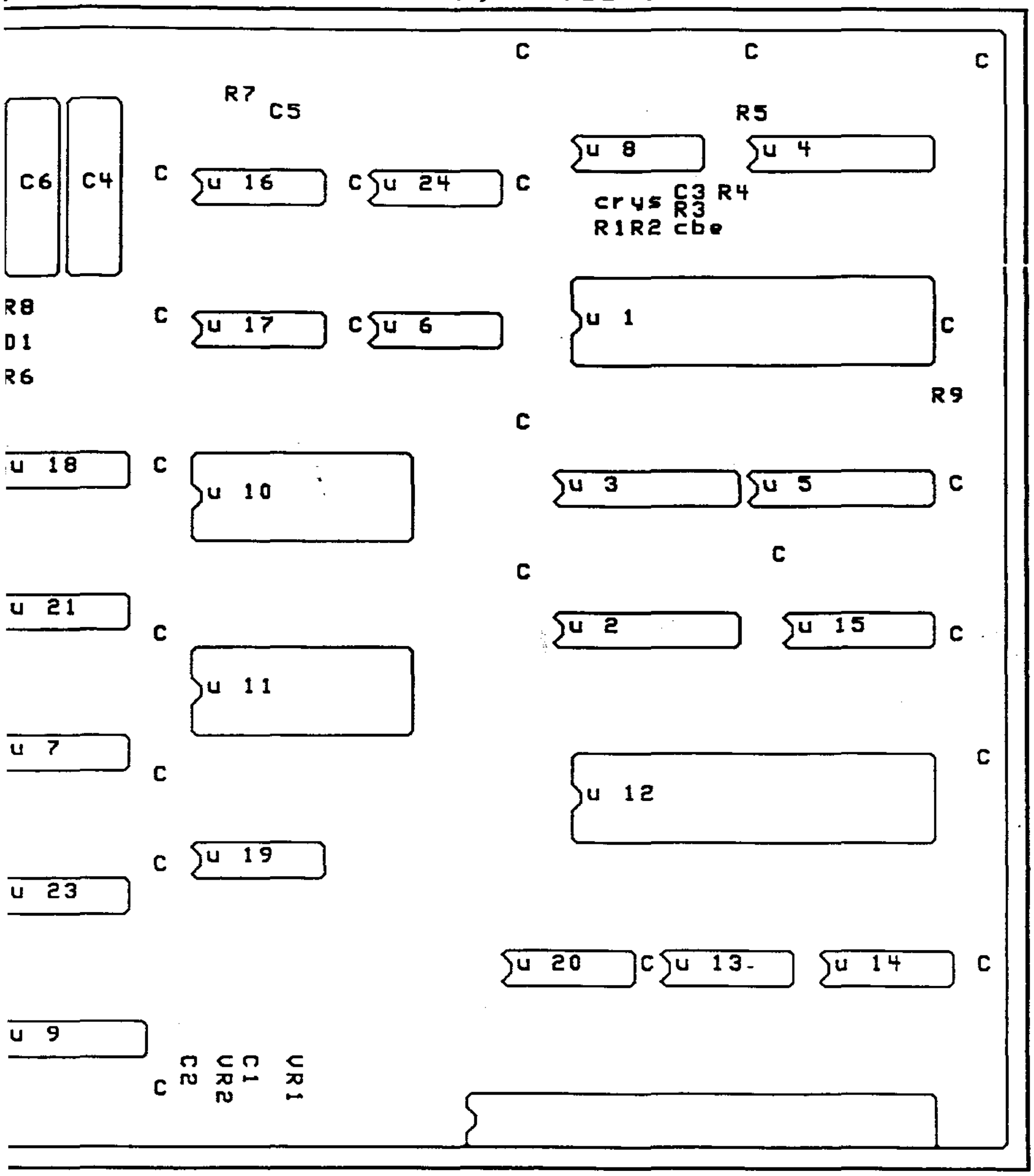
Chip Number	Type	Gates used	Description
1	Z80	-	CPU
2	74LS244	All	Buffer
3	74LS244	All	Buffer
4	74LS244	7 off 8	Buffer
5	74LS245	All	Bidirectional Buffer
6	74LS32	All	Quad 2 Input OR
7	74LS04	4	Hex Inverter
8	74LS04	All	Hex Inverter
9	74123	All	Dual Monostable
10	2716	-	EPROM
11	6116	-	Static RAM
12	8255	-	Interface
13	7417	All	Hex Open Collector Buffers
14	7417	All	Hex Open Collector Buffers
15	74LS42	All	BCD 4 to 10 Line Decoder
16	74121	All	Monostable
17	7474	All	Flip Flop
18	74132	All	Quad Nand
19	74LS00	3	Quad Nand
20	7417	4	Hex Open Collector Buffers
21	74LS32	3	Quad 2 Input OR
22			Not Used
23	74LS74	All	Dual Flip Flops
24	74LS74	All	Dual Flip Flops

Chip Count

Sheet 12 off 12

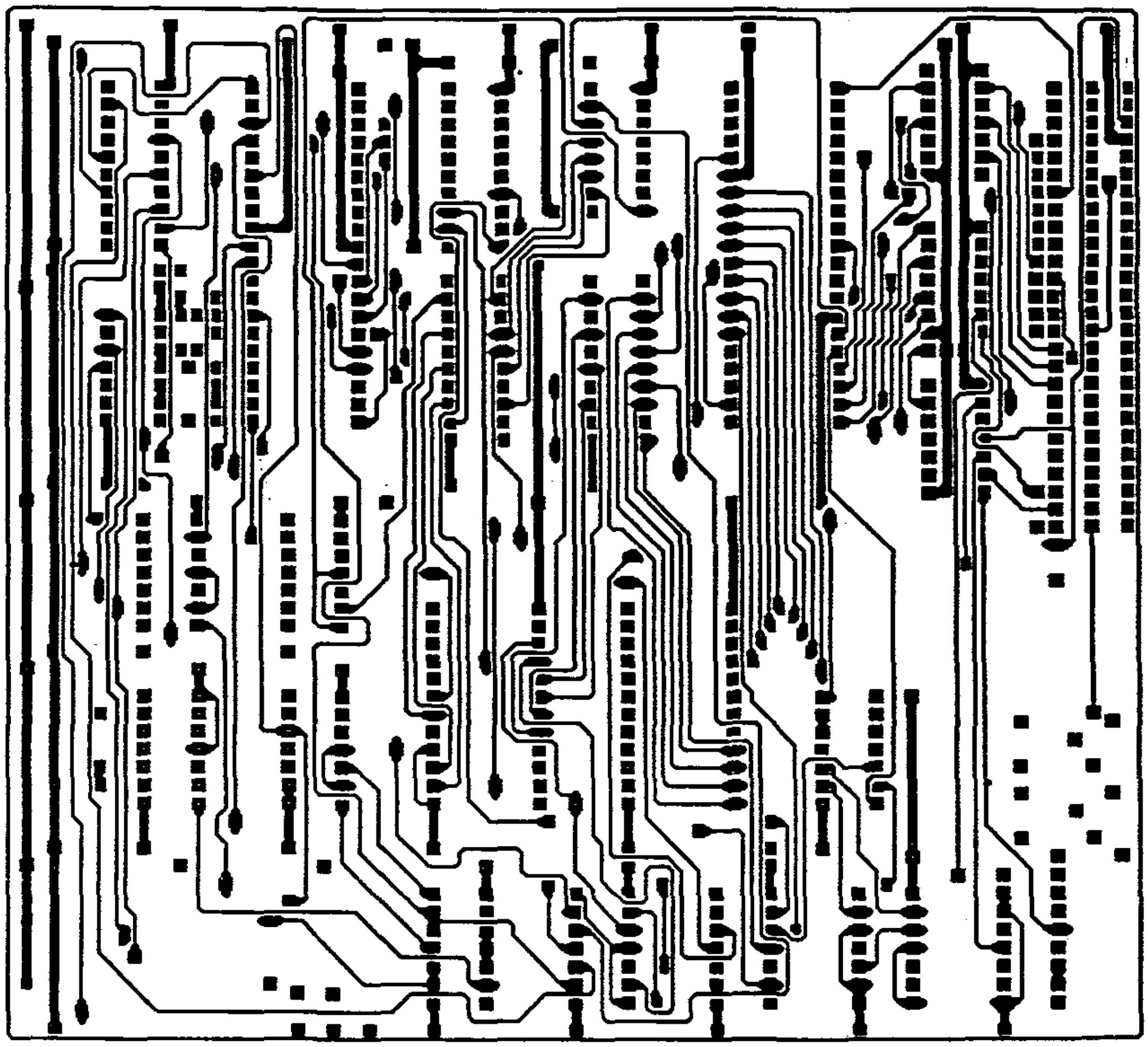
VFU CPU

o27.pcb
.2 r3 holes: 718 silkscreen
proximate size: 6.35 by 5.85 inches

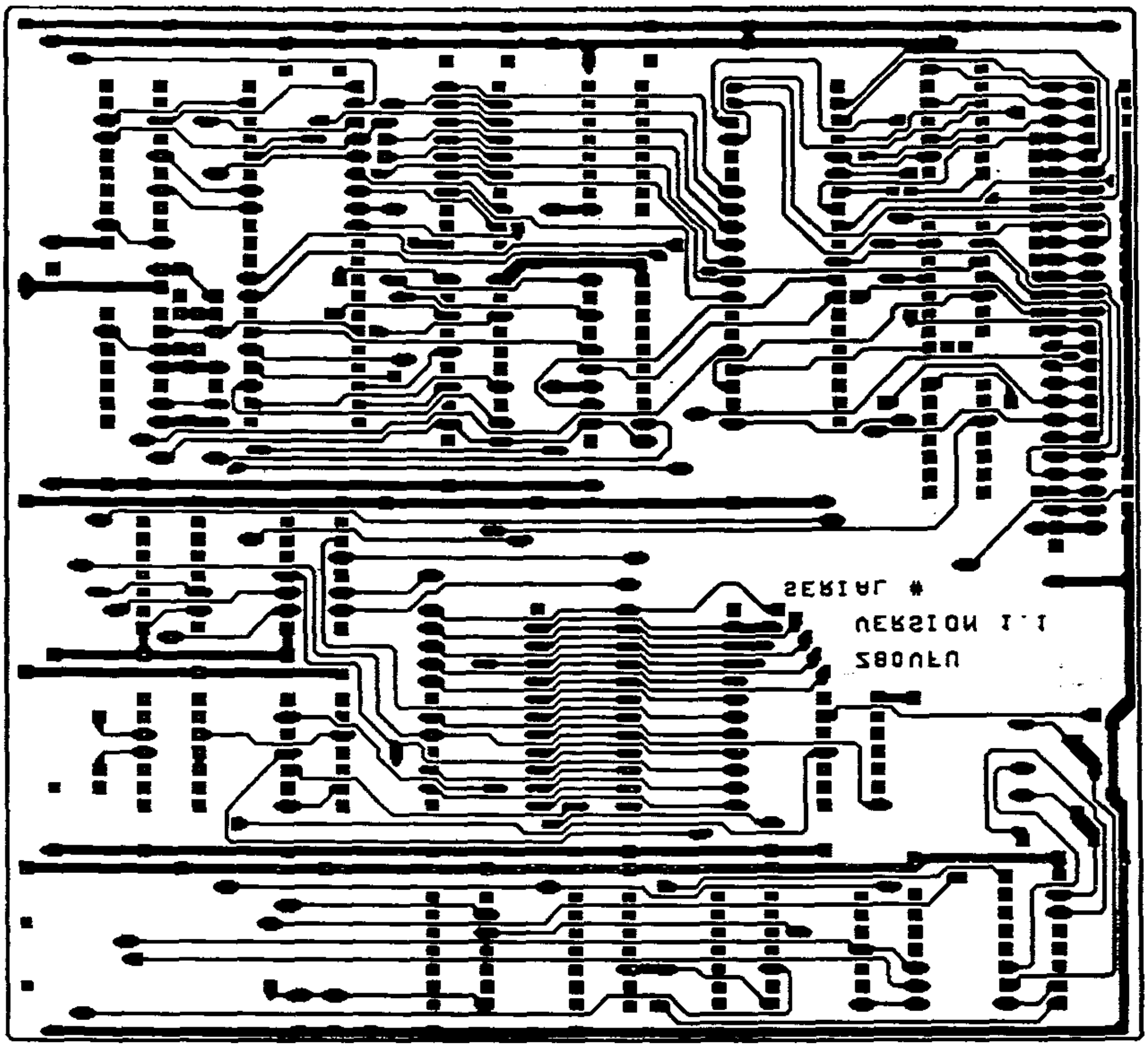


C13

```
2X artwork             1 Jan    80    00:52:20
z8027.pcb
v1.3 r3 holes:  718        component side
approximate size:  6.35 by  5.85 inches
```



```
2X artwork              1 Jan    80    01:10:54
z8027.pcb
v1.3 r3 holes:   718             solder side
approximate size:  6.35 by  5.85 inches
```



