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
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDC</td>
<td>Control Data Corporation</td>
</tr>
<tr>
<td>CPU</td>
<td>Z80 Central Processing Unit</td>
</tr>
<tr>
<td>DTL</td>
<td>Diode Transistor Logic</td>
</tr>
<tr>
<td>I/O</td>
<td>Input / Output</td>
</tr>
<tr>
<td>MICR</td>
<td>Magnetic Ink Character Recognition</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>PCB</td>
<td>Printed Circuit Board</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>ROM</td>
<td>Read Only Memory</td>
</tr>
<tr>
<td>TTL</td>
<td>Transistor Transistor Logic</td>
</tr>
<tr>
<td>VFU</td>
<td>Vertical Format Unit</td>
</tr>
<tr>
<td>VFU CPU</td>
<td>Vertical Format Unit Computer</td>
</tr>
<tr>
<td>Contents</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Title</td>
<td>I</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>II</td>
</tr>
<tr>
<td>Abbreviations</td>
<td>III</td>
</tr>
<tr>
<td>Contents</td>
<td>IV</td>
</tr>
<tr>
<td>List of Figures</td>
<td>VI</td>
</tr>
<tr>
<td>1. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Preface</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Motivation</td>
<td>1</td>
</tr>
<tr>
<td>2. Background</td>
<td>2</td>
</tr>
<tr>
<td>2.1 Tractors</td>
<td>2</td>
</tr>
<tr>
<td>2.2 Drive Pulley</td>
<td>2</td>
</tr>
<tr>
<td>2.3 Drive Shaft</td>
<td>3</td>
</tr>
<tr>
<td>2.4 Clutches</td>
<td>3</td>
</tr>
<tr>
<td>2.5 Sprocket Roller</td>
<td>3</td>
</tr>
<tr>
<td>2.6 Vertical Format Unit</td>
<td>4</td>
</tr>
<tr>
<td>2.7 Timing Gears</td>
<td>7</td>
</tr>
<tr>
<td>2.8 Operation</td>
<td>7</td>
</tr>
<tr>
<td>2.9 Data Transfer</td>
<td>8</td>
</tr>
<tr>
<td>2.9.1 Transmitter/Receiver</td>
<td>8</td>
</tr>
<tr>
<td>2.9.2 Interface Voltage Levels</td>
<td>9</td>
</tr>
<tr>
<td>2.9.3 Interface Signals</td>
<td>9</td>
</tr>
<tr>
<td>2.9.3.1 Ready</td>
<td>9</td>
</tr>
<tr>
<td>2.9.3.2 Line Ready</td>
<td>10</td>
</tr>
<tr>
<td>2.9.3.3 Character Request</td>
<td>10</td>
</tr>
<tr>
<td>2.9.3.4 Data Strobe</td>
<td>10</td>
</tr>
<tr>
<td>2.9.3.5 Data Bits</td>
<td>11</td>
</tr>
<tr>
<td>2.9.3.6 Control Bit</td>
<td>11</td>
</tr>
<tr>
<td>2.9.3.7 Processor Master Clear</td>
<td>11</td>
</tr>
<tr>
<td>2.9.4 Data Interchange Technique</td>
<td>11</td>
</tr>
<tr>
<td>2.9.5 Data Sequencing</td>
<td>13</td>
</tr>
<tr>
<td>2.9.6 Programming</td>
<td>13</td>
</tr>
<tr>
<td>2.9.7 Interface Circuits</td>
<td>13</td>
</tr>
<tr>
<td>3. Design Considerations</td>
<td>14</td>
</tr>
<tr>
<td>3.1 Requirements and Constraints of the New System</td>
<td>14</td>
</tr>
<tr>
<td>4. Hardware</td>
<td>15</td>
</tr>
<tr>
<td>4.1 Description of the Microprocessor board</td>
<td>15</td>
</tr>
<tr>
<td>4.1.1 Microprocessor</td>
<td>15</td>
</tr>
<tr>
<td>4.1.2 Buffer Circuit</td>
<td>16</td>
</tr>
<tr>
<td>4.1.3 Clock Generator Circuit</td>
<td>16</td>
</tr>
<tr>
<td>4.1.4 Reset Circuit</td>
<td>16</td>
</tr>
<tr>
<td>4.1.5 Interrupt Circuit</td>
<td>16</td>
</tr>
<tr>
<td>4.1.6 Wait Circuit</td>
<td>17</td>
</tr>
<tr>
<td>4.1.7 Control and Address Decoder Circuit</td>
<td>17</td>
</tr>
<tr>
<td>4.1.8 Memory Circuit</td>
<td>17</td>
</tr>
<tr>
<td>4.1.9 Interface Circuit</td>
<td>18</td>
</tr>
<tr>
<td>4.1.10 Output and Display Circuit</td>
<td>18</td>
</tr>
<tr>
<td>4.1.11 Connector Cables</td>
<td>18</td>
</tr>
<tr>
<td>4.1.12 Construction Details</td>
<td>18</td>
</tr>
<tr>
<td>4.2 Modifications done to the Printer</td>
<td>18</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>5. Software</td>
<td></td>
</tr>
<tr>
<td>5.1 Simple Outline of the Program</td>
<td>20</td>
</tr>
<tr>
<td>5.2 Initialization Routine</td>
<td>21</td>
</tr>
<tr>
<td>5.3 Interrupt Routine</td>
<td>23</td>
</tr>
<tr>
<td>5.4 Clear Routine</td>
<td>24</td>
</tr>
<tr>
<td>5.5 Compare Routine</td>
<td>24</td>
</tr>
<tr>
<td>5.6 Main Program</td>
<td>28</td>
</tr>
<tr>
<td>5.7 Mini Computer Software Modifications</td>
<td>32</td>
</tr>
<tr>
<td>5.8 Programming Details</td>
<td>32</td>
</tr>
<tr>
<td>6. The New System In Operation</td>
<td>33</td>
</tr>
<tr>
<td>6.1 System Operation</td>
<td>33</td>
</tr>
<tr>
<td>7. Future Developments and Conclusion</td>
<td>34</td>
</tr>
<tr>
<td>7.1 Future Developments</td>
<td>34</td>
</tr>
<tr>
<td>7.2 Conclusion</td>
<td>34</td>
</tr>
<tr>
<td>References</td>
<td>35</td>
</tr>
<tr>
<td>Appendix A Mini Computer Software</td>
<td>A1</td>
</tr>
<tr>
<td>Appendix B Mini Computer and Printer Interface Circuits</td>
<td>B1</td>
</tr>
<tr>
<td>Appendix C VFU CPU Circuit Diagrams</td>
<td>C1</td>
</tr>
<tr>
<td>Appendix D VFU CPU Software Listing</td>
<td>D1</td>
</tr>
<tr>
<td>Appendix E Z80 Architecture and Instructions</td>
<td>E1</td>
</tr>
<tr>
<td>Appendix F Mini Computer Mnemonics</td>
<td>F1</td>
</tr>
<tr>
<td>Appendix G EPROM Programer Software and Block Diagram</td>
<td>G1</td>
</tr>
</tbody>
</table>
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Paper Drive Unit Assembly</td>
<td>2</td>
</tr>
<tr>
<td>2.2</td>
<td>Forms Advance Assembly</td>
<td>3</td>
</tr>
<tr>
<td>2.3</td>
<td>Format Reader Drive Sprocket</td>
<td>4</td>
</tr>
<tr>
<td>2.4</td>
<td>Format Tape Installation</td>
<td>4</td>
</tr>
<tr>
<td>2.5</td>
<td>Brush Reader</td>
<td>5</td>
</tr>
<tr>
<td>2.6</td>
<td>Format Tape Reader Assembly</td>
<td>6</td>
</tr>
<tr>
<td>2.7</td>
<td>Stop Pulse and Format Reader Alignment</td>
<td>6</td>
</tr>
<tr>
<td>2.8</td>
<td>Stop Pulse Generator and Waveforms</td>
<td>7</td>
</tr>
<tr>
<td>2.9</td>
<td>Format Reader Synchronization</td>
<td>8</td>
</tr>
<tr>
<td>2.10</td>
<td>Format Coincidence</td>
<td>8</td>
</tr>
<tr>
<td>2.11</td>
<td>Typical Receiver and Transmitter Configurations</td>
<td>9</td>
</tr>
<tr>
<td>2.12</td>
<td>Control and Data Signals Between Printer and Mini Computer</td>
<td>10</td>
</tr>
<tr>
<td>2.13</td>
<td>Interface Timing Relationships</td>
<td>12</td>
</tr>
<tr>
<td>4.1</td>
<td>Block Diagram of the VFU CPU</td>
<td>15</td>
</tr>
<tr>
<td>5.1</td>
<td>The System Flowchart</td>
<td>21</td>
</tr>
<tr>
<td>5.2</td>
<td>The Initialization Flowchart</td>
<td>22</td>
</tr>
<tr>
<td>5.3</td>
<td>The Z80 Registers</td>
<td>23</td>
</tr>
<tr>
<td>5.4</td>
<td>The Interrupt Flowchart</td>
<td>25</td>
</tr>
<tr>
<td>5.5</td>
<td>The Clear Subroutine Flowchart</td>
<td>26</td>
</tr>
<tr>
<td>5.6</td>
<td>The Compare Subroutine Flowchart</td>
<td>27</td>
</tr>
<tr>
<td>5.7A</td>
<td>The Main Program Subroutine Flowchart</td>
<td>30</td>
</tr>
<tr>
<td>5.7B</td>
<td>The Main Program Subroutine Flowchart (cont.)</td>
<td>31</td>
</tr>
</tbody>
</table>

List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Memory Map of the VFU CPU</td>
<td>17</td>
</tr>
</tbody>
</table>

Page VI
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](image)

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, i.e. 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

**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 microseconds. 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, i.e., both the six and eight lines pulse fall at the same time.