Appendix D

VFU CPU Software Listing

```
                              ORG   0000H
                              LOAD  6000H
                   ;
                   ;DATE 1987.8.14
                   ;TIME 9.20
                   ;TITLE. NEW 2
                   ;TAPE NO. 2
                   ;START. 357
                   ;
                   ;MEMORY
                   ;
                   ;
                   ;ROM=0000-07FF
                   ;RAM=0800-0FFF
                   ;8255=9000-9003
                   ;6/8 OUT=8000
                   ;
                   ;
                   ;RAM BREAKDOWN
                   ;
                   ;PRINTER BUFFER=0800-0E00
                   ;VFU BUFFER    =0E01-0F00
                   ;STACK POINTER =0F02-0FFF
                   ;COMMAND       =0F01
                   ;
                   VFCH:      EQU   9H
                   OUT:       EQU   9000H
                   OUTC:      EQU   9002H
                   IN:        EQU   9001H
                   INC:       EQU   9002H
                   START:     EQU   0800H
                   STOP:      EQU   0E00H
                   COMMAND:   EQU   0F01H
                   VFUSTART:  EQU   0E01H
                   VFUSTOP:   EQU   0F00H
                   ;
                   ;
                   ;
                   ;INITIALIZE
                   ;
                   ;
0000 31FF0F                   LD    SP,0FFFH          ;SET STACK POINTER
0003 ED56                     IM    1                 ;SET INTERRUPT MODE 1
0005 3E8A                     LD    A,8AH             ;8255 CONTROLL WORD 6
0007 320390                   LD    (9003H),A         ;A,C0-3=OUT,B,C4-7=IN
000A 110008                   LD    DE,START          ;BUFFER START
000D 210008                   LD    HL,START          ;BUFFER START
0010 01000E                   LD    BC,STOP           ;BUFFER STOP
0013 D9                       EXX                     ;EXCHANGE REGISTERS
0014 11010E                   LD    DE,VFUSTART       ;VFU BUFFER START
0017 01000F                   LD    BC,VFUSTOP        ;VFU BUFFER STOP
001A D9                       EXX                     ;EXCHANGE REGISTERS AGAIN
001B 3EFF                     LD    A,0FFH
001D 32010F                   LD    (COMMAND),A       ;SET COMMAND HI
0020 CD4E00                   CALL  CLEAR             ;CLEAR MEMORY
0023 FB                       EI                      ;ENABLE INTERUPT
0024 210290                   LD    HL,OUTC           ;SET FOR OUTPUT
0027 CBC6                     SET   0,(HL)            ;SET STROBE PRINTER
0029 CBCE                     SET   1,(HL)            ;SET CHAR REQ D116
002B CBD6                     SET   2,(HL)            ;SET LINE READY D116
```

```
002D CBDE                SET  3,(HL)          ;SET READY D116
002F C37500              JP   VSTART          ;INPUT/OUTPUT
              ;
              ;
              ;LEAVE OUT THIS AREA
                         DS   06H
              ;
              ;
              ;START OF INTERRUPT
              ;
              ;
0038 F3                  DI                   ;DISABLE INTERRUPTS
0039 F5                  PUSH AF
003A D9                  EXX                  ;EXCHANGE REGISTERS
003B 1A                  LD   A,(DE)          ;GET DATA
003C 320080              LD   (8000H),A       ;OUTPUT IT TO 6/8LPI
003F CB4F                BIT  1,A             ;TEST FOR TOF
0041 2806                JR   Z,TOF
0043 13                  INC  DE
0044 D9       LEAVE:     EXX                  ;EXCHANGE REGISTERS
0045 F1                  POP  AF
0046 FB                  EI                   ;ENABLE INTERRUPTS
0047 ED4D                RETI                 ;RETURN FROM INTERRUPT
0049 11010E   TOF:       LD   DE,VFUSTART     ;RESET POINTER
004C 18F6                JR   LEAVE
              ;
              ;
              ;CLEAR MEMORY BUFFER
              ;IE FILL UP WITH FFH
              ;AND FILL VFU BUFFER
              ;WITH "00"
              ;
              ;
004E D5       CLEAR:     PUSH DE
004F F5                  PUSH AF
0050 C5                  PUSH BC
0051 3EFF     LOOPC:     LD   A,0FFH
0053 12                  LD   (DE),A
0054 13                  INC  DE
0055 CD6E00              CALL COMPARE
0058 C25100              JP   NZ,LOOPC
              ;
              ;
005B 01000F              LD   BC,VFUSTOP
005E 11010E              LD   DE,VFUSTART
0061 3E00     LOOPC2:    LD   A,00
0063 12                  LD   (DE),A
0064 13                  INC  DE
0065 CD6E00              CALL COMPARE
0068 20F7                JR   NZ,LOOPC2
006A C1                  POP  BC
006B F1                  POP  AF
006C D1                  POP  DE
006D C9                  RET
              ;
              ;
              ;
              ;
              ;
              ;
```

```
                ;
                ;
                ;
                ;COMPARE DE TO BC
                ;ACC A DESTROYED
                ;Z=0 IF EQUAL
                ;
006E 7A         COMPARE:    LD    A,D
006F B8                     CP    B                 ;COMPARE D&B
0070 2002                   JR    NZ,HOME
0072 7B                     LD    A,E
0073 B9                     CP    C                 ;COMPARE E&C
0074 C9         HOME:       RET                     ;RETURN TO CALLER
                ;
                ;
0075 210290     VSTART:     LD    HL,INC            ;SET UP INPUT PORT
0078 110008                 LD    DE,START          ;SET UP BUFFER START
007B CB9E                   RES   3,(HL)            ;RESET READY OUT
007D CB96                   RES   2,(HL)            ;RESET LINE READY OUT
007F CBCE       INPUTX:     SET   1,(HL)            ;SET CHARREQ
0081 CB8E                   RES   1,(HL)            ;RESET CHARREQ
0083 3A0190                 LD    A,(IN)            ;GET DATA
0086 CB77                   BIT   6,A               ;TEST FOR CONTROLL BIT
0088 200C                   JR    NZ,EOLINE         ;YES,IT IS
008A 7E         INPUTY:     LD    A,(HL)            ;NO,GET FLAGS
008B CB67                   BIT   4,A               ;TEST STROBE
008D 20FB                   JR    NZ,INPUTY         ;NO,LOOP BACK,&
                                                                WAIT
008F 3A0190                 LD    A,(IN)            ;GET DATA
0092 12                     LD    (DE),A            ;AND SAVE IT
0093 13                     INC   DE                ;INCREMENT POINTER
0094 18E9                   JR    INPUTX            ;GET NEXT CHARACTER
0096 110008     EOLINE:     LD    DE,START
0099 1A                     LD    A,(DE)            ;GET FIRST CHARACTER
009A FE01                   CP    1                 ;IS IT A 1
009C 2877                   JR    Z,SETVFU          ;YES,SET VFU TABLE
009E 7E         LOOPCR:     LD    A,(HL)            ;WAIT FOR PRINTER
009F CB6F                   BIT   5,A               ;TESTCHARREQ
00A1 20FB                   JR    NZ,LOOPCR         ;NOT READY
00A3 1840                   JR    OUTPUT1           ;YES,READY
                ;
                ;
00A5 210290     LOOP:       LD    HL,INC
00A8 110008                 LD    DE,START          ;MEMORY BUFFER START
00AB 7E         TESTRI:     LD    A,(HL)            ;TEST READY IN
00AC CB7F                   BIT   7,A
00AE 204F                   JR    NZ,SETRO          ;ITS A 1,SET READY OUT
0080 CB9E                   RES   3,(HL)            ;ITS A 0,TELL D116
                ;
                ;
0082 7E         TESTLRI:    LD    A,(HL)            ;TEST LINEREADYIN
0083 CB77                   BIT   6,A
0085 2050                   JR    NZ,SETLRO         ;ITS A 1,SET LINERDOUT
0087 CB96                   RES   2,(HL)            ;ITS A 0,TELL D116
                ;
                ;
                ;
                ;
00B9 7E         TESTCRI:    LD    A,(HL)            ;TEST CHAR.REQST
008A CB6F                   BIT   5,A
008C 204F                   JR    NZ,SETCRO         ;ITS A 1,SET CHARREQO
```

D3

```
00BE CBCE      INPUT:      SET   1,(HL)            ;MAKE IT HI
00C0 CB8E                  RES   1,(HL)            ;NOW MAKE IT LO
00C2 3A0190                LD    A,(IN)            ;TEST CONTROLL BIT
00C5 CB77                  BIT   6,A
00C7 200C                  JR    NZ,OUTPUT4        ;ITS A 1,END LINE INP
00C9 7E        INPUT2:     LD    A,(HL)            ;TEST STROBE IN
00CA CB67                  BIT   4,A
00CC 20FB                  JR    NZ,INPUT2         ;NOT YET SO LOOP BACK
               ;
00CE 3A0190                LD    A,(IN)            ;GET DATA
00D1 12                    LD    (DE),A            ;STORE IN MEMORY
00D2 13                    INC   DE                ;INCREMENT MEMORY POINTER
00D3 18E9                  JR    INPUT
               ;
00D5 12        OUTPUT4:    LD    (DE),A
00D6 47                    LD    B,A               ;SAVE CONTROLL WORD
00D7 1B        REDUCE1:    DEC   DE                ;DECREMENT MEMPOINTR
00D8 1A                    LD    A,(DE)            ;GET DATA
00D9 FE20                  CP    400               ;CHECK FOR A SPACE
00DB 2834                  JR    Z,REDUCE2         ;REPLACE WITH CONTRWRD
               ;
               ;
00DD 110008                LD    DE,START          ;GET START OF BUFFER
00E0 1A                    LD    A,(DE)            ;CHECK DATA
00E1 FE01                  CP    1                 ;COMPARE TO 1
00E3 2830                  JR    Z,SETVFU          ;YES,SET UP VFU TABLE
               ;
               ;
00E5 CBD6      OUTPUT1:    SET   2,(HL)            ;SET LINE READY
00E7 110008    OUTPUT:     LD    DE,START          ;RESET DE
00EA 1A        OUTPUT3:    LD    A,(DE)            ;GET DATA
00EB 320090                LD    (OUT),A           ;SEND IT
00EE CB86                  RES   0,(HL)            ;RESET STROBE
00F0 7E        CRIN:       LD    A,(HL)            ;CHECK FLAGS
00F1 CB6F                  BIT   5,A               ;WAIT FOR CHARREQ
00F3 28FB                  JR    Z,CRIN            ;WAIT FOR CHARREQ
00F5 CBC6                  SET   0,(HL)            ;SET STROBE
00F7 1A                    LD    A,(DE)            ;GET DATA AGAIN
00F8 CB77                  BIT   6,A               ;CHECK FOR CONTROLBIT
00FA 20A9                  JR    NZ,LOOP           ;YES,EOL,START NEWLINE
               ;
00FC 13                    INC   DE                ;NO,INCREMENT POINTER
00FD 18EB                  JR    OUTPUT3           ;AND LOOP
               ;
00FF CBDE      SETRO:      SET   3,(HL)            ;SET READY
0101 CBD6                  SET   2,(HL)            ;SET LINE READY
0103 CBCE                  SET   1,(HL)            ;SET CHARREQ
0105 18A4                  JR    TESTRI
               ;
               ;
0107 CBD6      SETLRO:     SET   2,(HL)            ;SET LINE READY
0109 CBCE                  SET   1,(HL)            ;SET CHARREQ
010B 189E                  JR    TESTRI
               ;
               ;
010D CBCE      SETCRO:     SET   1,(HL)            ;SET CHARREQ
010F 189A                  JR    TESTRI
               ;
               ;
               ;
```

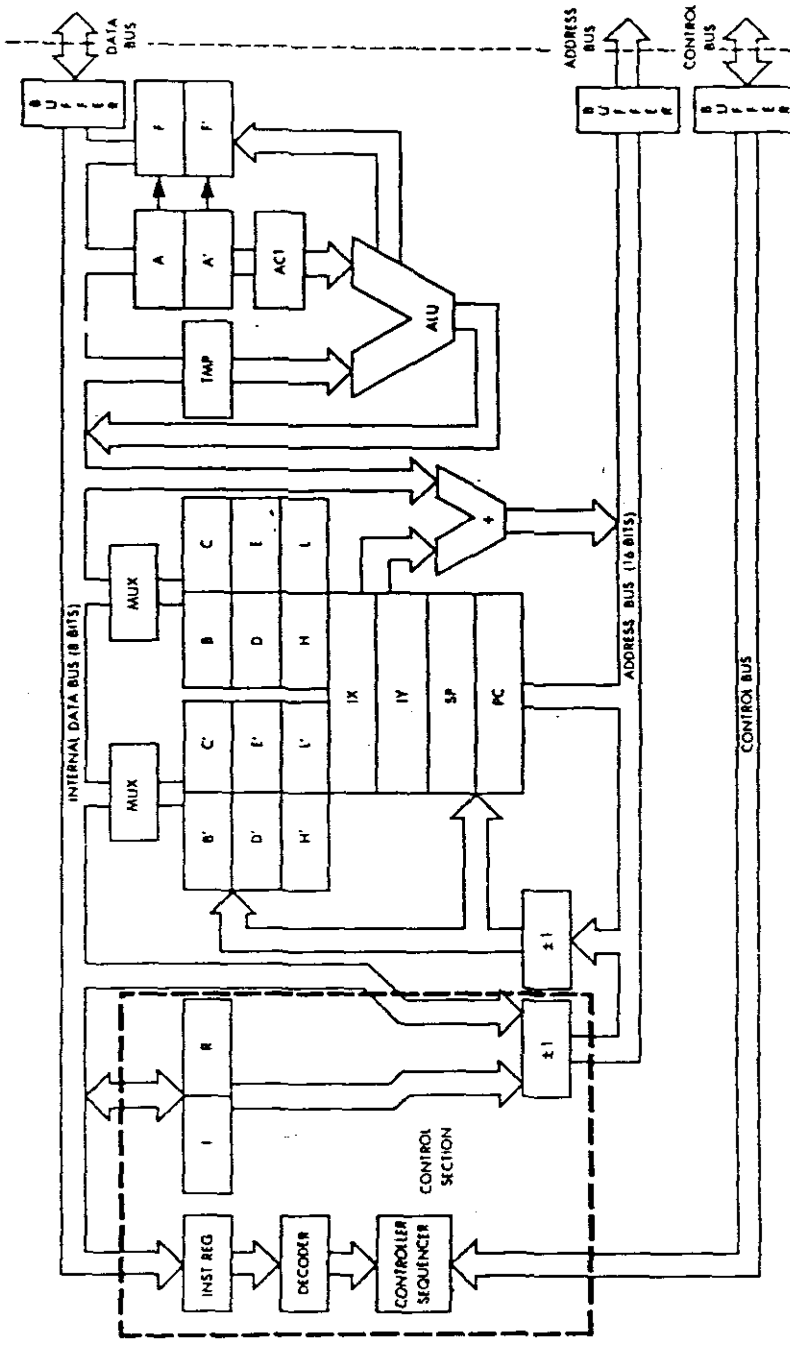
```
0111 47        REDUCE2:    LD    B,A              ;GET CONTROLL WORD
0112 12                    LD    (DE),A           ;AND STORE IT
0113 18C2                  JR    REDUCE1          ;REPEAT LOOP
               ;
               ;
0115 11010E    SETVFU:     LD    DE,VFUSTART      ;TABLE START
0118 010108                LD    BC,START+1       ;BUFFER START+1(DATA)
011B 0A        LOOPVFU:    LD    A,(BC)           ;GET DATA
011C 12                    LD    (DE),A           ;STORE IT
011D 13                    INC   DE
011E 03                    INC   BC
011F CB77                  BIT   6,A              ;TEST FOR CONTROLL BIT
0121 28F8                  JR    Z,LOOPVFU
0123 C3A500                JP    LOOP
               ;
               ;
                           END
```

```
PAGE    1

COMMAND      0F01 CLEAR       004E COMPARE     006E CRIN        00F0
EOLINE       0096 HOME        0074 IN          9001 INC         9002
INPUTX       007F INPUTY      008A INPUT       00BE INPUT2      00C9
LEAVE        0044 LOOPC       0051 LOOPC2      0061 LOOPCR      009E
LOOP         00A5 LOOPVFU     0118 OUT         9000 OUTC        9002
OUTPUT4      00D5 OUTPUT1     00E5 OUTPUT      00E7 OUTPUT3     00EA
REDUCE1      00D7 REDUCE2     0111 START       0800 STOP        0E00
SETRO        00FF SETLRO      0107 SETCRO      010D SETVFU      0115
TOF          0049 TESTRI      00AB TESTLRI     0082 TESTCRI     00B9
VFCH         0009 VFUSTART    0E01 VFUSTOP     0F00 VSTART      0075
```