![Diagram of coil core, adjusting screws, permanent magnet, mounting post, strobe pulse rotor, and parameters for waveform.](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

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.

TRANSMITTER

RECEIVER

Figure 2.11 Typical Receiver and Transmitter configurations

2.9.2 Interface Voltage Levels

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.

```
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 \(\mu\)sec.. AS \( T_1 \) BECOMES LARGER THAN 900 nsec., \( T_3 \) WILL INCREASE BY MULTIPLES OF 900 nsec. \((T_3 = 1.8 + n \times 900 \text{nsec. WHERE } n = 1, 2, 3, \ldots)\) \( T_2 \) IS ALWAYS \( \geq 0 \).

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

SKETCH BELOW REFLECTS THE PROPAGATION DELAYS WHICH CAN BE EXPECTED.

NOTE 3: \( T_3 \) AND \( T_4 \) ARE FOR APPROXIMATELY 10 FT. OF CABLE. TYPICAL CABLE DELAY IS APPROXIMATELY 1 nsec. 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 relevant 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 MASKABLE INTERRUPT and BUSREQ lines. However circuitry to allow future use of the bus request signal...
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 an Octal Bus Transceiver (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 MI 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 kilobytes 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 separately. These signals are used to output the processed six/eight lines per inch pulse to the printer.

<table>
<thead>
<tr>
<th>Memory locations</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000 - 07FF</td>
<td>Read Only Memory</td>
</tr>
<tr>
<td>0800 - 0FFF</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>8000 - 8FFF</td>
<td>Vertical Format Unit Signals</td>
</tr>
<tr>
<td>9000 - 9FFF</td>
<td>Input / Output Ports(8255)</td>
</tr>
</tbody>
</table>

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 kilobyte ROM (Type 2716) and one 2 kilobyte RAM (Type 6116). The ROM is decoded as the first two kilobyte, the RAM as the next two kilobyte.
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 impedance 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 was 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 +5 volt line (Appendix C Sheet 10) from the printer back plane to the interface plug with a

Page 18
thicker wire as the existing 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 exercise 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 secondary 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.
5.2 Initialization Routine

Refer to Figure 5.2 and Appendix D page 1. On power on the ZBOCPU 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 Interrupt Mode 0
Set up the stack pointer

Set interrupt mode 1

Set up 8255 I/O chip

Set up buffer pointers for VFU data

Set variable command hi

Clear buffers

Enable interrupts

Set up I/O signals for start up

Start

End

Figure 5.2 The Initialization Flowchart
The CPU now starts at memory location 00. The program now sets the Stack Pointer to OFFF Hex, sets interrupt 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 swap the registers is ‘EXX’. The program uses the normal set of registers for normal data and the alternate set for VFU data.

![Z80 Registers Diagram]

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 Interrupt 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 further 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 at by 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 similar 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
Figure 5.5 The Clear Subroutine Flowchart
Compare Register D with Register B

Equal?  
False

Compare Register E with Register C

True

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 EDLINE 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 computer 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 READY signal high at the stars of procedure INPUT. The next command is to set CHARACTER REQUEST low and then to test for a control byte. This procedure is almost identical to the procedure INPUTX at the beginning 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.
Set up control lines to printer and mini computer

Test control bit

Get first character out of memory buffer

Get data

Save it and increment buffer pointer

Test printer signals

Ready?

Send a character to the printer

Increment memory pointer

Control Byte?

Copy data buffer to VFU buffer

True

False

Figure 5.7A The Main Program Flowchart
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 Programming 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 the lack of available, critical, spare parts.

7.2 Conclusion

The VFU CPU was successfully 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 benefit in the development of the PC based programs as the printers interface was by then very well known.
References


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

5 Intel (1981). Component Data Catalog


Appendix A

Mini Computer Software

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page Zero Listing</td>
<td>A1</td>
</tr>
<tr>
<td>Display Part Number</td>
<td>A2</td>
</tr>
<tr>
<td>Byteprint Start</td>
<td>A3</td>
</tr>
<tr>
<td>Printer Routine</td>
<td>A4</td>
</tr>
<tr>
<td>Sample VFU Loader Routine</td>
<td>A6</td>
</tr>
</tbody>
</table>
PAGEZERO.SR

PAGE ZERO MODULE

9/1/77

10007 .MAIN 1

PAGEZERO.SR

IDON'T LIST UNASSEMBLED CODE.

LOCATIONS 0,2,5 ARE SET FOR RDOS INTERFACE

SYSTEM START ADDRESS

10 00001
07 00000
04 00000 00000
09 00000 00000
16 00002
11 00006
12 00005
13 00005 00000
14 00000
15 EOT
0008 - MAIN
01 1 PART NUMBER MODULE
02
03 91 002007 PARTS
04 000007 0000473 PARTID
05
06
07
08
09
10 1 BEFORE ASSEMBLING THE PROGRAM CHANGE THE PNEUMONICS
11 1 P1 THROUGH P5 TO THE CORRESPONDING DIGITS OF THE PART
12 1 NUMBER. CHANGE L2 AND L2 TO THE CORRESPONDING DIGITS
13 1 OF THE LEVEL NUMBERS.
14 1
15 1 THE CONNECTION NUMBER IS ONLY CHANGED WHEN PATCHING ON
16 1 THE MACHINE.
17 1
18 1 THE SOFTWARE PREFIX IS NOT CHANGED.
20
21 000010 PART.
22 000010 000011 4. I SOFTWARE PREFIX DO NOT CHANGE
23 000011 000001 1. I SOFTWARE PREFIX DO NOT CHANGE
24
25 000012 000000 P1
26 000013 000000 P2
27 000014 000000 P3
28 000015 000000 P4
29 000016 000000 P5
30
31 000017 000000 L1
32 000020 000001 L2
33
34 000021 000000 0 I CONNECTION NUMBER MUST BE CHANGED WHEN PATCHING.
35
01 0544 00736  JMP  SN120  ICONTINUE HERE
02   I
03  0564 50510=  IFIELD 1 DONE
04  0564 027=15  LDA  0+OHVF1
05  0564 101015  ADDX  0+OSN
06  0564 000=03  JMP  SNECK  INOV+OVERFLOW
07   I
08  0564 50510=  IRA\r RETURN
09  0564 102000  EDC  0+0  IRR=-1
10  0564 00407  JMP  SNEAR  IEXIT HERE
11   I
12  0564 50510=  SWERK=5
13  0564 027=11  LDA  0+OHWK1  IWORK AREA
14  0564 028=11  LDA  1+OA12  IFIELD 1
15  0564 030=13  LDA  2+OHNL  INO. OF DIGITS
16   I
17  0564 00601  JSP  Z=1  IMOVE IT
18  0564 00400  CMVE+1
19   I
20  0564 102440  SNEQ  0+0  ICLEAN
21   I
22  0564 50510=  SWERK=5
23   I
24  0564 034=11  LDA  2+DECUSR  IRETURN TO CALLER
25  0564 001=03  JMP  Z=1-3
26   I
27  0564 00547= OVFI1  OVFI  IADDRESS OF OVFI
28  0564 100447  WKA1  ADWKA  IADDRESS OF ADWKA
29  0564 200447  A12  A12  IADDRESS OF A12
30   I
31   I
32   I
33   I
34   I
35     I
36     I
37     I
38     I
39     I
40     I
41     I
42     I
43     I
44     I
45     I
46     I
47     I
48     I
49     I
50     I
51     I
52     I
53     I
54     I
55  0564 000000  0  HYTPRT:  SHRX  ISAVE THE RETURN
56  0564 00477  STA  3+,-1
57  0564 00145  STA  0+PRI1  IHYTE ADDRESS OF PRINT BUFFER
58  0564 00443  STA  1+PRI2  I# OF CHARACTERS TO BE PRINTED
59   I
0106 MAIN
01 005A7 PT110= *
02 005A7 PT910= *
03 005A7 PT710= *
04 005A7 PT510= *
05 005A7 PT310= *
06 005A7 PT110= *
07 005A7 PT910= *
08 005A7 PT710= *
09 005A7 PT510= *
10 005A7 PT310= *
11 005A7 PT110= *
12 005A7 PT910= *
13 005A7 PT710= *
14 005A7 PT510= *
15 005A7 PT310= *
16 005A7 PT110= *
17 005A7 PT910= *
18 005A7 PT710= *
19 005A7 PT510= *
20 005A7 PT310= *
21 005A7 PT110= *
22 005A7 PT910= *
23 005A7 PT710= *
24 005A7 PT510= *
25 005A7 PT310= *
26 005A7 PT110= *
27 005A7 PT910= *
28 005A7 PT710= *
29 005A7 PT510= *
30 005A7 PT310= *
31 005A7 PT110= *
32 005A7 PT910= *
33 005A7 PT710= *
34 005A7 PT510= *
35 005A7 PT310= *
36 005A7 PT110= *
37 005A7 PT910= *
38 005A7 PT710= *
39 005A7 PT510= *
40 005A7 PT310= *
41 005A7 PT110= *
42 005A7 PT910= *
43 005A7 PT710= *
44 005A7 PT510= *
45 005A7 PT310= *
46 005A7 PT110= *
47 005A7 PT910= *
48 005A7 PT710= *
49 005A7 PT510= *
50 005A7 PT310= *
51 005A7 PT110= *
52 005A7 PT910= *
53 005A7 PT710= *
54 005A7 PT510= *
55 005A7 PT310= *
56 005A7 PT110= *
57 005A7 PT910= *
58 005A7 PT710= *
59 005A7 PT510= *
60 005A7 PT310= *

I GET STATUS OF LINE PRINTER
I WAS STATUS WORD A ONE ?