Appendix E

Z80 Architecture and Instructions



Internal Z80 Organization

Z80 INSTRUCTION CODES

(The literal d is shown as 05 in the object code.)

OBJ CODE	SOURCE STATEMENT	
8E	ADC	A,(HL)
DD8E05	ADC	A,(IX+d)
FD8E05	ADC	A,(IY+d)
8F	ADC	A,A
88	ADC	A,B
89	ADC	A,C
8A	ADC	A,D
88	ADC	A,E
8C	ADC	A,H
8D	ADC	A,L
CE20	ADC	A,n
ED4A	ADC	HL,BC
ED5A	ADC	HL,DE
ED6A	ADC	HL,HL
ED7A	ADC	HL,SP
86	ADD	A,(HL)
DD8605	ADD	A,(IX+d)
FD8605	ADD	A,(IY+d)
87	ADD	A,A
80	ADD	A,B
81	ADD	A,C
82	ADD	A,D
83	ADD	A,E
84	ADD	A,H
85	ADD	A,L
C620	ADD	A,n
09	ADD	HL,BC
19	ADD	HL,DE
29	ADD	HL,HL
39	ADD	HL,SP
DD09	ADD	IX,BC
DD19	ADD	IX,DE
DD29	ADD	IX,IX
DD39	ADD	IX,SP
FD09	ADD	IY,BC
FD19	ADD	IY,DE
FD29	ADD	IY,IY
FD39	ADD	IY,SP
A6	AND	(HL)
DDA605	AND	(IX+d)
FDA605	AND	(IY+d)
A7	AND	A
A0	AND	B
A1	AND	C
A2	AND	D
A3	AND	E
A4	AND	H
A5	AND	L
E620	AND	n
CB46	BIT	0,(HL)
DDCB0546	BIT	0,(IX+d)
FDCB0546	BIT	0,(IY+d)
CB47	BIT	0,A
CB40	BIT	0,B
CB41	BIT	0,C
CB42	BIT	0,D
CB43	BIT	0,E
CB44	BIT	0,H
CB45	BIT	0,L
CB4E	BIT	1,(HL)
DDCB054E	BIT	1,(IX+d)
FDCB054E	BIT	1,(IY+d)
CB4F	BIT	1,A
CB48	BIT	1,B
CB49	BIT	1,C
CB4A	BIT	1,D
CB4B	BIT	1,E
CB4C	BIT	1,H
CB4D	BIT	1,L
CB56	BIT	2,(HL)
DDCB0556	BIT	2,(IX+d)
FDCB0556	BIT	2,(IY+d)
CB57	BIT	2,A
CB50	BIT	2,B
CB51	BIT	2,C
CB52	BIT	2,D
CB53	BIT	2,E
CB54	BIT	2,H
CB55	BIT	2,L
CB5E	BIT	3,(HL)
DDCB055E	BIT	3,(IX+d)
FDCB055E	BIT	3,(IY+d)
CB5F	BIT	3,A
CB58	BIT	3,B
CB59	BIT	3,C
CB5A	BIT	3,D
CB58	BIT	3,E
CB5C	BIT	3,H
CB5D	BIT	3,L
CB66	BIT	4,(HL)
DDCB0566	BIT	4,(IX+d)
FDCB0566	BIT	4,(IY+d)
CB67	BIT	4,A
CB60	BIT	4,B
CB61	BIT	4,C
CB62	BIT	4,D

OBJ CODE	SOURCE STATEMENT	
CB63	BIT	4,E
CB64	BIT	4,H
CB65	BIT	4,L
CB6E	BIT	5,(HL)
DDCB056E	BIT	5,(IX+d)
FDCB056E	BIT	5,(IY+d)
CB6F	BIT	5,A
CB68	BIT	5,B
CB69	BIT	5,C
CB6A	BIT	5,D
CB6B	BIT	5,E
CB6C	BIT	5,H
CB6D	BIT	5,L
CB76	BIT	6,(HL)
DDCB0576	BIT	6,(IX+d)
FDCB0576	BIT	6,(IY+d)
CB77	BIT	6,A
CB70	BIT	6,B
CB71	BIT	6,C
CB72	BIT	6,D
CB73	BIT	6,E
CB74	BIT	6,H
CB75	BIT	6,L
CB7E	BIT	7,(HL)
DDCB057E	BIT	7,(IX+d)
FDCB057E	BIT	7,(IY+d)
CB7F	BIT	7,A
CB78	BIT	7,B
CB79	BIT	7,C
CB7A	BIT	7,D
CB7B	BIT	7,E
CB7C	BIT	7,H
CB7D	BIT	7,L
DC8405	CALL	C,nn
FC8405	CALL	M,nn
D48405	CALL	NC,nn
C48405	CALL	NZ,nn
F48405	CALL	P,nn
EC8405	CALL	PE,nn
E48405	CALL	PO,nn
CC8405	CALL	Z,nn
CD8405	CALL	nn
3F	CCF	
BE	CP	(HL)
DDBE05	CP	(IX+d)
FDBE05	CP	(IY+d)
BF	CP	A
B8	CP	B
B9	CP	C
BA	CP	D
BB	CP	E
BC	CP	H
BD	CP	L
FE20	CP	n
EDA9	CPD	
EDB9	CPDR	

OBJ CODE	SOURCE STATEMENT	
EDB1	CPIR	
EDA1	CPI	
2F	CPL	
27	DAA	
35	DEC	(HL)
DD3505	DEC	(IX+d)
FD3505	DEC	(IY+d)
3D	DEC	A
05	DEC	B
0B	DEC	BC
0D	DEC	C
15	DEC	D
1B	DEC	DE
1D	DEC	E
25	DEC	H
2B	DEC	HL
DD2B	DEC	IX
FD2B	DEC	IY
2D	DEC	L
3B	DEC	SP
F3	DI	
102E	DJNZ	e
FB	EI	
E3	EX	(SP),HL
DDE3	EX	(SP),IX
FDE3	EX	(SP),IY
08	EX	AF,AF'
EB	EX	DE,HL
D9	EXX	
76	HALT	
ED46	IM	0
ED56	IM	1
ED5E	IM	2
ED78	IN	A,(C)
ED40	IN	B,(C)
ED48	IN	C,(C)
ED50	IN	D,(C)
ED58	IN	E,(C)
ED60	IN	H,(C)
ED68	IN	L,(C)
34	INC	(HL)
DD3405	INC	(IX+d)
FD3405	INC	(IY+d)
3C	INC	A
04	INC	B
03	INC	BC
0C	INC	C
14	INC	D
13	INC	DE
1C	INC	E
24	INC	H
23	INC	HL
DD23	INC	IX
FD23	INC	IY
2C	INC	L
33	INC	SP
DB20	IN	A,(n)

OBJ CODE	SOURCE STATEMENT	
EDAA	IND	
EDBA	INDR	
EDA2	INI	
EDB2	INIR	
C38405	JP	nn
E9	JP	(HL)
DDE9	JP	(IX)
FDE9	JP	(IY)
DA8405	JP	C,nn
FA8405	JP	M,nn
D28405	JP	NC,nn
C28405	JP	NZ,nn
F28405	JP	P,nn
EA8405	JP	PE,nn
E28405	JP	PO,nn
CA8405	JP	Z,nn
382E	JR	C,e
302E	JR	NC,e
202E	JR	NZ,e
282E	JR	Z,e
182E	JR	e
02	LD	(BC),A
12	LD	(DE),A
77	LD	(HL),A
70	LD	(HL),B
71	LD	(HL),C
72	LD	(HL),D
73	LD	(HL),E
74	LD	(HL),H
75	LD	(HL),L
3620	LD	(HL),n
DD7705	LD	(IX+d),A
DD7005	LD	(IX+d),B
DD7105	LD	(IX+d),C
DD7205	LD	(IX+d),D
DD7305	LD	(IX+d),E
DD7405	LD	(IX+d),H
DD7505	LD	(IX+d),L
DD360520	LD	(IX+d),n
FD7705	LD	(IY+d),A
FD7005	LD	(IY+d),B
FD7105	LD	(IY+d),C
FD7205	LD	(IY+d),D
FD7305	LD	(IY+d),E
FD7405	LD	(IY+d),H
FD7505	LD	(IY+d),L
FD360520	LD	(IY+d),n
328405	LD	(nn),A
ED438405	LD	(nn),BC
ED538405	LD	(nn),DE
228405	LD	(nn),HL
DD228405	LD	(nn),IX
FD228405	LD	(nn),IY
ED738405	LD	(nn),SP
0A	LD	A,(BC)
1A	LD	A,(DE)
7E	LD	A,(HL)

OBJ CODE	SOURCE STATEMENT	
DD7E05	LD	A,(IX+d)
FD7E05	LD	A,(IY+d)
3A8405	LD	A,(nn)
7F	LD	A,A
78	LD	A,B
79	LD	A,C
7A	LD	A,D
7B	LD	A,E
7C	LD	A,H
ED57	LD	A,I
7D	LD	A,L
3E20	LD	A,n
ED5F	LD	A,R
46	LD	B,(HL)
DD4605	LD	B,(IX+d)
FD4605	LD	B,(IY+d)
47	LD	B,A
40	LD	B,B
41	LD	B,C
42	LD	B,D
43	LD	B,E
44	LD	B,H
45	LD	B,L
0620	LD	B,n
ED4B8405	LD	BC,(nn)
018405	LD	BC,nn
4E	LD	C,(HL)
DD4E05	LD	C,(IX+d)
FD4E05	LD	C,(IY+d)
4F	LD	C,A
48	LD	C,B
49	LD	C,C
4A	LD	C,D
4B	LD	C,E
4C	LD	C,H
4D	LD	C,L
0E20	LD	C,n
56	LD	D,(HL)
DD5605	LD	D,(IX+d)
FD5605	LD	D,(IY+d)
57	LD	D,A
50	LD	D,B
51	LD	D,C
52	LD	D,D
53	LD	D,E
54	LD	D,H
55	LD	D,L
1620	LD	D,n
ED5B8405	LD	DE,(nn)
118405	LD	DE,nn
5E	LD	E,(HL)
DD5E05	LD	E,(IX+d)
FD5E05	LD	E,(IY+d)
5F	LD	E,A
58	LD	E,B
59	LD	E,C
5A	LD	E,D

OBJ CODE	SOURCE STATEMENT	
5B	LD	E,E
5C	LD	E,H
5D	LD	E,L
1E20	LD	E,n
66	LD	H,(HL)
DD6605	LD	H,(IX+d)
FD6605	LD	H,(IY+d)
67	LD	H,A
60	LD	H,B
61	LD	H,C
62	LD	H,D
63	LD	H,E
64	LD	H,H
65	LD	H,L
2620	LD	H,n
2A8405	LD	HL,(nn)
218405	LD	HL,nn
ED47	LD	I,A
DD2A8405	LD	IX,(nn)
DD218405	LD	IX,nn
FD2A8405	LD	IY,(nn)
FD218405	LD	IY,nn
6E	LD	L,(HL)
DD6E05	LD	L,(IX+d)
FD6E05	LD	L,(IY+d)
6F	LD	L,A
68	LD	L,B
69	LD	L,C
6A	LD	L,D
6B	LD	L,E
6C	LD	L,H
6D	LD	L,L
2E20	LD	L,n
ED4F	LD	R,A
ED7B8405	LD	SP,(nn)
F9	LD	SP,HL
DDF9	LD	SP,IX
FDF9	LD	SP,IY
318405	LD	SP,nn
EDA8	LDD	
EDB8	LDDR	
EDA0	LDI	
EDB0	LDIR	
ED44	NEG	
00	NOP	
B6	OR	(HL)
DDB605	OR	(IX+d)
FDB605	OR	(IY+d)
B7	OR	A
B0	OR	B
B1	OR	C
B2	OR	D
B3	OR	E
B4	OR	H
B5	OR	L
F620	OR	n
EDBB	OTDR	