I PRINTER OFF-LINE HIT
I GET PRINTER READY HIT
I IS PRINTER READY
I YES - READY
I LINK LIGHT
I INCREMENT DELAY COUNTER
I LOOP
I LOOP
I TURN OFF-LINE LIGHT OUT
I TURN IT OFF AGAIN
FIRST TIME THROUGH, BUSY AND DONE WILL BE ZERO
I SKIP IF BUSY NOT ZERO
I PRINTER IS DONE WITH LAST LINE, PROCESS NEW
HERE WE WAIT UNTIL PRINTER DONE WITH LAST LINE
I CONTINUE
I IS TIME TO START THE NEXT LINE
I ACURRENT BYTE ADDRESS OF PRINT BUFFER
# OF BYTES TO BE PRINTED THIS LINE
ADJUST CHARACTER COUNT
I 1=BYTE ADDRESS OF CHARACTER TO BE PRINTED
I GET THE FIRST CHARACTER
I WRITE THE CHARACTER TO BE PRINTED
I SKIP IF BUSY ZERO
I LOAD CHARACTER
I HUMP PRINT BUFFER BYTE ADDRESS
I YES, INITIALIZE PRINTING
I IS LAST CHARACTER LOADED
I YES, INITIALIZE PRINTING
0107 .MAIN
01 05734 102400 SUB 0,0 IGN WAIT
02 05735 006167 JSR "SKDA ITILL LINE PRINTED.
03 04
04
04 005736 SLPT*. SERVICE LPT (LINE DONE) ENTRY POINT
07 08 EXIT 0+BYTEPRINT RETURN TO CALLER
09 05736 034725 LDA 3+BYTEPRINT
10 05737 001401 JMP 0+1+7
12 05740 000000 PPT11 0 BYTE ADDRESS PRINT BUFFER
13 05741 000000 PPT11 0 NUMBER OF CHARACTERS TO BE PRINTED
14 15 ******

A5
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

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,DUMPO 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
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,DUMPO Restore Register 0
10023 020406 LDA 0,DUMPO Restore Register 0
10024 002401 JMP @ +1 Jump Indirect +1
10025 006433 Start Of Normal Program
10026 000000 DUMP3
10027 000000 DUMP2
10030 000000 DUMP1
10031 000000 DUMPO
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

A6
Appendix B

Mini Computer and Printer Interface Circuits

Panel and Printer Control Sheet 12 - - - - B1
Panel and Printer Control Sheet 13 - - - - B2
Appendix C

VFU CPU Circuit Diagrams

Clock Generator Circuit - - - - - - - - - - C1
Reset Circuit - - - - - - - - - - - - - - - - C2
Interrupt Circuit - - - - - - - - - - C3
Cpu - - - - - - - - - - - - - - - - - - - - C4
Wait Circuit - - - - - - - - - - - - - - C5
Control and Address Decoder - - - - - - - C6
Memory Circuit - - - - - - - - - - - - - - C7
8255 Circuit - - - - - - - - - - - - - C8
Output and Display Circuit - - - - - - - - C9
Connector Cables - - - - - - - - - - - - - C10
Miscellaneous - - - - - - - - - - - - - - C11
Chip Count - - - - - - - - - - - - - - - C12
Component Layout - - - - - - - - - - - - - C13
Artwork (Component Side) - - - - - - - - - C14
Artwork (Solder Side) - - - - - - - - - - - C15
Clock Generator Circuit
Sheet 1 off 12
VFU CPU
Reset Circuit
Sheet 2 of 12
VFU CPU
8255 Circuit
Sheet 8 of 12
VFU CPU
Note 1: □ Edge Connector

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

Note 3 Characters in brackets denote Twisted Pair Earth Returns
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
<table>
<thead>
<tr>
<th>Chip Number</th>
<th>Type</th>
<th>Gates used</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>280</td>
<td>1</td>
<td>CPU</td>
</tr>
<tr>
<td>2</td>
<td>74LS244</td>
<td>All</td>
<td>Buffer</td>
</tr>
<tr>
<td>3</td>
<td>74LS244</td>
<td>All</td>
<td>Buffer</td>
</tr>
<tr>
<td>4</td>
<td>74LS244</td>
<td>7 off 8</td>
<td>Buffer</td>
</tr>
<tr>
<td>5</td>
<td>74LS245</td>
<td>All</td>
<td>Bidirectional Buffer</td>
</tr>
<tr>
<td>6</td>
<td>74LS32</td>
<td>All</td>
<td>Quad 2 Input OR</td>
</tr>
<tr>
<td>7</td>
<td>74LS04</td>
<td>4</td>
<td>Hex Inverter</td>
</tr>
<tr>
<td>8</td>
<td>74LS04</td>
<td>All</td>
<td>Hex Inverter</td>
</tr>
<tr>
<td>9</td>
<td>74123</td>
<td>All</td>
<td>Dual Monostable</td>
</tr>
<tr>
<td>10</td>
<td>2716</td>
<td>-</td>
<td>EPROM</td>
</tr>
<tr>
<td>11</td>
<td>6116</td>
<td>-</td>
<td>Static RAM</td>
</tr>
<tr>
<td>12</td>
<td>8255</td>
<td>-</td>
<td>Interface</td>
</tr>
<tr>
<td>13</td>
<td>7417</td>
<td>All</td>
<td>Hex Open Collector Buffers</td>
</tr>
<tr>
<td>14</td>
<td>7417</td>
<td>All</td>
<td>Hex Open Collector Buffers</td>
</tr>
<tr>
<td>15</td>
<td>74LS42</td>
<td>All</td>
<td>BCD 4 to 10 Line Decoder</td>
</tr>
<tr>
<td>16</td>
<td>74121</td>
<td>All</td>
<td>Monostable</td>
</tr>
<tr>
<td>17</td>
<td>7474</td>
<td>All</td>
<td>Flip Flop</td>
</tr>
<tr>
<td>18</td>
<td>74132</td>
<td>All</td>
<td>Quad Nand</td>
</tr>
<tr>
<td>19</td>
<td>74LS00</td>
<td>3</td>
<td>Quad Nand</td>
</tr>
<tr>
<td>20</td>
<td>7417</td>
<td>4</td>
<td>Hex Open Collector Buffers</td>
</tr>
<tr>
<td>21</td>
<td>74LS32</td>
<td>3</td>
<td>Quad 2 Input OR</td>
</tr>
<tr>
<td>22</td>
<td></td>
<td></td>
<td>Not Used</td>
</tr>
<tr>
<td>23</td>
<td>74LS74</td>
<td>All</td>
<td>Dual Flip Flops</td>
</tr>
<tr>
<td>24</td>
<td>74LS74</td>
<td>All</td>
<td>Dual Flip Flops</td>
</tr>
</tbody>
</table>
Appendix D

VFU CPU Software Listing

VFU CPU Software Listing  - - - - - - - - - D1
Cross Index  - - - - - - - - - - - - - - - D6
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 0F01H
VFUSTOP: EQU 0F00H

; INITIALIZE

0000 31FF0F LD SP,OFFFH ;SET STACK POINTER
0003 ED56 IM 1 ;SET INTERRUPT MODE 1
0005 3E8A LD A,8AH ;8255 CONTROL 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
0018 3EFF LD A,OFFH
001D 32010F LD (COMMAND),A ;SET COMMAND HI
0020 CD4E00 CALL CLEAR ;CLEAR MEMORY
0023 FB EI ;ENABLE INTERRUPT
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
61 002D CEDE  SET 3,(HL)  ;SET READY D116
62 002F C37500  JP  VSTART  ;INPUT/OUTPUT
63
64
65  ;LEAVE OUT THIS AREA
66  DS  06H
67
68
69  ;START OF INTERRUPT
70
71
72 0038 F3  ;DISABLE INTERRUPTS
73 0039 F5  PUSH AF
74 003A D9  ;EXCHANGE REGISTERS
75 003B 1A  LD  A,(DE)
76 003C 320080  LD  (8000H),A  ;GET DATA
77 003F C94F  BIT  1,A
78 0041 2806  JR  Z,TOF  ;OUTPUT IT TO 6/8LPI
79 0043 13  INC  DE  ;TEST FOR TOF
80 0044 D9  LEAVE:  EXX  ;EXCHANGE REGISTERS
81 0045 F1  POP  AF  ;ENABLE INTERRUPTS
82 0046 F8  EI
83 0047 E04D  RETI  ;RETURN FROM INTERRUPT
84 0049 11010E  TOF:  LD  DE,VFUSTART  ;RESET POINTER
85 004C 18F6  JR  LEAVE
86
87
88  ;CLEAR MEMORY BUFFER
89  ;IE FILL UP WITH FFH
90  ;AND FILL VFU BUFFER  ;WITH "00"
91
92
93
94 004E D5  CLEAR:  PUSH DE
95 004F F5  PUSH AF
96 0050 C5  PUSH BC
97 0051 3EFF  LOOPC:  LD  A,OFFH
98 0053 12  LD  (DE),A
99 0054 13  INC  DE
100 0055 C06E00  CALL COMPARE
101 005B C25100  JP  NZ,LOOPC
102
103
104 005B 01000F  LD  BC,VFUSTOP  ;COMPARE
105 005E 11010E  LD  DE,VFUSTART
106 0061 3E00  LOOPC2:  LD  A,00
107 0063 12  LD  (DE),A
108 0064 13  INC  DE
109 0065 C06E00  CALL COMPARE
110 006B 20F7  JR  NZ,LOOPC2
111 006A C1  POP  BC
112 006B F1  POP  AF
113 006C D1  POP  DE
114 006D C9  RET
115
116
117
118
119
120
; COMPARE DE TO BC
; ACC A DESTROYED
; Z=0 IF EQUAL

; COMPARE: 
; COMPARE D&B

; COMPARE E&C

HOME: 
; RETURN TO CALLER

; COMPARE TO BC
; ACC A DESTROYED
; Z=0 IF EQUAL

; COMPARE LD A,D

; COMPARE CP B

; COMPARE JR NZ,HOME

; COMPARE CP C

; COMPARE JR A,E

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR D&B

; COMPARE JR E&C

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR DE

; COMPARE JR BC

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR D&B

; COMPARE JR E&C

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR DE

; COMPARE JR BC

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR D&B

; COMPARE JR E&C

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR DE

; COMPARE JR BC

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR D&B

; COMPARE JR E&C

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR DE

; COMPARE JR BC

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR D&B

; COMPARE JR E&C

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR DE

; COMPARE JR BC

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR D&B

; COMPARE JR E&C

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR DE

; COMPARE JR BC

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR D&B

; COMPARE JR E&C

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR DE

; COMPARE JR BC

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR D&B

; COMPARE JR E&C

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR DE

; COMPARE JR BC

; COMPARE JR HOME:

; RETURN TO CALLER

; COMPARE JR D&B

; COMPARE JR E&C

; COMPARE JR HOME:

; RETURN TO CALLER
181 00BE CBCE  INPUT:  SET 1 , (HL) ;MAKE IT HI
182 00C0 CB8E  RES 1 , (HL) ;NOW MAKE IT LO
183 00C2 3A0190  LD A , (IN) ;TEST CONTROL BIT
184 00C5 CB77  BIT 6 , A
185 00C7 200C  JR NZ , OUTPUT4 ;ITS A 1 , END LINE INP
186 00C9 7E  INPUT2:  LD A , (HL) ;TEST STROBE IN
187 00CA CB67  BIT 4 , A
188 00CC 20FB  JR NZ , INPUT2 ;NOT YET SO LOOP BACK
189
190 00CE 3A0190  LD A , (IN) ;GET DATA
191 00D1 12  LD (DE) , A ;STORE IN MEMORY
192 00D2 13  INC DE ;INCREMENT MEMORY POINTER
193 00D3 18E9  JR INPUT
194
195 00D5 12  OUTPUT4:  LD (DE) , A
196 00D6 47  LD B , A ;SAVE CONTROL WORD
197 00D7 1B  REDUCE1:  DEC DE ;DECREMENT MEMPOINTER
198 00DB 1A  LD A , (DE) ;GET DATA
199 00D9 FE20  CP 400 ;CHECK FOR A SPACE
200 00DB 2B34  JR NZ , REDUCE2 ;REPLACE WITH CONTROLWORD
201
202
203 00DD 110008  LD DE , START ;GET START OF BUFFER
204 00E0 1A  LD A , (DE) ;CHECK DATA
205 00E1 FE01  CP 1 ;COMPARE TO 1
206 00E3 2B30  JR NZ , OUTPUT2 ;YES , SET UP VFU TABLE
207
208
209 00E5 CBD6  OUTPUT1:  SET 2 , (HL) ;SET LINE READY
210 00E7 110008  OUTPUT:  LD DE , START ;RESET DE
211 00EA 1A  LD A , (DE) ;GET DATA
212 00EB 320090  OUTPUT3:  LD (OUT) , A ;SEND IT
213 00EE CB86  RES 0 , (HL) ;RESET STROBE
214 00F0 7E  CRIN:  LD A , (HL) ;CHECK FLAGS
215 00F1 CB6F  BIT 5 , A ;WAIT FOR CHARREQ
216 00F3 28F8  JR NZ , CRIN ;WAIT FOR CHARREQ
217 00F5 CB66  SET 0 , (HL) ;SET STROBE
218 00F7 1A  LD A , (DE) ;GET DATA AGAIN
219 00F8 CB77  BIT 6 , A ;CHECK FOR CONTROLBIT
220 00FA 20A9  JR NZ , LOOP ;YES , EOL , START NEWLINE
221
222 00FC 13  INC DE ;NO , INCREMENT POINTER
223 00FD 18EB  JR OUTPUT3 ;AND LOOP
224
225 00FF CBD6  SETRO:  SET 3 , (HL) ;SET READY
226 0101 CB66  SET 2 , (HL) ;SET LINE READY
227 0103 CBCE  SET 1 , (HL) ;SET CHARREQ
228 0105 18A4  JR TESTRI
229
230
231 0107 CBD6  SETLRO:  SET 2 , (HL) ;SET LINE READY
232 0109 CBCE  SET 1 , (HL) ;SET CHARREQ
233 010B 189E  JR TESTRI
234
235
236 010D CBCE  SETCRO:  SET 1 , (HL) ;SET CHARREQ
237 010F 189A  JR TESTRI
238
239
240
241 0111 47  REDUCE2 :  LD  B,A ;GET CONTROLL WORD
242 0112 12  LD (DE),A ;AND STORE IT
243 0113 1BC2  JR  REDUCE1 ;REPEAT LOOP
244
245
246 0115 11010E  SETVFU:  LD  DE,VFUSTART ;TABLE START
247 0118 010108  LD  BC,START+1 ;BUFFER START+1(DATA)
248 011B 0A  LOOPVFU:  LD  A,(BC) ;GET DATA
249 011C 12  LD  (DE),A ;STORE IT
250 011D 13  INC  DE
251 011E 03  INC  BC
252 011F CB77  BIT  6,A ;TEST FOR CONTROLL BIT
253 0121 2BF8  JR  Z,LOOPVFU
254 0123 C3A500  JP  LOOP
255
256
257  END

05
<table>
<thead>
<tr>
<th>COMMAND</th>
<th>0F01 CLEAR</th>
<th>004E COMPARE</th>
<th>006E CRIN</th>
<th>00F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>EOLINE</td>
<td>0096 HOME</td>
<td>0074 IN</td>
<td>9001 INC</td>
<td>9002</td>
</tr>
<tr>
<td>INPUTX</td>
<td>007F INPUTV</td>
<td>008A INPUT</td>
<td>008E INPUT2</td>
<td>00C9</td>
</tr>
<tr>
<td>LEAVE</td>
<td>0044 LOOPC</td>
<td>0051 LOOPC2</td>
<td>0061 LOOPCR</td>
<td>009E</td>
</tr>
<tr>
<td>LOOP</td>
<td>00A5 LOOPVFU</td>
<td>0118 OUT</td>
<td>9000 OUTC</td>
<td>9002</td>
</tr>
<tr>
<td>OUTPUT4</td>
<td>00D5 OUTPUT1</td>
<td>00E5 OUTPUT</td>
<td>00E7 OUTPUT3</td>
<td>00EA</td>
</tr>
<tr>
<td>REDUCE1</td>
<td>00D7 REDUCE2</td>
<td>0111 START</td>
<td>0800 STOP</td>
<td>0E00</td>
</tr>
<tr>
<td>SETRO</td>
<td>00FF SETLRO</td>
<td>0107 SETCRO</td>
<td>010D SETVFU</td>
<td>0115</td>
</tr>
<tr>
<td>TOF</td>
<td>0049 TESTRI</td>
<td>00A8 TESTLRI</td>
<td>0082 TESTCRI</td>
<td>0089</td>
</tr>
<tr>
<td>VFCH</td>
<td>0009 VFUSTART</td>
<td>0E01 VFUSTOP</td>
<td>0F00 VSTART</td>
<td>0075</td>
</tr>
</tbody>
</table>
Appendix E

Z80 Architecture and Instructions

Internal Z80 Organization  ---  ---  ---  ---  E1
Z80 Instruction Codes  ---  ---  ---  ---  E2
Internal Z80 Organization
## Z80 INSTRUCTION CODES

(The literal 'd' is shown as 0< in the object code.)