OBJ CODE	SOURCE STATEMENT	
EDB3	OTIR	
ED79	OUT	(C),A
ED41	OUT	(C),B
ED49	OUT	(C),C
ED51	OUT	(C),D
ED59	OUT	(C),E
ED61	OUT	(C),H
ED69	OUT	(C),L
D320	OUT	(n),A
EDAB	OUTD	
EDA3	OUTI	
F1	POP	AF
C1	POP	BC
D1	POP	DE
E1	POP	HL
DDE1	POP	IX
FDE1	POP	IY
F5	PUSH	AF
C5	PUSH	BC
D5	PUSH	DE
E5	PUSH	HL
DDE5	PUSH	IX
FDE5	PUSH	IY
CB86	RES	0,(HL)
DDCB0586	RES	0,(IX+d)
FDCB0586	RES	0,(IY+d)
CB87	RES	0,A
CB80	RES	0,B
CB81	RES	0,C
CB82	RES	0,D
CB83	RES	0,E
CB84	RES	0,H
CB85	RES	0,L
CB8E	RES	1,(HL)
DDCB058E	RES	1,(IX+d)
FDCB058E	RES	1,(IY+d)
CB8F	RES	1,A
CB88	RES	1,B
CB89	RES	1,C
CB8A	RES	1,D
CB8B	RES	1,E
CB8C	RES	1,H
CB8D	RES	1,L
CB96	RES	2,(HL)
DDCB0596	RES	2,(IX+d)
FDCB0596	RES	2,(IY+d)
CB97	RES	2,A
CB90	RES	2,B
CB91	RES	2,C
CB92	RES	2,D
CB93	RES	2,E
CB94	RES	2,H
CB95	RES	2,L
CB9E	RES	3,(HL)
DDCB059E	RES	3,(IX+d)
FDCB059E	RES	3,(IY+d)

OBJ CODE	SOURCE STATEMENT	
CB9F	RES	3,A
CB98	RES	3,B
CB99	RES	3,C
CB9A	RES	3,D
CB9B	RES	3,E
CB9C	RES	3,H
CB9D	RES	3,L
CBA6	RES	4,(HL)
DDCB05A6	RES	4,(IX+d)
FDCB05A6	RES	4,(IY+d)
CBA7	RES	4,A
CBA0	RES	4,B
CBA1	RES	4,C
CBA2	RES	4,D
DBA3	RES	4,E
CBA4	RES	4,H
CBA5	RES	4,L
CBAE	RES	5,(HL)
DDCB05AE	RES	5,(IX+d)
FDCB05AE	RES	5,(IY+d)
CBAF	RES	5,A
CBA8	RES	5,B
CBA9	RES	5,C
CBAA	RES	5,D
CBAB	RES	5,E
CBAC	RES	5,H
CBAD	RES	5,L
CBB6	RES	6,(HL)
DDCB05B6	RES	6,(IX+d)
FDCB05B6	RES	6,(IY+d)
CBB7	RES	6,A
CBB0	RES	6,B
CBB1	RES	6,C
CBB2	RES	6,D
CBB3	RES	6,E
CBB4	RES	6,H
CBB5	RES	6,L
CBBE	RES	7,(HL)
DDCB05BE	RES	7,(IX+d)
FDCB05BE	RES	7,(IY+d)
CBBF	RES	7,A
CBB8	RES	7,B
CBB9	RES	7,C
CBBA	RES	7,D
CBBB	RES	7,E
CBBC	RES	7,H
CBBD	RES	7,L
C9	RET	
D8	RET	C
F8	RET	M
D0	RET	NC
C0	RET	NZ
F0	RET	P
E8	RET	PE
E0	RET	PO
C8	RET	Z

OBJ CODE	SOURCE STATEMENT	
ED4D	RETI	
ED45	RETN	
CB16	RL	(HL)
DDCB0516	RL	(IX+d)
FDCB0516	RL	(IY+d)
CB17	RL	A
CB10	RL	B
CB11	RL	C
CB12	RL	D
CB13	RL	E
CB14	RL	H
CB15	RL	L
17	RLA	
CB06	RLC	(HL)
DDCB0506	RLC	(IX+d)
FDCB0506	RLC	(IY+d)
CB07	RLC	A
CB00	RLC	B
CB01	RLC	C
CB02	RLC	D
CB03	RLC	E
CB04	RLC	H
CB05	RLC	L
07	RLCA	
ED6F	RLD	
CB1E	RR	(HL)
DDCB051E	RR	(IX+d)
FDCB051E	RR	(IY+d)
CB1F	RR	A
CB18	RR	B
CB19	RR	C
CB1A	RR	D
CB1B	RR	E
CB1C	RR	H
CB1D	RR	L
1F	RRA	
CB0E	RRC	(HL)
DDCB050E	RRC	(IX+d)
FDCB050E	RRC	(IY+d)
CB0F	RRC	A
CB08	RRC	B
CB09	RRC	C
CB0A	RRC	D
CB0B	RRC	E
CB0C	RRC	H
CB0D	RRC	L
0F	RRCA	
ED67	RRD	
C7	RST	00H
CF	RST	08H
D7	RST	10H
DF	RST	18H
E7	RST	20H
EF	RST	28H
F7	RST	30H
FF	RST	38H
DE20	SBC	A,n

OBJ CODE	SOURCE STATEMENT	
9E	SBC	A,(HL)
DD9E05	SBC	A,(IX+d)
FD9E05	SBC	A,(IY+d)
9F	SBC	A,A
98	SBC	A,B
99	SBC	A,C
9A	SBC	A,D
9B	SBC	A,E
9C	SBC	A,H
9D	SBC	A,L
ED42	SBC	HL,BC
ED52	SBC	HL,DE
ED62	SBC	HL,HL
ED72	SBC	HL,SP
37	SCF	
CBC6	SET	0,(HL)
DDCB05C6	SET	0,(IX+d)
FDCB05C6	SET	0,(IY+d)
CBC7	SET	0,A
CBC0	SET	0,B
CBC1	SET	0,C
CBC2	SET	0,D
CBC3	SET	0,E
CBC4	SET	0,H
CBC5	SET	0,L
CBCE	SET	1,(HL)
DDCB05CE	SET	1,(IX+d)
FDCB05CE	SET	1,(IY+d)
CBCF	SET	1,A
CBC8	SET	1,B
CBC9	SET	1,C
CBCA	SET	1,D
CBCB	SET	1,E
CBCC	SET	1,H
CBCD	SET	1,L
CBD6	SET	2,(HL)
DDCB05D6	SET	2,(IX+d)
FDCB05D6	SET	2,(IY+d)
CBD7	SET	2,A
CBD0	SET	2,B
CBD1	SET	2,C
CBD2	SET	2,D
CBD3	SET	2,E
CBD4	SET	2,H
CBD5	SET	2,L
CBD8	SET	3,B
CBDE	SET	3,(HL)
DDCB05DE	SET	3,(IX+d)
FDCB05DE	SET	3,(IY+d)
CBDF	SET	3,A
CBD9	SET	3,C
CBDA	SET	3,D
CBDB	SET	3,E
CBDC	SET	3,H
CBDD	SET	3,L
CBE6	SET	4,(HL)

OBJ CODE	SOURCE STATEMENT	
DDCB05E6	SET	4,(IX+d)
FDCB05E6	SET	4,(IY+d)
CBE7	SET	4,A
CBE0	SET	4,B
CBE1	SET	4,C
CBE2	SET	4,D
CBE3	SET	4,E
CBE4	SET	4,H
CBE5	SET	4,L
CBEE	SET	5,(HL)
DDCB05EE	SET	5,(IX+d)
FDCB05EE	SET	5,(IY+d)
CBEF	SET	5,A
CBE8	SET	5,B
CBE9	SET	5,C
CBEA	SET	5,D
CBEB	SET	5,E
CBEC	SET	5,H
CBED	SET	5,L
CBF6	SET	6,(HL)
DDCB05F6	SET	6,(IX+d)
FDCB05F6	SET	6,(IY+d)
CBF7	SET	6,A
CBF0	SET	6,B
CBF1	SET	6,C
CBF2	SET	6,D
CBF3	SET	6,E
CBF4	SET	6,H
CBF5	SET	6,L
CBFE	SET	7,(HL)
DDCB05FE	SET	7,(IX+d)
FDCB05FE	SET	7,(IY+d)
CBFF	SET	7,A
CBF8	SET	7,B
CBF9	SET	7,C
CBFA	SET	7,D
CBFB	SET	7,E
CBFC	SET	7,H
CBFD	SET	7,L
CB26	SLA	(HL)
DDCB0526	SLA	(IX+d)
FDCB0526	SLA	(IY+d)
CB27	SLA	A
CB20	SLA	B
CB21	SLA	C
CB22	SLA	D
CB23	SLA	E
CB24	SLA	H
CB25	SLA	L
CB2E	SRA	(HL)
DDCB052E	SRA	(IX+d)
FDCB052E	SRA	(IY+d)
CB2F	SRA	A
CB28	SRA	B
CB29	SRA	C
CB2A	SRA	D

OBJ CODE	SOURCE STATEMENT	
CB2B	SRA	E
CB2C	SRA	H
CB2D	SRA	L
CB3E	SRL	(HL)
DDCB053E	SRL	(IX+d)
FDCB053E	SRL	(IY+d)
CB3F	SRL	A
CB38	SRL	B
CB39	SRL	C
CB3A	SRL	D
CB3B	SRL	E
CB3C	SRL	H
CB3D	SRL	L
96	SUB	(HL)
DD9605	SUB	(IX+d)
FD9605	SUB	(IY+d)
97	SUB	A
90	SUB	B
91	SUB	C
92	SUB	D
93	SUB	E
94	SUB	H
95	SUB	L
D620	SUB	n
AE	XOR	(HL)
DDAE05	XOR	(IX+d)
FDAE05	XOR	(IY+d)
AF	XOR	A
A8	XOR	B
A9	XOR	C
AA	XOR	D
AB	XOR	E
AC	XOR	H
AD	XOR	L
EE20	XOR	n

Appendix F

Mini Computer Mnemonics

INSTRUCTION MNEMONICS

ALPHABETIC LISTING

ADC	102000	Add the complement of ACS to ACD; use Carry as base for carry bit.
ADCC	102060	Add the complement of ACS to ACD; use complement of Carry as base for carry bit.
ADCCL	102160	Add the complement of ACS to ACD; use complement of Carry as base for carry bit; rotate left.
ADCCR	102260	Add the complement of ACS to ACD; use complement of Carry as base for carry bit; rotate right.
ADCCS	102360	Add the complement of ACS to ACD; use complement of Carry as base for carry bit; swap halves of result.
ADCL	102100	Add the complement of ACS to ACD; use Carry as base for carry bit; rotate left.
ADCO	102040	Add the complement of ACS to ACD; use 1 as base for carry bit.
ADCOL	102140	Add the complement of ACS to ACD; use 1 as base for carry bit; rotate left.
ADCOR	102240	Add the complement of ACS to ACD; use 1 as base for carry bit; rotate right.
ADCOS	102340	Add the complement of ACS to ACD; use 1 as base for carry bit; swap halves of result.
ADCR	102200	Add the complement of ACS to ACD; use Carry as base for carry bit; rotate right.
ADCS	102300	Add the complement of ACS to ACD; use Carry as base for carry bit; swap halves of result.
ADCZ	102020	Add the complement of ACS to ACD; use 0 as base for carry bit.
ADCZL	102120	Add the complement of ACS to ACD; use 0 as base for carry bit; rotate left.
ADCZR	102220	Add the complement of ACS to ACD; use 0 as base for carry bit; rotate right.
ADCZS	102320	Add the complement of ACS to ACD; use 0 as base for carry bit; swap halves of result.
ADD	103000	Add ACS to ACD; use Carry as base for carry bit.
ADDC	103060	Add ACS to ACD; use complement of Carry as base for carry bit.
ADDCL	103160	Add ACS to ACD; use complement of Carry as base for carry bit; rotate left.
ADDCR	103260	Add ACS to ACD; use complement of Carry as base for carry bit; rotate right.
ADDCS	103360	Add ACS to ACD; use complement of Carry as base for carry bit; swap halves of result.
ADDL	103100	Add ACS to ACD; use Carry as base for carry bit; rotate left.
ADDO	103040	Add ACS to ACD; use 1 as base for carry bit.
ADDOL	103140	Add ACS to ACD; use 1 as base for carry bit; rotate left.

ADDOR	103240	Add ACS to ACD; use 1 as base for carry bit; rotate right.
ADDOS	103340	Add ACS to ACD; use 1 as base for carry bit; swap halves of result.
ADDR	103200	Add ACS to ACD; use Carry as base for carry bit; rotate right.
ADDS	103300	Add ACS to ACD; use Carry as base for carry bit; swap halves of result.
ADDZ	103020	Add ACS to ACD; use 0 as base for carry bit.
ADDZL	103120	Add ACS to ACD; use 0 as base for carry bit; rotate left.
ADDZR	103220	Add ACS to ACD; use 0 as base for carry bit; rotate right.
ADDZS	103320	Add ACS to ACD; use 0 as base for carry bit; swap halves of result.
AND	103400	And ACS with ACD; use Carry as carry bit.
ANDC	103460	And ACS with ACD; use complement of Carry as carry bit.
ANDCL	103560	And ACS with ACD; use complement of Carry as carry bit; rotate left.
ANDCR	103660	And ACS with ACD; use complement of Carry as carry bit; rotate right.
ANDCS	103760	And ACS with ACD; use complement of Carry as carry bit; swap halves of result.
ANDL	103500	And ACS with ACD; use Carry as carry bit; rotate left.
ANDO	103440	And ACS with ACD; use 1 as carry bit.
ANDOL	103540	And ACS with ACD; use 1 as carry bit; rotate left.
ANDOR	103640	And ACS with ACD; use 1 as carry bit; rotate right.
ANDOS	103740	And ACS with ACD; use 1 as carry bit; swap halves of result.
ANDR	103600	And ACS with ACD; use Carry as carry bit; rotate right.
ANDS	103700	And ACS with ACD; use Carry as carry bit; swap halves of result.
ANDZ	103420	And ACS with ACD; use 0 as carry bit.
ANDZL	103520	And ACS with ACD; use 0 as carry bit; rotate left.
ANDZR	103620	And ACS with ACD; use 0 as carry bit; rotate right.
ANDZS	103720	And ACS with ACD; use 0 as carry bit; swap halves of result.
COM	100000	Place the complement of ACS in ACD; use Carry as carry bit.
COMC	100060	Place the complement of ACS in ACD; use complement of Carry as carry bit.
COMCL	100160	Place the complement of ACS in ACD; use complement of Carry as carry bit; rotate left.
COMCR	100260	Place the complement of ACS in ACD; use complement of Carry as carry bit; rotate right.
COMCS	100360	Place the complement of ACS in ACD; use complement of Carry as carry bit; swap halves of result.
COML	100100	Place the complement of ACS in ACD; use Carry as carry bit; rotate left.
COMO	100040	Place the complement of ACS in ACD; use 1 as carry bit.
COMOL	100140	Place the complement of ACS in ACD; use 1 as carry bit; rotate left.