<table>
<thead>
<tr>
<th>OBJ CODE</th>
<th>SOURCE STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>8E</td>
<td>ADC A, HL:1</td>
</tr>
<tr>
<td>DD8E05</td>
<td>ADC A, (HL)+d</td>
</tr>
<tr>
<td>FD8E05</td>
<td>ADC A, (HL)+d</td>
</tr>
<tr>
<td>8F</td>
<td>ADC A, B</td>
</tr>
<tr>
<td>89</td>
<td>ADC A, C</td>
</tr>
<tr>
<td>8A</td>
<td>ADC A, D</td>
</tr>
<tr>
<td>8B</td>
<td>ADC A, E</td>
</tr>
<tr>
<td>8C</td>
<td>ADC A, H</td>
</tr>
<tr>
<td>8D</td>
<td>ADC A, L</td>
</tr>
<tr>
<td>CE20</td>
<td>ADC A, H</td>
</tr>
<tr>
<td>ED6A</td>
<td>ADC HL, BC</td>
</tr>
<tr>
<td>ED6A</td>
<td>ADC HL, DE</td>
</tr>
<tr>
<td>ED6A</td>
<td>ADC HL, HL</td>
</tr>
<tr>
<td>ED67A</td>
<td>ADC HL, SP</td>
</tr>
<tr>
<td>DD8605</td>
<td>ADD A, HL:1</td>
</tr>
<tr>
<td>FD8605</td>
<td>ADD A, (HL)+d</td>
</tr>
<tr>
<td>87</td>
<td>ADD A, A</td>
</tr>
<tr>
<td>80</td>
<td>ADD A, B</td>
</tr>
<tr>
<td>81</td>
<td>ADD A, C</td>
</tr>
<tr>
<td>82</td>
<td>ADD A, D</td>
</tr>
<tr>
<td>83</td>
<td>ADD A, E</td>
</tr>
<tr>
<td>84</td>
<td>ADD A, H</td>
</tr>
<tr>
<td>85</td>
<td>ADD A, L</td>
</tr>
<tr>
<td>CE20</td>
<td>ADD A, H</td>
</tr>
<tr>
<td>09</td>
<td>ADD HL, BC</td>
</tr>
<tr>
<td>19</td>
<td>ADD HL, DE</td>
</tr>
<tr>
<td>29</td>
<td>ADD HL, HL</td>
</tr>
<tr>
<td>39</td>
<td>ADD HL, SP</td>
</tr>
<tr>
<td>DD09</td>
<td>ADD IX, BC</td>
</tr>
<tr>
<td>DD19</td>
<td>ADD IX, DE</td>
</tr>
<tr>
<td>DD29</td>
<td>ADD IX, IX</td>
</tr>
<tr>
<td>DD39</td>
<td>ADD IX, SP</td>
</tr>
<tr>
<td>FD09</td>
<td>ADD IY, BC</td>
</tr>
<tr>
<td>FD19</td>
<td>ADD IY, DE</td>
</tr>
<tr>
<td>FD29</td>
<td>ADD IY, IX</td>
</tr>
<tr>
<td>FD29</td>
<td>ADD IY, SP</td>
</tr>
<tr>
<td>AE</td>
<td>AND (HL)</td>
</tr>
<tr>
<td>DD6A05</td>
<td>AND (IX)+d</td>
</tr>
<tr>
<td>FD6A05</td>
<td>AND (IY)+d</td>
</tr>
<tr>
<td>A7</td>
<td>AND A</td>
</tr>
<tr>
<td>A9</td>
<td>AND B</td>
</tr>
<tr>
<td>A1</td>
<td>AND C</td>
</tr>
<tr>
<td>A2</td>
<td>AND D</td>
</tr>
<tr>
<td>A3</td>
<td>AND E</td>
</tr>
<tr>
<td>A4</td>
<td>AND H</td>
</tr>
<tr>
<td>A5</td>
<td>AND L</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OBJ CODE</th>
<th>SOURCE STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>E620</td>
<td>AND a</td>
</tr>
<tr>
<td>CB46</td>
<td>BIT 0, HL:1</td>
</tr>
<tr>
<td>DDC80546</td>
<td>BIT 0,(IX)+d</td>
</tr>
<tr>
<td>FDC80546</td>
<td>BIT 0,(IY)+d</td>
</tr>
<tr>
<td>CB47</td>
<td>BIT 0, A</td>
</tr>
<tr>
<td>CB40</td>
<td>BIT 0, B</td>
</tr>
<tr>
<td>CB41</td>
<td>BIT 0, C</td>
</tr>
<tr>
<td>CB42</td>
<td>BIT 0, D</td>
</tr>
<tr>
<td>CB43</td>
<td>BIT 0, E</td>
</tr>
<tr>
<td>CB44</td>
<td>BIT 0, H</td>
</tr>
<tr>
<td>CB45</td>
<td>BIT 0, L</td>
</tr>
<tr>
<td>CB4E</td>
<td>BIT 1, HL:1</td>
</tr>
<tr>
<td>DDC80546</td>
<td>BIT 1,(IX)+d</td>
</tr>
<tr>
<td>FDC80546</td>
<td>BIT 1,(IY)+d</td>
</tr>
<tr>
<td>CB4F</td>
<td>BIT 1, A</td>
</tr>
<tr>
<td>CB48</td>
<td>BIT 1, B</td>
</tr>
<tr>
<td>CB49</td>
<td>BIT 1, C</td>
</tr>
<tr>
<td>CB4A</td>
<td>BIT 1, D</td>
</tr>
<tr>
<td>CB48</td>
<td>BIT 1, E</td>
</tr>
<tr>
<td>CB4C</td>
<td>BIT 1, H</td>
</tr>
<tr>
<td>CB4D</td>
<td>BIT 1, L</td>
</tr>
<tr>
<td>CB56</td>
<td>BIT 2, HL:1</td>
</tr>
<tr>
<td>DDC80556</td>
<td>BIT 2,(IX)+d</td>
</tr>
<tr>
<td>FDC80556</td>
<td>BIT 2,(IY)+d</td>
</tr>
<tr>
<td>CB57</td>
<td>BIT 2, A</td>
</tr>
<tr>
<td>CB50</td>
<td>BIT 2, B</td>
</tr>
<tr>
<td>CB51</td>
<td>BIT 2, C</td>
</tr>
<tr>
<td>CB52</td>
<td>BIT 2, D</td>
</tr>
<tr>
<td>CB53</td>
<td>BIT 2, E</td>
</tr>
<tr>
<td>CB54</td>
<td>BIT 2, H</td>
</tr>
<tr>
<td>CB55</td>
<td>BIT 2, L</td>
</tr>
<tr>
<td>CB5E</td>
<td>BIT 3, HL:1</td>
</tr>
<tr>
<td>DDC8055E</td>
<td>BIT 3,(IX)+d</td>
</tr>
<tr>
<td>FDC8055E</td>
<td>BIT 3,(IY)+d</td>
</tr>
<tr>
<td>CB5F</td>
<td>BIT 3, A</td>
</tr>
<tr>
<td>CB58</td>
<td>BIT 3, B</td>
</tr>
<tr>
<td>CB55</td>
<td>BIT 3, C</td>
</tr>
<tr>
<td>CB5A</td>
<td>BIT 3, D</td>
</tr>
<tr>
<td>CB58</td>
<td>BIT 3, E</td>
</tr>
<tr>
<td>CB5C</td>
<td>BIT 3, H</td>
</tr>
<tr>
<td>CB5D</td>
<td>BIT 3, L</td>
</tr>
<tr>
<td>CB66</td>
<td>BIT 4, HL:1</td>
</tr>
<tr>
<td>DDC80566</td>
<td>BIT 4,(IX)+d</td>
</tr>
<tr>
<td>FDC80566</td>
<td>BIT 4,(IY)+d</td>
</tr>
<tr>
<td>CB67</td>
<td>BIT 4, A</td>
</tr>
<tr>
<td>CB60</td>
<td>BIT 4, B</td>
</tr>
<tr>
<td>CB61</td>
<td>BIT 4, C</td>
</tr>
<tr>
<td>CB62</td>
<td>BIT 4, D</td>
</tr>
</tbody>
</table>

E2
<table>
<thead>
<tr>
<th>OBJ CODE</th>
<th>SOURCE STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CB63</td>
<td>BIT 4.E</td>
</tr>
<tr>
<td>CB64</td>
<td>BIT 4.H</td>
</tr>
<tr>
<td>CB65</td>
<td>BIT 4.L</td>
</tr>
<tr>
<td>CB66</td>
<td>BIT 5.L(HL)</td>
</tr>
<tr>
<td>DBC08576</td>
<td>BIT 5.(IX+d)</td>
</tr>
<tr>
<td>FDC08576</td>
<td>BIT 5.(IX+d)</td>
</tr>
<tr>
<td>CB8F</td>
<td>BIT 5.A</td>
</tr>
<tr>
<td>CB88</td>
<td>BIT 5.B</td>
</tr>
<tr>
<td>CB89</td>
<td>BIT 5.C</td>
</tr>
<tr>
<td>CB8A</td>
<td>BIT 5.D</td>
</tr>
<tr>
<td>CB8B</td>
<td>BIT 5.E</td>
</tr>
<tr>
<td>CB8C</td>
<td>BIT 5.H</td>
</tr>
<tr>
<td>CB8D</td>
<td>BIT 5.L</td>
</tr>
<tr>
<td>CB76</td>
<td>BIT 6.L(HL)</td>
</tr>
<tr>
<td>DBC08576</td>
<td>BIT 6.(IX+d)</td>
</tr>
<tr>
<td>FDC08576</td>
<td>BIT 6.(IX+d)</td>
</tr>
<tr>
<td>CB77</td>
<td>BIT 6.A</td>
</tr>
<tr>
<td>CB78</td>
<td>BIT 6.B</td>
</tr>
<tr>
<td>CB79</td>
<td>BIT 6.C</td>
</tr>
<tr>
<td>CB70</td>
<td>BIT 6.D</td>
</tr>
<tr>
<td>CB71</td>
<td>BIT 6.E</td>
</tr>
<tr>
<td>CB74</td>
<td>BIT 6.H</td>
</tr>
<tr>
<td>CB75</td>
<td>BIT 6.L</td>
</tr>
<tr>
<td>CB7E</td>
<td>BIT 7.L(HL)</td>
</tr>
<tr>
<td>DBC08576</td>
<td>BIT 7.(IX+d)</td>
</tr>
<tr>
<td>FDC08576</td>
<td>BIT 7.(IX+d)</td>
</tr>
<tr>
<td>CB7F</td>
<td>BIT 7.A</td>
</tr>
<tr>
<td>CB77</td>
<td>BIT 7.B</td>
</tr>
<tr>
<td>CB7C</td>
<td>BIT 7.C</td>
</tr>
<tr>
<td>CB7A</td>
<td>BIT 7.D</td>
</tr>
<tr>
<td>CB78</td>
<td>BIT 7.E</td>
</tr>
<tr>
<td>CB7C</td>
<td>BIT 7.H</td>
</tr>
<tr>
<td>CB70</td>
<td>BIT 7.L</td>
</tr>
<tr>
<td>DBC405</td>
<td>CALL C,mn</td>
</tr>
<tr>
<td>FBC405</td>
<td>CALL M,mn</td>
</tr>
<tr>
<td>CDB405</td>
<td>CALL N2,mn</td>
</tr>
<tr>
<td>FCB405</td>
<td>CALL P,mn</td>
</tr>
<tr>
<td>ECD405</td>
<td>CALL PE,mn</td>
</tr>
<tr>
<td>CC48405</td>
<td>CALL PM,mn</td>
</tr>
<tr>
<td>CD4805</td>
<td>CALL Z,mn</td>
</tr>
<tr>
<td>3F</td>
<td>CCF</td>
</tr>
<tr>
<td>8E</td>
<td>CP</td>
</tr>
<tr>
<td>80E5</td>
<td>CP</td>
</tr>
<tr>
<td>FDB805</td>
<td>CP</td>
</tr>
<tr>
<td>8F</td>
<td>CP</td>
</tr>
<tr>
<td>89</td>
<td>CP</td>
</tr>
<tr>
<td>8A</td>
<td>CP</td>
</tr>
<tr>
<td>8B</td>
<td>CP</td>
</tr>
<tr>
<td>8C</td>
<td>CP</td>
</tr>
<tr>
<td>8D</td>
<td>CP</td>
</tr>
<tr>
<td>8E</td>
<td>CP</td>
</tr>
<tr>
<td>8F</td>
<td>CP</td>
</tr>
<tr>
<td>8G</td>
<td>CP</td>
</tr>
<tr>
<td>8H</td>
<td>CP</td>
</tr>
<tr>
<td>8I</td>
<td>CP</td>
</tr>
<tr>
<td>8J</td>
<td>CP</td>
</tr>
<tr>
<td>8K</td>
<td>CP</td>
</tr>
<tr>
<td>8L</td>
<td>CP</td>
</tr>
<tr>
<td>FE20</td>
<td>CP</td>
</tr>
<tr>
<td>ED20</td>
<td>CP</td>
</tr>
<tr>
<td>ED89</td>
<td>CPDR</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OBJ CODE</th>
<th>SOURCE STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDB1</td>
<td>CPCH</td>
</tr>
<tr>
<td>EDA1</td>
<td>CPI</td>
</tr>
<tr>
<td>2F</td>
<td>CPL</td>
</tr>
<tr>
<td>27</td>
<td>DAA</td>
</tr>
<tr>
<td>35</td>
<td>DEC</td>
</tr>
<tr>
<td>DD3505</td>
<td>DEC</td>
</tr>
<tr>
<td>FD3505</td>
<td>DEC</td>
</tr>
<tr>
<td>3D</td>
<td>DEC</td>
</tr>
<tr>
<td>06</td>
<td>DEC</td>
</tr>
<tr>
<td>08</td>
<td>DEC</td>
</tr>
<tr>
<td>00</td>
<td>DEC</td>
</tr>
<tr>
<td>10</td>
<td>DEC</td>
</tr>
<tr>
<td>29</td>
<td>DEC</td>
</tr>
<tr>
<td>28</td>
<td>DEC</td>
</tr>
<tr>
<td>DD28</td>
<td>DEC</td>
</tr>
<tr>
<td>FD28</td>
<td>DEC</td>
</tr>
<tr>
<td>20</td>
<td>DEC</td>
</tr>
<tr>
<td>38</td>
<td>DEC</td>
</tr>
<tr>
<td>F3</td>
<td>DJNZ</td>
</tr>
<tr>
<td>F8</td>
<td>E</td>
</tr>
<tr>
<td>E3</td>
<td>EX</td>
</tr>
<tr>
<td>DDE3</td>
<td>EX</td>
</tr>
<tr>
<td>FDE3</td>
<td>EX</td>
</tr>
<tr>
<td>00</td>
<td>EX</td>
</tr>
<tr>
<td>EE</td>
<td>EX</td>
</tr>
<tr>
<td>D0</td>
<td>EX</td>
</tr>
<tr>
<td>76</td>
<td>MALT</td>
</tr>
<tr>
<td>E046</td>
<td>IM</td>
</tr>
<tr>
<td>E056</td>
<td>IM</td>
</tr>
<tr>
<td>E051</td>
<td>IM</td>
</tr>
<tr>
<td>E078</td>
<td>IN</td>
</tr>
<tr>
<td>E040</td>
<td>IN</td>
</tr>
<tr>
<td>E048</td>
<td>IN</td>
</tr>
<tr>
<td>E050</td>
<td>IN</td>
</tr>
<tr>
<td>E058</td>
<td>IN</td>
</tr>
<tr>
<td>E060</td>
<td>IN</td>
</tr>
<tr>
<td>E068</td>
<td>IN</td>
</tr>
<tr>
<td>24</td>
<td>IN</td>
</tr>
<tr>
<td>0D3405</td>
<td>INC</td>
</tr>
<tr>
<td>FD3405</td>
<td>INC</td>
</tr>
<tr>
<td>3C</td>
<td>INC</td>
</tr>
<tr>
<td>04</td>
<td>INC</td>
</tr>
<tr>
<td>02</td>
<td>INC</td>
</tr>
<tr>
<td>0C</td>
<td>INC</td>
</tr>
<tr>
<td>14</td>
<td>INC</td>
</tr>
<tr>
<td>13</td>
<td>INC</td>
</tr>
<tr>
<td>1C</td>
<td>INC</td>
</tr>
<tr>
<td>24</td>
<td>INC</td>
</tr>
<tr>
<td>22</td>
<td>INC</td>
</tr>
<tr>
<td>0D22</td>
<td>INC</td>
</tr>
<tr>
<td>2C</td>
<td>INC</td>
</tr>
<tr>
<td>32</td>
<td>INC</td>
</tr>
<tr>
<td>DB20</td>
<td>IN</td>
</tr>
<tr>
<td>OBJ CODE</td>
<td>SOURCE STATEMENT</td>
</tr>
<tr>
<td>----------</td>
<td>------------------</td>
</tr>
<tr>
<td>58</td>
<td>LD H, E</td>
</tr>
<tr>
<td>5C</td>
<td>LD H, E</td>
</tr>
<tr>
<td>5D</td>
<td>LD H, E</td>
</tr>
<tr>
<td>5E20</td>
<td>LD H, E</td>
</tr>
<tr>
<td>66</td>
<td>LD H, (HL)</td>
</tr>
<tr>
<td>D06605</td>
<td>LD H, (IX+D)</td>
</tr>
<tr>
<td>F06605</td>
<td>LD H, (IX-Y)</td>
</tr>
<tr>
<td>F7</td>
<td>LD A</td>
</tr>
<tr>
<td>60</td>
<td>LD B</td>
</tr>
<tr>
<td>61</td>
<td>LD C</td>
</tr>
<tr>
<td>62</td>
<td>LD D</td>
</tr>
<tr>
<td>63</td>
<td>LD E</td>
</tr>
<tr>
<td>64</td>
<td>LD H</td>
</tr>
<tr>
<td>65</td>
<td>LD L</td>
</tr>
<tr>
<td>2620</td>
<td>LD M, A</td>
</tr>
<tr>
<td>2A8405</td>
<td>LD M, (HL)</td>
</tr>
<tr>