COMOR	100240	Place the complement of ACS in ACD; use 1 as carry bit; rotate right.
COMOS	100340	Place the complement of ACS in ACD; use 1 as carry bit; swap halves of result.
COMR	100200	Place the complement of ACS in ACD; use Carry as carry bit; rotate right.
COMS	100300	Place the complement of ACS in ACD; use Carry as carry bit; swap halves of result.
COMZ	100020	Place the complement of ACS in ACD; use 0 as carry bit.
COMZL	100120	Place the complement of ACS in ACD; use 0 as carry bit; rotate left.
COMZR	100220	Place the complement of ACS in ACD; use 0 as carry bit; rotate right.
COMZS	100320	Place the complement of ACS in ACD; use 0 as carry bit; swap halves of result.
DIA	060400	Data in, A buffer to AC.
DIAC	060600	Data in, A buffer to AC; clear device.
DIAP	060700	Data in, A buffer to AC; send special pulse to device.
DIAS	060500	Data in, A buffer to AC; start device.
DIB	061400	Data in, B buffer to AC.
DIBC	061600	Data in, B buffer to AC; clear device.
DIBP	061700	Data in, B buffer to AC; send special pulse to device.
DIBS	061500	Data in, B buffer to AC; start device.
DIC	062400	Data in, C buffer to AC.
DICC	062600	Data in, C buffer to AC; clear device.
DICP	062700	Data in, C buffer to AC; send special pulse to device.
DICS	062500	Data in, C buffer to AC; start device.
DIV	073101	If overflow, set Carry. Otherwise divide AC0-AC1 by AC2. Put quotient in AC1, remainder in AC0.
DOA	061000	Data out, AC to A buffer.
DOAC	061200	Data out, AC to A buffer; clear device.
DOAP	061300	Data out, AC to A buffer; send special pulse to device.
DOAS	061100	Data out, AC to A buffer; start device.
DOB	062000	Data out, AC to B buffer.
DOBC	062200	Data out, AC to B buffer; clear device.
DOBP	062300	Data out, AC to B buffer; send special pulse to device.
DOBS	062100	Data out, AC to B buffer; start device.
DOC	063000	Data out, AC to C buffer.
DOCC	063200	Data out, AC to C buffer; clear device.
DOCP	063300	Data out, AC to C buffer; send special pulse to device
DOCS	063100	Data out, AC to C buffer; start device.
DSZ	014000	Decrement location *E* by 1 and skip if result is zero.

HALT	063077	Halt the processor (= DOC 0,CPU).
INC	101400	Place ACS + 1 in ACD; use Carry as base for carry bit.
INCC	101460	Place ACS + 1 in ACD; use complement of Carry as base for carry bit.
INCCL	101560	Place ACS + 1 in ACD; use complement of Carry as base for carry bit; rotate left.
INCCR	101660	Place ACS + 1 in ACD; use complement of Carry as base for carry bit; rotate right.
INCCS	101760	Place ACS + 1 in ACD; use complement of Carry as base for carry bit; swap halves of result.
INCL	101500	Place ACS + 1 in ACD; use Carry as base for carry bit; rotate left.
INCO	101440	Place ACS + 1 in ACD; use 1 as base for carry bit.
INCOL	101540	Place ACS + 1 in ACD; use 1 as base for carry bit; rotate left.
INCOR	101640	Place ACS + 1 in ACD; use 1 as base for carry bit; rotate right.
INCOS	101740	Place ACS + 1 in ACD; use 1 as base for carry bit; swap halves of result.
INCR	101600	Place ACS + 1 in ACD; use Carry as base for carry bit; rotate right.
INCS	101700	Place ACS + 1 in ACD; use Carry as base for carry bit; swap halves of result.
INCZ	101420	Place ACS + 1 in ACD; use 0 as base for carry bit.
INCZL	101520	Place ACS + 1 in ACD; use 0 as base for carry bit; rotate left.
INCZR	101620	Place ACS + 1 in ACD; use 0 as base for carry bit; rotate right.
INCZS	101720	Place ACS + 1 in ACD; use 0 as base for carry bit; swap halves of result.
INTA	061477	Acknowledge interrupt by loading code of nearest device that is requesting an interrupt into AC bits 10–15 (= DIB –,CPU).
INTDS	060277	Disable interrupt by clearing Interrupt On (= NIOC CPU).
INTEN	060177	Enable interrupt by setting Interrupt On (= NIOS CPU).
IORST	062677	Clear all IO devices, clear Interrupt On, reset clock to line frequency (= DICC 0,CPU).
ISZ	010000	Increment location *E* by 1 and skip if result is zero.
JMP	000000	Jump to location *E* (put *E* in PC).
JSR	004000	Load PC + 1 in AC3 and jump to subroutine at location *E* (put *E* in PC).
LDA	020000	Load contents of location *E* into AC.
MOV	101000	Move ACS to ACD; use Carry as carry bit.
MOVC	101060	Move ACS to ACD; use complement of Carry as carry bit.
MOVCL	101160	Move ACS to ACD; use complement of Carry as carry bit; rotate left.
MOVCR	101260	Move ACS to ACD; use complement of Carry as carry bit; rotate right.
MOVCS	101360	Move ACS to ACD; use complement of Carry as carry bit; swap halves of result.
MOVL	101100	Move ACS to ACD; use Carry as carry bit; rotate left.

MOVO	101040	Move ACS to ACD; use 1 as carry bit.
MOVOL	101140	Move ACS to ACD; use 1 as carry bit; rotate left.
MOVOR	101240	Move ACS to ACD; use 1 as carry bit; rotate right.
MOVOS	101340	Move ACS to ACD; use 1 as carry bit; swap halves of result.
MOVR	101200	Move ACS to ACD; use Carry as carry bit; rotate right.
MOVS	101300	Move ACS to ACD; use Carry as carry bit; swap halves of result.
MOVZ	101020	Move ACS to ACD; use 0 as carry bit.
MOVZL	101120	Move ACS to ACD; use 0 as carry bit; rotate left.
MOVZR	101220	Move ACS to ACD; use 0 as carry bit; rotate right.
MOVZS	101320	Move ACS to ACD; use 0 as carry bit; swap halves of result.
MSKO	062077	Set up Interrupt Disable flags according to mask in AC (= DOB -,CPU).
MUL	073301	Multiply AC1 by AC2, add product to AC0, put result in AC0-AC1.
NEG	100400	Place negative of ACS in ACD; use Carry as base for carry bit.
NEGC	100460	Place negative of ACS in ACD; use complement of Carry as base for carry bit.
NEGCL	100560	Place negative of ACS in ACD; use complement of Carry as base for carry bit; rotate left.
NEGCR	100660	Place negative of ACS in ACD; use complement of Carry as base for carry bit; rotate right.
NEGCS	100760	Place negative of ACS in ACD; use complement of Carry as base for carry bit; swap halves of result.
NEGL	100500	Place negative of ACS in ACD; use Carry as base for carry bit; rotate left.
NEGO	100440	Place negative of ACS in ACD; use 1 as base for carry bit.
NEGOL	100540	Place negative of ACS in ACD; use 1 as base for carry bit; rotate left.
NEGOR	100640	Place negative of ACS in ACD; use 1 as base for carry bit; rotate right.
NEGOS	100740	Place negative of ACS in ACD; use 1 as base for carry bit; swap halves of result.
NEGR	100600	Place negative of ACS in ACD; use Carry as carry bit; rotate right.
NEGS	100700	Place negative of ACS in ACD; use Carry as carry bit; swap halves of result.
NEGZ	100420	Place negative of ACS in ACD; use 0 as base for carry bit.
NEGZL	100520	Place negative of ACS in ACD; use 0 as base for carry bit; rotate left.
NEGZR	100620	Place negative of ACS in ACD; use 0 as base for carry bit; rotate right.
NEGZS	100720	Place negative of ACS in ACD; use 0 as base for carry bit; swap halves of result.
NIO	060000	No operation.
NIOC	060200	Clear device.
NIOP	060300	Send special pulse to device.

NIOS	060100	Start device.
READS	060477	Read console data switches into AC (= DIA –,CPU).
SBN	000007	Skip if both carry and result are nonzero (skip function in an arithmetic or logical instruction).
SEZ	000006	Skip if either carry or result is zero (skip function in an arithmetic or logical instruction).
SKP	000001	Skip (skip function in an arithmetic or logical instruction).
SKPBN	063400	Skip if Busy is 1.
SKPBZ	063500	Skip if Busy is 0.
SKPDN	063600	Skip if Done is 1.
SKPDZ	063700	Skip if Done is 0.
SNC	000003	Skip if carry bit is 1 (skip function in an arithmetic or logical instruction).
SNR	000005	Skip if result is nonzero (skip function in an arithmetic or logical instruction).
STA	040000	Store AC in location *E*.
SUB	102400	Subtract ACS from ACD; use Carry as base for carry bit.
SUBC	102460	Subtract ACS from ACD; use complement of Carry as base for carry bit.
SUBCL	102560	Subtract ACS from ACD; use complement of Carry as base for carry bit; rotate left.
SUBCR	102660	Subtract ACS from ADC; use complement of Carry as base for carry bit; rotate right.
SUBCS	102760	Subtract ACS from ACD; use complement of Carry as base for carry bit; swap halves of result.
SUBL	102500	Subtract ACS from ACD; use Carry as base for carry bit; rotate left.
SUBO	102440	Subtract ACS from ACD; use 1 as base for carry bit.
SUBOL	102540	Subtract ACS from ACD; use 1 as base for carry bit; rotate left.
SUBOR	102640	Subtract ACS from ACD; use 1 as base for carry bit; rotate right.
SUBOS	102740	Subtract ACS from ACD; use 1 as base for carry bit; swap halves of result.
SUBR	102600	Subtract ACS from ACD; use Carry as base for carry bit; rotate right.
SUBS	102700	Subtract ACS from ACD; use Carry as base for carry bit; swap halves of result.
SUBZ	102420	Subtract ACS from ACS; use 0 as base for carry bit.
SUBZL	102520	Subtract ACS from ACD; use 0 as base for carry bit; rotate left.
SUBZR	102620	Subtract ACS from ACD; use 0 as base for carry bit; rotate right.
SUBZS	102720	Subtract ACS from ACD; use 0 as base for carry bit; swap halves of result.
SZC	000002	Skip if carry is 0 (skip function in an arithmetic or logical instruction).
SZR	000004	Skip if result is zero (skip function in an arithmetic or logical instruction).

@	002000	When this character appears in an instruction, the assembler places a 1 in bit 5 to produce indirect addressing.
@	100000	When this character appears with a 15-bit address, the assembler places a 1 in bit 0, making the address indirect.
#	000010	Appending this character to the mnemonic for an arithmetic or logical instruction places a 1 in bit 12 to prevent the processor from loading the 17-bit result in Carry and ACD. Thus the result of an instruction can be tested for a skip without affecting Carry or the accumulators.

Appendix G

EPROM Programer

This section was included only for intrest sake

```
                 ORG   6000H
                 LOAD  6000H
    START:       EQU   6000H
    CR:          EQU   13
    FF:          EQU   12
    BS:          EQU   8
    BLANK:       EQU   32

    ;EXTERNAL VARIABLE SPACE

    COUNT:       EQU   START-14          ;COUNTER
    ADDR:        EQU   START-12          ;ADD STORE
    DAT:         EQU   START-10          ;DATA STORE
    STOPE:       EQU   START-8           ;EPROM STOP
    STOPR:       EQU   START-6           ;RAM STOP
    STARTE:      EQU   START-4           ;EPROM START
    STARTR:      EQU   START-2           ;RAM START
    ;
    ;
    ;
    HBUFF:       EQU   10F0H
    FILESIZE:    EQU   HBUFF+18
    FILESTART:   EQU   HBUFF+20
    FILEXEC:     EQU   HBUFF+22
    TBUFF:       EQU   11A3H
    EPROMA:      EQU   57356             ;EPROM PORT A
    EPROMB:      EQU   57357             ;EPROM PORT B
    EPROMC:      EQU   57358             ;EPROM PORT C
    EPROMP:      EQU   57359             ;EPROM PROGRAM REG
    TIME:        EQU   58393             ;50 MILISEC

    ;CALLS TO SP-1002 MONITOR

    ASC:         EQU   03DAH             ;LOW NIBBLE A TO ASCI
    PRTHL:       EQU   03BAH             ;PRINT HL IN HEX
    PRTHX:       EQU   03C3H             ;PRINT A IN HEX
    PRNTS:       EQU   000CH             ;PRINT A SPACE
    NL:          EQU   0009H             ;CR+LF
    MSG:         EQU   0015H             ;MESSAGE FROM (DE)
    PHEAD:       EQU   21H               ;OUTPUT TAPEHEADER
    PDATA:       EQU   24H               ;OUTPUT TAPEDATA
    LHEAD:       EQU   27H               ;READ TAPE HEADER
    LDATA:       EQU   2AH               ;READ TAPE DATA
    SCREEN:      EQU   0FB1H             ;HL TO CURSOR POSI
    BRKEY:       EQU   001EH             ;SHIFT/BREAK PRESSE
    PRNT:        EQU   0012H             ;PRINT ASCII
    CURSOR:      EQU   1171H
    HEX:         EQU   03F9H             ;ASCII TO NIBBLE
    CONTROL:     EQU   0DDCH             ;OUTPUT CONTROLL
    GETCHAR:     EQU   09B3H             ;SCAN KEYBOARD
    NORMAL:      EQU   0BCEH             ;CONVERT TO ASCII
    SPECIAL:     EQU   0BB9H             ;CONV TO SCRN CHA
    BELL:        EQU   003EH             ;SOUND BELL
    ;
    ;
    ;

    ;START OF MAIN PGM
    ;
    ;
```

```
                 ;
6000 CD0900      MAIN:       CALL NL               ;***EPROM PROGRAMER***
6003 111362                  LD   DE,M6
6006 CD1500                  CALL MSG
6009 CD0900                  CALL NL
600C CD1E00      BREAK:      CALL BRKEY            ;SHIFT/BREAK ?
600F CA0012                  JP   Z,1200H          ;GOTO ZEN
6012 CDB309                  CALL GETCHAR
6015 FE10                    CP   16               ;P
6017 2827                    JR   Z,PROGRAM        ;PROGRAM EPROM
6019 FE03                    CP   3                ;C
601B CAD560                  JP   Z,CHECK          ;CHECK EPROM CLEAN
601E FE16                    CP   22               ;V
6020 CAFA60                  JP   Z,VERIFY         ;CHECK PROGRAMED OK
6023 FE04                    CP   4                ;D
6025 CA0C61                  JP   Z,DUPLICATE      ;COPY EPROM'S
6028 FE12                    CP   18               ;R
602A CA6561                  JP   Z,READ           ;READ EPROM TO SCREEN
602D FE14                    CP   20               ;T
602F CA3460                  JP   Z,PRINT          ;SCREEN OR PRINTER
6032 18D8                    JR   BREAK
                 ;
                 ;MUST READ BEFORE PRINT
6034 CD0900      PRINT:      CALL NL               ;CR+LF
6037 11CE62                  LD   DE,M19
603A CD1500                  CALL MSG
603D C3D165                  JP   PRONT
                 ;
6040 CD0900      PROGRAM:    CALL NL
6043 112962                  LD   DE,M7
6046 CD1500                  CALL MSG
6049 CD9361                  CALL STAEP
604C CD1763                  CALL GETNUM
604F 22FC5F                  LD   (STARTE),HL      ;SAVE EPROM START
6052 CDBF61                  CALL STARA
6055 CD1763                  CALL GETNUM
6058 22FE5F                  LD   (STARTR),HL      ;SAVE RAM START
605B CDD561                  CALL STORA
605E CD1763                  CALL GETNUM
6061 23                      INC  HL               ;STOP+1
6062 22FA5F                  LD   (STOPR),HL       ;SAVE RAM STOP
6065 CDE863                  CALL INIT             ;SET UP ALL OUTPUT
6068 CD0900                  CALL NL
606B 11D562                  LD   DE,M20
606E CD1500                  CALL MSG
6071 CD0900                  CALL NL               ;LINE FEED
6074 CDB309                  CALL GETCHAR
6077 FE01                    CP   1                ;A
6079 CA8D60                  JP   Z,PG16
607C FE02                    CP   2                ;B
607E CAB160                  JP   Z,PG32
6081 117F62                  LD   DE,M14           ;ERROR
6084 CD1500                  CALL MSG
6087 CD0900                  CALL NL
608A C30060                  JP   MAIN
                 ;
                 ;
                 ;
608D CD0900      PG16:       CALL NL
6090 11F562                  LD   DE,M21
```

```
6093 CD1500                   CALL MSG
6096 116762                   LD   DE,M17
6099 CD1500                   CALL MSG
609C CDB309                   CALL GETCHAR
609F CD0900                   CALL NL
60A2 115B62                   LD   DE,M13
60A5 CD1500                   CALL MSG
60A8 CDB309                   CALL GETCHAR            ;WAIT FOR ANY KEY
60AB CD4764                   CALL PGMR
60AE C30060                   JP   MAIN
                 ;
                 ;
60B1 CD0900      PG32:        CALL NL
60B4 110663                   LD   DE,M22
60B7 CD1500                   CALL MSG
60BA 116762                   LD   DE,M17
60BD CD1500                   CALL MSG
60C0 CDB309                   CALL GETCHAR
60C3 CD0900                   CALL NL
60C6 115B62                   LD   DE,M13
60C9 CD1500                   CALL MSG
60CC CDB309                   CALL GETCHAR
60CF CD7C64                   CALL PGM32
60D2 C30060                   JP   MAIN
                 ;
                 ;
                 ;
                 ;
60D5 CD0900      CHECK:       CALL NL
60D8 113162                   LD   DE,M8
60DB CD1500                   CALL MSG
60DE CD9361                   CALL STAEP
60E1 CD1763                   CALL GETNUM
60E4 22FC5F                   LD   (STARTE),HL        ;SAVE EPROM START
60E7 CDA961                   CALL STOEP
60EA CD1763                   CALL GETNUM
60ED 23                       INC  HL                 ;STOP+1
60EE 22F85F                   LD   (STOPE),HL         ;SAVE EPROM STOP
60F1 CD0C64                   CALL INIT2              ;SET UP FOR READ
60F4 CD8164                   CALL CHK                ;DO CHECK FOR CLEAN ROM
60F7 C30060                   JP   MAIN
                 ;
                 ;
                 ;MUST HAVE PROGRAMED
60FA CD0900      VERIFY:      CALL NL
60FD 113762                   LD   DE,M9
6100 CD1500                   CALL MSG
6103 CD0C64                   CALL INIT2
6106 CD8865                   CALL VERF
6109 C30060                   JP   MAIN
                 ;
                 ;
610C CD0900      DUPLICATE:   CALL NL
610F 113E62                   LD   DE,M10
6112 CD1500                   CALL MSG
6115 CD9361                   CALL STAEP
6118 CD1763                   CALL GETNUM
611B 22FC5F                   LD   (STARTE),HL        ;EPROM START
611E CDA961                   CALL STOEP
6121 CD1763                   CALL GETNUM
```

```
6124 23                      INC  HL
6125 22F85F                  LD   (STOPE),HL        ;EPROM STOP
6128 CD8F61                  CALL STARA
612B CD1763                  CALL GETNUM
612E 22FE5F                  LD   (STARTR),HL       ;RAM START
6131 D5                      PUSH DE
6132 E5                      PUSH HL
6133 37                      SCF
6134 3F                      CCF
6135 2AF85F                  LD   HL,(STOPE)
6138 ED5BFC5F                LD   DE,(STARTE)
613C ED52                    SBC  HL,DE
613E ED5BFE5F                LD   DE,(STARTR)
6142 ED5A                    ADC  HL,DE
6144 22FA5F                  LD   (STOPR),HL
6147 E1                      POP  HL
6148 D1                      POP  DE
6149 CD0C64                  CALL INIT2
614C CD5965                  CALL REDO              ;READ INTO RAM
614F CDE863                  CALL INIT
6152 D5                      PUSH DE
6153 CD0900                  CALL NL
6156 115B62                  LD   DE,M13
6159 CD1500                  CALL MSG
615C CDB309                  CALL GETCHAR           ;SCAN KEYBOARD
615F CD4764                  CALL PGMR
6162 C30060                  JP   MAIN
                 ;
                 ;
6165 CD0900      READ:       CALL NL
6168 114862                  LD   DE,M11
616B CD1500                  CALL MSG
616E CD9361                  CALL STAEP             ;EPROM START
6171 CD1763                  CALL GETNUM
6174 22FC5F                  LD   (STARTE),HL
6177 CDA961                  CALL STOEP             ;EPROM STOP
617A CD1763                  CALL GETNUM
617D 23                      INC  HL
617E 22F85F                  LD   (STOPE),HL
6181 CD8F61                  CALL STARA             ;RAM START
6184 CD1763                  CALL GETNUM
6187 22FE5F                  LD   (STARTR),HL
618A CD0C64                  CALL INIT2
618D CD0765                  CALL READE
6190 C30060                  JP   MAIN
                 ;
                 ;
                 ;
6193 CD0900      STAEP:      CALL NL                ;STA ADD EPROM
6196 11EB61                  LD   DE,M1
6199 CD1500                  CALL MSG
619C 11F861                  LD   DE,M3
619F CD1500                  CALL MSG
61A2 110162                  LD   DE,M4
61A5 CD1500                  CALL MSG
61A8 C9                      RET
61A9 CD0900      STOEP:      CALL NL                ;STOP ADD EPROM
61AC 11F261                  LD   DE,M2
61AF CD1500                  CALL MSG
61B2 11F861                  LD   DE,M3
```

```
6185 CD1500                   CALL MSG
6188 110162                   LD   DE,M4
618B CD1500                   CALL MSG
618E C9                       RET
618F CD0900   STARA:          CALL NL                 ;STA ADD RAM
61C2 11EB61                   LD   DE,M1
61C5 CD1500                   CALL MSG
61C8 11F861                   LD   DE,M3
61CB CD1500                   CALL MSG
61CE 110862                   LD   DE,M5
61D1 CD1500                   CALL MSG
61D4 C9                       RET
61D5 CD0900   STORA:          CALL NL                 ;STOP ADD RAM
61D8 11F261                   LD   DE,M2
61DB CD1500                   CALL MSG
61DE 11F861                   LD   DE,M3
61E1 CD1500                   CALL MSG
61E4 110862                   LD   DE,M5
61E7 CD1500                   CALL MSG
61EA C9                       RET
61EB 53544152 M1:             DB   'START ',CR
61EF 54200D
61F2 53544F50 M2:             DB   'STOP ',CR
61F6 200D
61F8 41444452 M3:             DB   'ADDRESS ',CR
61FC 45535320
6200 0D
6201 28455052 M4:             DB   '(EPROM) :',CR
6205 4F4D2920
6209 3A0D
620B 2852414D M5:             DB   '(RAM) :',CR
620F 29203A0D
6213 2A2A2A45 M6:             DB   '***EPROM '
6217 50524F4D
621B 20
621C 50524F47                 DB   'PROGRAMER***',CR
6220 52414D45
6224 522A2A2A
6228 0D
6229 50524F47 M7:             DB   'PROGRAM',CR
622D 52414D0D
6231 43484543 M8:             DB   'CHECK',CR
6235 4B0D
6237 56455249 M9:             DB   'VERIFY',CR
623B 46590D
623E 4455504C M10:            DB   'DUPLICATE',CR
6242 49434154
6246 450D
6248 52454144 M11:            DB   'READ',CR
624C 0D
624D 4552524F M12:            DB   'ERROR,NOT HEX',CR
6251 522C4E4F
6255 54204845
6259 580D
625B 434F4E4E M13:            DB   'CONNECT 25 V'
625F 45435420
6263 32352056
6267 20505245 M17:            DB   ' PRESS ANY KEY '
626B 53532041
626F 4E59204B
```

```
6273 455920
6276 544F2053                DB    'TO START',CR
627A 54415254
627E 0D
627F 4552524F M14:           DB    'ERROR ',CR
6283 52200D
6286 56494445 M15:           DB    'VIDEO (V) OR '
628A 4F202856
628E 29204F52
6292 20
6293 5052494E                DB    'PRINTER (E)',CR
6297 54455220
629B 2845290D
629F 52454144 M16:           DB    'READ FROM ROM0(0)
62A3 2046524F
62A7 4D20524F
62AB 4D302830
62AF 29
62B0 204F5220                DB    ' OR ROM1(1)',CR
62B4 524F4D31
62B8 2831290D
62BC 53574954 M18:           DB    'SWITCH PRINTER'
62C0 43482050
62C4 52494E54
62C8 4552
62CA 204F4E0D                DB    ' ON',CR
62CE 5052494E M19:           DB    'PRINT ',CR
62D2 54200D
62D5 50524F47 M20:           DB    'PROGAM A '
62D9 414D2041
62DD 20
62DE 32373136                DB    '2716 (A) OR A '
62E2 20284129
62E6 204F5220
62EA 4120
62EC 32373332                DB    '2732 (B)',CR
62F0 20284229
62F4 0D
62F5 53455420 M21:           DB    'SET SWITCH TO '
62F9 53574954
62FD 43482054
6301 4F20
6303 31360D                  DB    '16',CR