<td>218405</td>
<td>LD M, (IX+D)</td>
</tr>
<tr>
<td>ED47</td>
<td>LD (I)</td>
</tr>
<tr>
<td>D02A8405</td>
<td>LD (IX)</td>
</tr>
<tr>
<td>D0218405</td>
<td>LD (IX)</td>
</tr>
<tr>
<td>F02A8405</td>
<td>LD (IX)</td>
</tr>
<tr>
<td>F0218405</td>
<td>LD (IX)</td>
</tr>
<tr>
<td>FGE5</td>
<td>LD (H)</td>
</tr>
<tr>
<td>GD6E05</td>
<td>LD (IX-Y)</td>
</tr>
<tr>
<td>GD6E05</td>
<td>LD (IY+X)</td>
</tr>
<tr>
<td>6F</td>
<td>LD L</td>
</tr>
<tr>
<td>66</td>
<td>LD L</td>
</tr>
<tr>
<td>69</td>
<td>LD L</td>
</tr>
<tr>
<td>6A</td>
<td>LD L</td>
</tr>
<tr>
<td>6B</td>
<td>LD L</td>
</tr>
<tr>
<td>6C</td>
<td>LD L</td>
</tr>
<tr>
<td>6D</td>
<td>LD L</td>
</tr>
<tr>
<td>2E20</td>
<td>LD L</td>
</tr>
<tr>
<td>ED4F</td>
<td>LD R</td>
</tr>
<tr>
<td>ED88</td>
<td>LD (SP)</td>
</tr>
<tr>
<td>F5</td>
<td>LD SP, (HL)</td>
</tr>
<tr>
<td>D0F9</td>
<td>LD SP, (IX)</td>
</tr>
<tr>
<td>F0F9</td>
<td>LD SP, (IX)</td>
</tr>
<tr>
<td>3E8405</td>
<td>LD SP, (IX)</td>
</tr>
<tr>
<td>3E48</td>
<td>LD (DD)</td>
</tr>
<tr>
<td>E0DB</td>
<td>LD (I)</td>
</tr>
<tr>
<td>E0A0</td>
<td>LD (I)</td>
</tr>
<tr>
<td>E080</td>
<td>LD (I)</td>
</tr>
<tr>
<td>E0A4</td>
<td>NEG</td>
</tr>
<tr>
<td>00</td>
<td>NOP</td>
</tr>
<tr>
<td>86</td>
<td>OUT H, (HL)</td>
</tr>
<tr>
<td>D08605</td>
<td>OUT (IX+D)</td>
</tr>
<tr>
<td>F08605</td>
<td>OUT (IX-Y)</td>
</tr>
<tr>
<td>67</td>
<td>QRF A</td>
</tr>
<tr>
<td>80</td>
<td>QRF B</td>
</tr>
<tr>
<td>81</td>
<td>QRF C</td>
</tr>
<tr>
<td>82</td>
<td>QRF D</td>
</tr>
<tr>
<td>83</td>
<td>QRF E</td>
</tr>
<tr>
<td>84</td>
<td>QRF H</td>
</tr>
<tr>
<td>85</td>
<td>QRF L</td>
</tr>
<tr>
<td>F620</td>
<td>QRF M</td>
</tr>
<tr>
<td>EDB8</td>
<td>OUT DM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OBJ CODE</th>
<th>SOURCE STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>E083</td>
<td>OUT CLA</td>
</tr>
<tr>
<td>E079</td>
<td>OUT CLI</td>
</tr>
<tr>
<td>E041</td>
<td>OUT CLI</td>
</tr>
<tr>
<td>E049</td>
<td>OUT CLI</td>
</tr>
<tr>
<td>E051</td>
<td>OUT CI</td>
</tr>
<tr>
<td>E059</td>
<td>OUT CIE</td>
</tr>
<tr>
<td>E061</td>
<td>OUT CIL</td>
</tr>
<tr>
<td>E069</td>
<td>OUT CLM</td>
</tr>
<tr>
<td>D320</td>
<td>OUT M1 A</td>
</tr>
<tr>
<td>E0A6</td>
<td>OUT D</td>
</tr>
<tr>
<td>E0A3</td>
<td>OUT F</td>
</tr>
<tr>
<td>F1</td>
<td>POP AF</td>
</tr>
<tr>
<td>C1</td>
<td>POP BC</td>
</tr>
<tr>
<td>D1</td>
<td>POP DE</td>
</tr>
<tr>
<td>E1</td>
<td>POP HL</td>
</tr>
<tr>
<td>D0E1</td>
<td>POP IX</td>
</tr>
<tr>
<td>F0E1</td>
<td>POP IY</td>
</tr>
<tr>
<td>F5</td>
<td>PUSH AF</td>
</tr>
<tr>
<td>C5</td>
<td>PUSH BC</td>
</tr>
<tr>
<td>D5</td>
<td>PUSH DE</td>
</tr>
<tr>
<td>E5</td>
<td>PUSH HL</td>
</tr>
<tr>
<td>D0E5</td>
<td>PUSH IX</td>
</tr>
<tr>
<td>CF86</td>
<td>RES (HL)</td>
</tr>
<tr>
<td>D0CB0586</td>
<td>RES (IX-D)</td>
</tr>
<tr>
<td>F0CB0586</td>
<td>RES (IX-D)</td>
</tr>
<tr>
<td>CB87</td>
<td>RES A</td>
</tr>
<tr>
<td>CB80</td>
<td>RES B</td>
</tr>
<tr>
<td>CB81</td>
<td>RES C</td>
</tr>
<tr>
<td>CB82</td>
<td>RES D</td>
</tr>
<tr>
<td>CB83</td>
<td>RES E</td>
</tr>
<tr>
<td>CB84</td>
<td>RES H</td>
</tr>
<tr>
<td>CB85</td>
<td>RES L</td>
</tr>
<tr>
<td>CB88</td>
<td>RES M2</td>
</tr>
<tr>
<td>D0CB058E</td>
<td>RES (IX-D)</td>
</tr>
<tr>
<td>F0CB058E</td>
<td>RES (IX-D)</td>
</tr>
<tr>
<td>CB8F</td>
<td>RES A</td>
</tr>
<tr>
<td>CB88</td>
<td>RES B</td>
</tr>
<tr>
<td>CB89</td>
<td>RES C</td>
</tr>
<tr>
<td>CB8A</td>
<td>RES D</td>
</tr>
<tr>
<td>CB88</td>
<td>RES E</td>
</tr>
<tr>
<td>CB8C</td>
<td>RES H</td>
</tr>
<tr>
<td>CB8D</td>
<td>RES L</td>
</tr>
<tr>
<td>CB96</td>
<td>RES M2</td>
</tr>
<tr>
<td>D0CB0596</td>
<td>RES (IX-D)</td>
</tr>
<tr>
<td>F0CB0596</td>
<td>RES (IX-D)</td>
</tr>
<tr>
<td>CB97</td>
<td>RES D</td>
</tr>
<tr>
<td>CB99</td>
<td>RES D</td>
</tr>
<tr>
<td>CB91</td>
<td>RES C</td>
</tr>
<tr>
<td>CB92</td>
<td>RES D</td>
</tr>
<tr>
<td>CB93</td>
<td>RES E</td>
</tr>
<tr>
<td>CB94</td>
<td>RES H</td>
</tr>
<tr>
<td>CB95</td>
<td>RES L</td>
</tr>
<tr>
<td>CB9E</td>
<td>RES M2</td>
</tr>
<tr>
<td>D0CB059E</td>
<td>RES (IX-D)</td>
</tr>
<tr>
<td>F0CB059E</td>
<td>RES (IX-D)</td>
</tr>
<tr>
<td>OBJ CODE</td>
<td>SOURCE STATEMENT</td>
</tr>
<tr>
<td>----------</td>
<td>------------------</td>
</tr>
<tr>
<td>C89F</td>
<td>RES 3A</td>
</tr>
<tr>
<td>C89E</td>
<td>RES 3B</td>
</tr>
<tr>
<td>C89D</td>
<td>RES 3C</td>
</tr>
<tr>
<td>C89C</td>
<td>RES 3D</td>
</tr>
<tr>
<td>C89B</td>
<td>RES 3E</td>
</tr>
<tr>
<td>C89A</td>
<td>RES 3F</td>
</tr>
<tr>
<td>C88F</td>
<td>RES 4A</td>
</tr>
<tr>
<td>C88E</td>
<td>RES 4B</td>
</tr>
<tr>
<td>C88D</td>
<td>RES 4C</td>
</tr>
<tr>
<td>C88C</td>
<td>RES 4D</td>
</tr>
<tr>
<td>C88B</td>
<td>RES 4E</td>
</tr>
<tr>
<td>C88A</td>
<td>RES 4F</td>
</tr>
<tr>
<td>C87F</td>
<td>RES 5A</td>
</tr>
<tr>
<td>C87E</td>
<td>RES 5B</td>
</tr>
<tr>
<td>C87D</td>
<td>RES 5C</td>
</tr>
<tr>
<td>C87C</td>
<td>RES 5D</td>
</tr>
<tr>
<td>C87B</td>
<td>RES 5E</td>
</tr>
<tr>