6306 53455420 M22:           DB    'SET SWITCH TO '
630A 53574954
630E 43482054
6312 4F20
6314 33320D                  DB    '32',CR
              ;
              ;
              ;
6317 210000   GETNUM:        LD    HL,0000H         ;CLEAR HL TO 0
631A CD1E00                  CALL  BRKEY            ;SHIFT/BREAK ?
631D CA0060                  JP    Z,MAIN
6320 CDB309                  CALL  GETCHAR          ;GET MSN
6323 CDCE0B                  CALL  NORMAL           ;CONVERT TO ASCII
6326 47                      LD    B,A
6327 CD1200                  CALL  PRNT
632A 78                      LD    A,B
632B CDF903                  CALL  HEX              ;CONVERT TO HEX
```

```
632E DC9E63             CALL C,ERROR
6331 E60F               AND  0FH            ;MASK
6333 CB07               RLC  A
6335 CB07               RLC  A
6337 CB07               RLC  A
6339 CB07               RLC  A
633B 67                 LD   H,A
633C CD1E00             CALL BRKEY          ;SHIFT/BREAK ?
633F CA0060             JP   Z,MAIN
6342 CDB309             CALL GETCHAR        ;GET NMSN
6345 CDCE0B             CALL NORMAL
6348 47                 LD   B,A
6349 CD1200             CALL PRNT
634C 78                 LD   A,B
634D CDF903             CALL HEX
6350 DC9E63             CALL C,ERROR
6353 E60F               AND  0FH
6355 B4                 OR   H
6356 67                 LD   H,A
6357 CD1E00             CALL BRKEY          ;SHIFT/BREAK ?
635A CA0060             JP   Z,MAIN
635D CDB309             CALL GETCHAR        ;SLSN
6360 CDCE0B             CALL NORMAL
6363 47                 LD   B,A
6364 CD1200             CALL PRNT
6367 78                 LD   A,B
6368 CDF903             CALL HEX
636B DC9E63             CALL C,ERROR
636E E60F               AND  0FH
6370 CB07               RLC  A
6372 CB07               RLC  A
6374 CB07               RLC  A
6376 CB07               RLC  A
6378 B5                 OR   L
6379 6F                 LD   L,A
637A CD1E00             CALL BRKEY          ;SHIFT/BREAK ?
637D CA0060             JP   Z,MAIN
6380 CDB309             CALL GETCHAR
6383 CDCE0B             CALL NORMAL
6386 47                 LD   B,A
6387 CD1200             CALL PRNT
638A 78                 LD   A,B
638B CDF903             CALL HEX
638E DC9E63             CALL C,ERROR
6391 E60F               AND  0FH
6393 B5                 OR   L
6394 6F                 LD   L,A
6395 CDB309   LOOPCR:   CALL GETCHAR        ;SCAN KEYBOARD
6398 FECD               CP   0CDH           ;CARRIDGE RETURN
639A C29563             JP   NZ,LOOPCR
639D C9                 RET
              ;
              ;
              ;ERROR ROUTINE
              ;
              ;
639E CD0900   ERROR:    CALL NL             ;CR+LF
63A1 114D62             LD   DE,M12
63A4 CD1500             CALL MSG
63A7 C9                 RET
```

```
                     ;
                     ;
                     ;8255 ROUTINES
                     ;
63A8 E5              V25ON:          PUSH HL
63A9 210EE0                          LD   HL,EPROMC
63AC CBEE                            SET  5,(HL)              ;25 VOLT ON
63AE E1                              POP  HL
63AF C9                              RET
                     ;
                     ;
63B0 E5              V25OFF:         PUSH HL
63B1 210EE0                          LD   HL,EPROMC
63B4 CBAE                            RES  5,(HL)              ;25 VOLT OFF
63B6 E1                              POP  HL
63B7 C9                              RET
                     ;
                     ;
63B8 E5              CE1OFF:         PUSH HL
63B9 210EE0                          LD   HL,EPROMC
63BC CBFE                            SET  7,(HL)              ;CE1 OFF (HI)
63BE E1                              POP  HL
63BF C9                              RET
                     ;
                     ;
63C0 E5              CE1ON:          PUSH HL                  ;CE1 ON (LO)
63C1 210EE0                          LD   HL,EPROMC
63C4 CBBE                            RES  7,(HL)
63C6 E1                              POP  HL
63C7 C9                              RET
                     ;
                     ;
63C8 E5              CE0OFF:         PUSH HL                  ;CE0 OFF (HI)
63C9 210EE0                          LD   HL,EPROMC
63CC CBF6                            SET  6,(HL)
63CE E1                              POP  HL
63CF C9                              RET
                     ;
                     ;
63D0 E5              CE0ON:          PUSH HL                  ;CE0 ON (LO)
63D1 210EE0                          LD   HL,EPROMC
63D4 CBB6                            RES  6,(HL)
63D6 E1                              POP  HL
63D7 C9                              RET
                     ;
                     ;
63D8 E5              OEOFF:          PUSH HL                  ;OE OFF (HI)
63D9 210EE0                          LD   HL,EPROMC
63DC CBE6                            SET  4,(HL)
63DE E1                              POP  HL
63DF C9                              RET
                     ;
                     ;
63E0 E5              OEON:           PUSH HL                  ;OE ON (LO)
63E1 210EE0                          LD   HL,EPROMC
63E4 CBA6                            RES  4,(HL)
63E6 E1                              POP  HL
63E7 C9                              RET
                     ;
                     ;
```

```
                        ;INITIALIZE 8255
63E8 E5         INIT:           PUSH HL
63E9 F5                         PUSH AF
63EA 210FE0                     LD   HL,EPROMP
63ED 3E80                       LD   A,80H
63EF 77                         LD   (HL),A             ;ALL IO TO OUTPUT
63F0 2B                         DEC  HL
63F1 CBF6                       SET  6,(HL)             ;CE0 OFF (HI)
63F3 CBFE                       SET  7,(HL)             ;CE1 OFF (HI)
63F5 CBE6                       SET  4,(HL)             ;OE OFF (HI)
63F7 CBAE                       RES  5,(HL)             ;25 VOLT OFF (LO)
63F9 CB86                       RES  0,(HL)
63FB CB8E                       RES  1,(HL)
63FD CB96                       RES  2,(HL)
63FF CB9E                       RES  3,(HL)
6401 3E00                       LD   A,00
6403 320DE0                     LD   (EPROMB),A
6406 320CE0                     LD   (EPROMA),A
6409 F1                         POP  AF
640A E1                         POP  HL
640B C9                         RET

                        ;
                        ;
640C E5         INIT2:          PUSH HL
640D F5                         PUSH AF
640E 210FE0                     LD   HL,EPROMP
6411 3E90                       LD   A,90H
6413 77                         LD   (HL),A             ;PORT A TO INPUT
6414 2B                         DEC  HL
6415 CBF6                       SET  6,(HL)             ;CE0 OFF (HI)
6417 CBFE                       SET  7,(HL)             ;CE1 OFF (HI)
6419 CBE6                       SET  4,(HL)             ;OE OFF (HI)
641B CBAE                       RES  5,(HL)             ;25 VOLT OFF
641D CB86                       RES  0,(HL)
641F CB8E                       RES  1,(HL)
6421 CB96                       RES  2,(HL)
6423 CB9E                       RES  3,(HL)
6425 3E00                       LD   A,00
6427 320DE0                     LD   (EPROMB),A
642A F1                         POP  AF
642B E1                         POP  HL
642C C9                         RET
                        ;
                        ;
                        ;
                        ;
                        ;COMPARE DE TO BC
                        ;Z=0 IF EQUAL
                        ;
642D D5         COMPARE:        PUSH DE
642E C5                         PUSH BC
642F 7A                         LD   A,D
6430 B8                         CP   B                  ;COMPARE D+B
6431 C23664                     JP   NZ,HOME
6434 7B                         LD   A,E
6435 B9                         CP   C                  ;COMPARE E+C
6436 C1         HOME:           POP  BC
6437 D1                         POP  DE
6438 C9                         RET
```

```
                  ;
                  ;
                  ;WAIT LOOP
                  ;
6439 E5           WAIT:       PUSH HL
643A 2119E4                   LD   HL,TIME
643D 2C           LOOP:       INC  L
643E C23D64                   JP   NZ,LOOP
6441 24                       INC  H
6442 C23D64                   JP   NZ,LOOP
6445 E1                       POP  HL
6446 C9                       RET
                  ;
                  ;PROGRAM LOOP
                  ;
                  ;
6447 ED5BFE5F PGMR:           LD   DE,(STARTR)
644B ED4BFA5F                 LD   BC,(STOPR)
644F 2AFC5F                   LD   HL,(STARTE)
6452 CD5E64       LOOP2:      CALL PGM
6455 23                       INC  HL
6456 13                       INC  DE
6457 CD2D64                   CALL COMPARE
645A C25264                   JP   NZ,LOOP2
645D C9                       RET                        ;RETURN TO MAIN PGM
                  ;
                  ;
                  ;PROGRAM A LOCATION
                  ;
645E 7D           PGM:        LD   A,L
645F 320DE0                   LD   (EPROMB),A            ;ADDRESS LO
6462 7C                       LD   A,H
6463 CBE7                     SET  4,A
6465 CBBF                     RES  7,A
6467 CBEF                     SET  5,A                   ;25V ON
6469 CBF7                     SET  6,A                   ;CE0 HI
646B 320EE0                   LD   (EPROMC),A
646E 1A                       LD   A,(DE)
646F 320CE0                   LD   (EPROMA),A            ;DATA
6472 CDB863                   CALL CE1OFF                ;CE1=HI
6475 CD3964                   CALL WAIT                  ;WAIT 50MILISEC
6478 CDC063                   CALL CE1ON                 ;CE1=LO
647B C9                       RET
                  ;
                  ;
647C ED5BFE5F PGM32:          LD   DE,(STARTR)
6480 ED4BFA5F                 LD   BC,(STOPR)
6484 2AFC5F                   LD   HL,(STARTE)
6487 CD9364       PLOOP2:     CALL P32
648A 23                       INC  HL
648B 13                       INC  DE
648C CD2D64                   CALL COMPARE
648F C28764                   JP   NZ,PLOOP2
6492 C9                       RET
                  ;
                  ;
                  ;PROGRAM A LOCATION
                  ;
6493 7D           P32:        LD   A,L
6494 320DE0                   LD   (EPROMB),A            ;ADDRESS LO
```

```
6497 7C                      LD    A,H
6498 CBE7                    SET   4,A
649A CBFF                    SET   7,A               ;CE1 HI
649C CBEF                    SET   5,A               ;25V ON
649E CBF7                    SET   6,A               ;CE0 HI
64A0 320EE0                  LD    (EPROMC),A
64A3 1A                      LD    A,(DE)            ;DATA
64A4 320CE0                  LD    (EPROMA),A
64A7 CDC063                  CALL  CE1ON             ;CE1 LO
64AA CD3964                  CALL  WAIT              ;WAIT 50 MILLISEC
64AD CDB863                  CALL  CE1OFF            ;CE1 HI
64B0 C9                      RET
              ;
              ;
              ;
              ;CHECK FOR CLEAN ROM
              ;
64B1 ED5BFC5F CHK:           LD    DE,(STARTE)
64B5 ED4BF85F                LD    BC,(STOPE)
64B9 CDC664   LOOP3:         CALL  CK                ;DOTHE CHECK
64BC 13                      INC   DE
64BD CD2D64                  CALL  COMPARE           ;CHECK IF OVER
64C0 C2B964                  JP    NZ,LOOP3
64C3 C30060                  JP    MAIN
              ;
              ;
64C6 7B       CK:            LD    A,E
64C7 320DE0                  LD    (EPROMB),A        ;ADDRESS LO
64CA 7A                      LD    A,D
64CB CBA7                    RES   4,A               ;OE=LO
64CD CBFF                    SET   7,A               ;CE1=HI
64CF CBAF                    RES   5,A               ;25V OFF
64D1 CBF7                    SET   6,A               ;CE0=HI
64D3 320EE0                  LD    (EPROMC),A
64D6 CDC063                  CALL  CE1ON
64D9 3A0CE0                  LD    A,(EPROMA)        ;GET DATA
64DC CDB863                  CALL  CE1OFF
64DF FEFF                    CP    0FFH
64E1 C4E564                  CALL  NZ,ERROR2
64E4 C9                      RET
              ;
              ;
64E5 ED53F45F ERROR2:        LD    (ADDR),DE
64E9 32F65F                  LD    (DAT),A
64EC CD0900                  CALL  NL
64EF D5                      PUSH  DE
64F0 117F62                  LD    DE,M14
64F3 CD1500                  CALL  MSG
64F6 D1                      POP   DE
64F7 2AF45F                  LD    HL,(ADDR)         ;GET ADDRESS
64FA CDBA03                  CALL  PRTHL
64FD CD0C00                  CALL  PRNTS
6500 3AF65F                  LD    A,(DAT)           ;GET DATA
6503 CDC303                  CALL  PRTHX
6506 C9                      RET
              ;
              ;
              ;
6507 CD0900   READE:         CALL  NL                ;CR+LF
650A 119F62                  LD    DE,M16
```

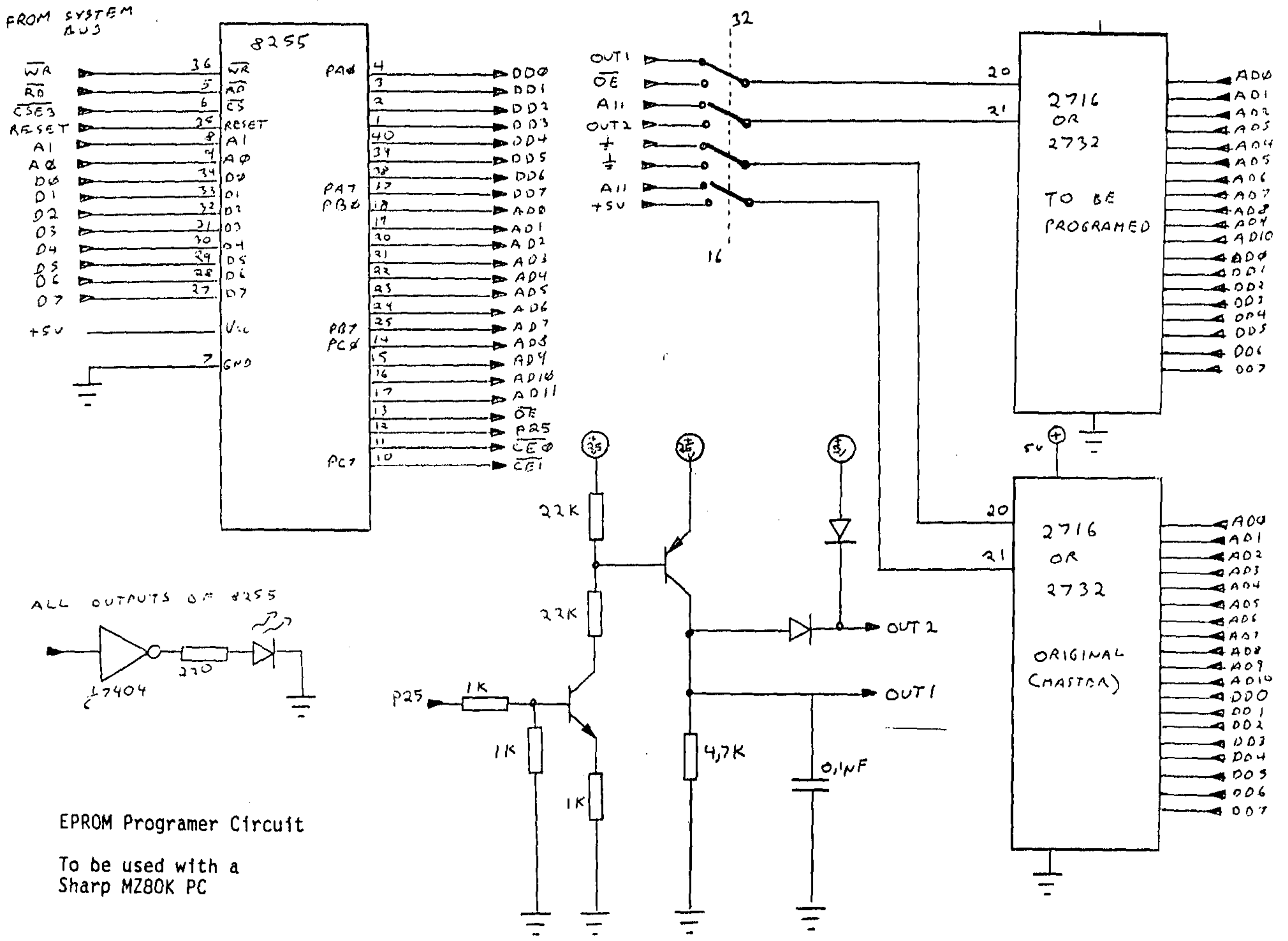
```
650D CD1500                CALL MSG
6510 CDB309                CALL GETCHAR
6513 FE20                  CP   32            ;0
6515 CA5965                JP   Z,RED0        ;ROM0
6518 FE21                  CP   33            ;1
651A CA2765                JP   Z,RED1        ;ROM1
651D 117F62                LD   DE,M14        ;ERROR
6520 CD1500                CALL MSG
6523 CD3E00                CALL BELL
6526 C9                    RET
               ;
               ;
6527 ED5BFC5F RED1:        LD   DE,(STARTE)   ;GET DATA
652B ED4BF85F              LD   BC,(STOPE)
652F 2AFE5F                LD   HL,(STARTR)
6532 CD3E65   LOOPR:       CALL REC1
6535 23                    INC  HL
6536 13                    INC  DE
6537 CD2D64                CALL COMPARE
653A C23265                JP   NZ,LOOPR
653D C9                    RET
               ;
653E 7B       REC1:        LD   A,E
653F 320DE0                LD   (EPROMB),A
6542 7A                    LD   A,D
6543 CBA7                  RES  4,A
6545 CBFF                  SET  7,A
6547 CBAF                  RES  5,A
6549 CBF7                  SET  6,A
654B 320EE0                LD   (EPROMC),A
654E CDC063                CALL CE1ON
6551 3A0CE0                LD   A,(EPROMA)
6554 CDB863                CALL CE1OFF
6557 77                    LD   (HL),A
6558 C9                    RET
               ;
               ;
               ;
               ;
6559 ED5BFC5F RED0:        LD   DE,(STARTE)   ;GET DATA
655D ED4BF85F              LD   BC,(STOPE)
6561 2AFE5F                LD   HL,(STARTR)
6564 CD7065   LOOPQ:       CALL REC0          ;READ ROM
6567 23                    INC  HL
6568 13                    INC  DE
6569 CD2D64                CALL COMPARE
656C C26465                JP   NZ,LOOPQ
656F C9                    RET
               ;
               ;
6570 7B       REC0:        LD   A,E
6571 320DE0                LD   (EPROMB),A
6574 7A                    LD   A,D
6575 CBA7                  RES  4,A
6577 CBFF                  SET  7,A
6579 CBAF                  RES  5,A
657B CBF7                  SET  6,A
657D 320EE0                LD   (EPROMC),A
6580 CDD063                CALL CE0ON
6583 3A0CE0                LD   A,(EPROMA)
```

```
6586 CDC863             CALL CEOOFF
6589 77                 LD   (HL),A
658A C9                 RET
              ;
              ;
              ;
              ;
658B ED5BFE5F VERF:     LD   DE,(STARTR)     ;SET UP ADDRESS
658F ED4BFA5F           LD   BC,(STOPR)
6593 2AFC5F             LD   HL,(STARTE)
6596 D5       LOOPV:    PUSH DE
6597 54                 LD   D,H
6598 5D                 LD   E,L             ;LD DE,HL
6599 CD3E65             CALL REC1            ;READ FROM EPROM
659C D1                 POP  DE
659D 32F25F             LD   (COUNT),A       ;TEMP STORAGE
65A0 1A                 LD   A,(DE)          ;GET DATA FROM RAM
65A1 E5                 PUSH HL
65A2 21F25F             LD   HL,COUNT
65A5 BE                 CP   (HL)            ;COMPARE TO COUNT
65A6 E1                 POP  HL
65A7 C4B365             CALL NZ,ERP          ;ERROR
65AA 13                 INC  DE
65AB 23                 INC  HL
65AC CD2D64             CALL COMPARE
65AF C8                 RET  Z               ;ALL DONE
65B0 C39665             JP   LOOPV           ;LOOP BACK
              ;
65B3 D5       ERP:      PUSH DE
65B4 CD0900             CALL NL
65B7 117F62             LD   DE,M14
65BA CD1500             CALL MSG
65BD CD3E00             CALL BELL            ;SOUND BELL
65C0 D1                 POP  DE
65C1 E5                 PUSH HL
65C2 CD0C00             CALL PRNTS
65C5 CDBA03             CALL PRTHL
65C8 CD0C00             CALL PRNTS
65CB 1A                 LD   A,(DE)
65CC CDC303             CALL PRTHX
65CF E1                 POP  HL
65D0 C9                 RET
              ;
              ;
              ;
65D1 CD0900   PRONT:    CALL NL              ;CR+LF
65D4 118662             LD   DE,M15          ;'VIDEO OR PRINTER'
65D7 CD1500             CALL MSG
65DA CDB309             CALL GETCHAR         ;SCAN KEYBOARD
65DD FE16               CP   22              ;V
65DF CAF665             JP   Z,VIDEO
65E2 FE05               CP   5               ;E
65E4 CA3866             JP   Z,PRINTER
65E7 117F62             LD   DE,M14          ;ERROR
65EA CD1500             CALL MSG
65ED CD3E00             CALL BELL
65F0 CD0900             CALL NL
65F3 C30060             JP   MAIN
              ;
              ;
```

```
65F6 ED5BFC5F VIDEO:     LD   DE,(STARTE)
65FA ED4BF85F            LD   BC,(STOPE)
65FE 2AFE5F              LD   HL,(STARTR)
6601 3EF8     LOOPH:     LD   A,0F8H
6603 32F25F              LD   (COUNT),A          ;COUNTER
6606 CD1E00              CALL BRKEY              ;CHECK FOR SHIFT/BREAK
6609 CA0060              JP   Z,MAIN
660C CD0900              CALL NL
660F E5                  PUSH HL
6610 62                  LD   H,D
6611 6B                  LD   L,E
6612 CDBA03              CALL PRTHL
6615 E1                  POP  HL
6616 CD0C00              CALL PRNTS              ;SPACE
6619 7E       DATA:      LD   A,(HL)
661A CDC303              CALL PRTHX
661D CD0C00              CALL PRNTS              ;SPACE
6620 13                  INC  DE
6621 23                  INC  HL
6622 CD2D64              CALL COMPARE
6625 CA0060              JP   Z,MAIN
6628 3AF25F              LD   A,(COUNT)
662B 3C                  INC  A
662C 32F25F              LD   (COUNT),A
662F C21966              JP   NZ,DATA
6632 CD0900              CALL NL
6635 C30166              JP   LOOPH
                      ;
                      ;
                      ;
6638 ED5BFC5F PRINTER:   LD   DE,(STARTE)
663C ED4BF85F            LD   BC,(STOPE)
6640 2AFE5F              LD   HL,(STARTR)
6643 D5                  PUSH DE
6644 CD0900              CALL NL
6647 11BC62              LD   DE,M18
664A CD1500              CALL MSG
664D 116762              LD   DE,M17
6650 CD1500              CALL MSG
6653 CDB309              CALL GETCHAR
6656 D1                  POP  DE
6657 3EF0     LOOPX:     LD   A,0F0H
6659 32F25F              LD   (COUNT),A
665C CD1E00              CALL BRKEY              ;CHECK FOR SHIFT/BREAK
665F CA0060              JP   Z,MAIN
6662 3E20                LD   A,' '
6664 CDC166              CALL ACPRNT
6667 7A                  LD   A,D
6668 CDB866              CALL ROTATE
666B CDAF66              CALL CONPRNT
666E 7A                  LD   A,D
666F CDAF66              CALL CONPRNT
6672 7B                  LD   A,E
6673 CDB866              CALL ROTATE
6676 CDAF66              CALL CONPRNT
6679 7B                  LD   A,E
667A CDAF66              CALL CONPRNT
667D 3E20     LOOPY:     LD   A,' '
667F CDC166              CALL ACPRNT
6682 7E                  LD   A,(HL)
```

```
6683 CDB866                  CALL ROTATE
6686 CDAF66                  CALL CONPRNT
6689 7E                      LD   A,(HL)
668A CDAF66                  CALL CONPRNT
668D 13                      INC  DE
668E 23                      INC  HL
668F CD2D64                  CALL COMPARE
6692 CAA766                  JP   Z,CRLF
6695 3AF25F                  LD   A,(COUNT)
6698 3C                      INC  A
6699 32F25F                  LD   (COUNT),A
669C C27D66                  JP   NZ,LOOPY
669F 3E0D                    LD   A,CR
66A1 CDC166                  CALL ACPRNT
66A4 C35766                  JP   LOOPX
                 ;
                 ;
66A7 3E0D        CRLF:       LD   A,CR
66A9 CDC166                  CALL ACPRNT
66AC C30060                  JP   MAIN
                 ;
                 ;
                 ;
                 ;
66AF E60F        CONPRNT:    AND  0FH              ;MASK IN LOWER NIBB
66B1 CDDA03                  CALL ASC
66B4 CDC166                  CALL ACPRNT
66B7 C9                      RET
                 ;
                 ;
66B8 CB0F        ROTATE:     RRC  A
66BA CB0F                    RRC  A
66BC CB0F                    RRC  A
66BE CB0F                    RRC  A
66C0 C9                      RET
                 ;
                 ;
                 ;
66C1 E5          ACPRNT:     PUSH HL
66C2 C5                      PUSH BC
66C3 2114E0                  LD   HL,0E014H
66C6 46          LOOPS:      LD   B,(HL)
66C7 CB48                    BIT  1,B
66C9 C2C666                  JP   NZ,LOOPS
66CC 77                      LD   (HL),A
66CD C1                      POP  BC
66CE E1                      POP  HL
66CF C9                      RET
                 ;
                 ;
                 ;
                 ;
                             END
```

```
ADDR       5FF4 ASC        03DA ACPRNT     66C1 BS         0008
BLANK      0020 BRKEY      001E BELL       003E BREAK      600C
CR         000D COUNT      5FF2 CURSOR     1171 CONTROL    0DDC
CHECK      60D5 CE1OFF     6388 CE1ON      63C0 CE0OFF     63C8
CE0ON      63D0 COMPARE    642D CHK        64B1 CK         64C6
CRLF       66A7 CONPRNT    66AF DAT        5FF6 DUPLICATE  610C
DATA       6619 EPROMA     E00C EPROMB     E00D EPROMC     E00E
EPROMP     E00F ERROR      639E ERROR2     64E5 ERP        65B3
FF         000C FILESIZE   1102 FILESTART  1104 FILEXEC    1106
GETCHAR    09B3 GETNUM     6317 HBUFF      10F0 HEX        03F9
HOME       6436 INIT       63E8 INIT2      640C LHEAD      0027
LDATA      002A LOOPCR     6395 LOOP       643D LOOP2      6452
LOOP3      6489 LOOPR      6532 LOOPQ      6564 LOOPV      6596
LOOPH      6601 LOOPX      6657 LOOPY      667D LOOPS      66C6
MSG        0015 MAIN       6000 M1         61EB M2         61F2
M3         61F8 M4         6201 M5         620B M6         6213
M7         6229 M8         6231 M9         6237 M10        623E
M11        6248 M12        624D M13        625B M17        6267
M14        627F M15        6286 M16        629F M18        62BC
M19        62CE M20        62D5 M21        62F5 M22        6306
NL         0009 NORMAL     0BCE OEOFF      63D8 OEON       63E0
PRTHL      03BA PRTHX      03C3 PRNTS      000C PHEAD      0021
PDATA      0024 PRNT       0012 PRINT      6034 PROGRAM    6040
PG16       608D PG32       60B1 PGMR       6447 PGM        645E
PGM32      647C PLOOP2     6487 P32        6493 PRONT      65D1
PRINTER    6638 READ       6165 READE      6507 RED1       6527
REC1       653E REDO       6559 RECO       6570 ROTATE     66B8
START      6000 STOPE      5FF8 STOPR      5FFA STARTE     5FFC
STARTR     5FFE SCREEN     0FB1 SPECIAL    0BB9 STAEP      6193
STOEP      61A9 STARA      61BF STORA      61D5 TBUFF      11A3
TIME       E419 VERIFY     60FA V25ON      63A8 V25OFF     63B0
VERF       658B VIDEO      65F6 WAIT       6439
```



EPROM Programer Circuit

To be used with a Sharp MZ80K PC