<td>C87A</td>
<td>RES 5F</td>
</tr>
<tr>
<td>C86F</td>
<td>RES 6A</td>
</tr>
<tr>
<td>C86E</td>
<td>RES 6B</td>
</tr>
<tr>
<td>C86D</td>
<td>RES 6C</td>
</tr>
<tr>
<td>C86C</td>
<td>RES 6D</td>
</tr>
<tr>
<td>C86B</td>
<td>RES 6E</td>
</tr>
<tr>
<td>C86A</td>
<td>RES 6F</td>
</tr>
<tr>
<td>C85F</td>
<td>RES 7A</td>
</tr>
<tr>
<td>C85E</td>
<td>RES 7B</td>
</tr>
<tr>
<td>C85D</td>
<td>RES 7C</td>
</tr>
<tr>
<td>C85C</td>
<td>RES 7D</td>
</tr>
<tr>
<td>C85B</td>
<td>RES 7E</td>
</tr>
<tr>
<td>C85A</td>
<td>RES 7F</td>
</tr>
<tr>
<td>C84F</td>
<td>RES 8A</td>
</tr>
<tr>
<td>C84E</td>
<td>RES 8B</td>
</tr>
<tr>
<td>C84D</td>
<td>RES 8C</td>
</tr>
<tr>
<td>C84C</td>
<td>RES 8D</td>
</tr>
<tr>
<td>C84B</td>
<td>RES 8E</td>
</tr>
<tr>
<td>C84A</td>
<td>RES 8F</td>
</tr>
<tr>
<td>C83F</td>
<td>RES 9A</td>
</tr>
<tr>
<td>C83E</td>
<td>RES 9B</td>
</tr>
<tr>
<td>C83D</td>
<td>RES 9C</td>
</tr>
<tr>
<td>C83C</td>
<td>RES 9D</td>
</tr>
<tr>
<td>C83B</td>
<td>RES 9E</td>
</tr>
<tr>
<td>C83A</td>
<td>RES 9F</td>
</tr>
<tr>
<td>C82F</td>
<td>RES A0</td>
</tr>
<tr>
<td>C82E</td>
<td>RES A1</td>
</tr>
<tr>
<td>C82D</td>
<td>RES A2</td>
</tr>
<tr>
<td>C82C</td>
<td>RES A3</td>
</tr>
<tr>
<td>C82B</td>
<td>RES A4</td>
</tr>
<tr>
<td>C82A</td>
<td>RES A5</td>
</tr>
<tr>
<td>C81F</td>
<td>RES A6</td>
</tr>
<tr>
<td>C81E</td>
<td>RES A7</td>
</tr>
<tr>
<td>C81D</td>
<td>RES A8</td>
</tr>
<tr>
<td>C81C</td>
<td>RES A9</td>
</tr>
<tr>
<td>C81B</td>
<td>RES AA</td>
</tr>
<tr>
<td>C81A</td>
<td>RES AB</td>
</tr>
<tr>
<td>C80F</td>
<td>RES AC</td>
</tr>
<tr>
<td>C80E</td>
<td>RES AD</td>
</tr>
<tr>
<td>C80D</td>
<td>RES AE</td>
</tr>
<tr>
<td>C80C</td>
<td>RES AF</td>
</tr>
<tr>
<td>C80B</td>
<td>RES B0</td>
</tr>
<tr>
<td>C80A</td>
<td>RES B1</td>
</tr>
<tr>
<td>C7F</td>
<td>RES B2</td>
</tr>
<tr>
<td>C7E</td>
<td>RES B3</td>
</tr>
<tr>
<td>C7D</td>
<td>RES B4</td>
</tr>
<tr>
<td>C7C</td>
<td>RES B5</td>
</tr>
<tr>
<td>C7B</td>
<td>RES B6</td>
</tr>
<tr>
<td>C7A</td>
<td>RES B7</td>
</tr>
<tr>
<td>C6F</td>
<td>RES C0</td>
</tr>
<tr>
<td>C6E</td>
<td>RES C1</td>
</tr>
<tr>
<td>C6D</td>
<td>RES C2</td>
</tr>
<tr>
<td>C6C</td>
<td>RES C3</td>
</tr>
<tr>
<td>C6B</td>
<td>RES C4</td>
</tr>
<tr>
<td>C6A</td>
<td>RES C5</td>
</tr>
<tr>
<td>C5F</td>
<td>RES C6</td>
</tr>
<tr>
<td>C5E</td>
<td>RES C7</td>
</tr>
<tr>
<td>C5D</td>
<td>RES C8</td>
</tr>
<tr>
<td>C5C</td>
<td>RES C9</td>
</tr>
<tr>
<td>C5B</td>
<td>RES CA</td>
</tr>
<tr>
<td>C5A</td>
<td>RES CB</td>
</tr>
<tr>
<td>C4F</td>
<td>RES CC</td>
</tr>
<tr>
<td>C4E</td>
<td>RES CD</td>
</tr>
<tr>
<td>C4D</td>
<td>RES CE</td>
</tr>
<tr>
<td>C4C</td>
<td>RES CF</td>
</tr>
<tr>
<td>C4B</td>
<td>RES D0</td>
</tr>
<tr>
<td>C4A</td>
<td>RES D1</td>
</tr>
<tr>
<td>C3F</td>
<td>RES D2</td>
</tr>
<tr>
<td>C3E</td>
<td>RES D3</td>
</tr>
<tr>
<td>C3D</td>
<td>RES D4</td>
</tr>
<tr>
<td>C3C</td>
<td>RES D5</td>
</tr>
<tr>
<td>C3B</td>
<td>RES D6</td>
</tr>
<tr>
<td>C3A</td>
<td>RES D7</td>
</tr>
<tr>
<td>C2F</td>
<td>RES D8</td>
</tr>
<tr>
<td>C2E</td>
<td>RES D9</td>
</tr>
<tr>
<td>C2D</td>
<td>RES DA</td>
</tr>
<tr>
<td>C2C</td>
<td>RES DB</td>
</tr>
<tr>
<td>C2B</td>
<td>RES DC</td>
</tr>
<tr>
<td>C2A</td>
<td>RES DD</td>
</tr>
<tr>
<td>C1F</td>
<td>RES DE</td>
</tr>
<tr>
<td>C1E</td>
<td>RES DF</td>
</tr>
<tr>
<td>C1D</td>
<td>RES DG</td>
</tr>
<tr>
<td>C1C</td>
<td>RES DH</td>
</tr>
<tr>
<td>C1B</td>
<td>RES DJ</td>
</tr>
<tr>
<td>C1A</td>
<td>RES DK</td>
</tr>
<tr>
<td>C0F</td>
<td>RES DL</td>
</tr>
<tr>
<td>C0E</td>
<td>RES DM</td>
</tr>
<tr>
<td>C0D</td>
<td>RES DN</td>
</tr>
<tr>
<td>C0C</td>
<td>RES DO</td>
</tr>
<tr>
<td>C0B</td>
<td>RES DP</td>
</tr>
<tr>
<td>C0A</td>
<td>RES DQ</td>
</tr>
<tr>
<td>CO</td>
<td>RET 00</td>
</tr>
<tr>
<td>D8</td>
<td>RET 01</td>
</tr>
<tr>
<td>DF</td>
<td>RET 02</td>
</tr>
<tr>
<td>DO</td>
<td>RET 03</td>
</tr>
<tr>
<td>CO</td>
<td>RET 04</td>
</tr>
<tr>
<td>D0</td>
<td>RET 05</td>
</tr>
<tr>
<td>D0</td>
<td>RET 06</td>
</tr>
<tr>
<td>D0</td>
<td>RET 07</td>
</tr>
<tr>
<td>D0</td>
<td>RET 08</td>
</tr>
<tr>
<td>EF</td>
<td>RET 10</td>
</tr>
<tr>
<td>E0</td>
<td>RET 11</td>
</tr>
<tr>
<td>EF</td>
<td>RET 12</td>
</tr>
<tr>
<td>E0</td>
<td>RET 13</td>
</tr>
<tr>
<td>EF</td>
<td>RET 14</td>
</tr>
<tr>
<td>E0</td>
<td>RET 15</td>
</tr>
<tr>
<td>EF</td>
<td>RET 16</td>
</tr>
<tr>
<td>E0</td>
<td>RET 17</td>
</tr>
<tr>
<td>EF</td>
<td>RET 18</td>
</tr>
<tr>
<td>E0</td>
<td>RET 19</td>
</tr>
<tr>
<td>EF</td>
<td>RET 20</td>
</tr>
<tr>
<td>E0</td>
<td>RET 21</td>
</tr>
<tr>
<td>EF</td>
<td>RET 22</td>
</tr>
<tr>
<td>E0</td>
<td>RET 23</td>
</tr>
<tr>
<td>EF</td>
<td>RET 24</td>
</tr>
<tr>
<td>E0</td>
<td>RET 25</td>
</tr>
<tr>
<td>EF</td>
<td>RET 26</td>
</tr>
<tr>
<td>E0</td>
<td>RET 27</td>
</tr>
<tr>
<td>EF</td>
<td>RET 28</td>
</tr>
<tr>
<td>E0</td>
<td>RET 29</td>
</tr>
<tr>
<td>EF</td>
<td>RET 30</td>
</tr>
<tr>
<td>E0</td>
<td>RET 31</td>
</tr>
<tr>
<td>EF</td>
<td>RET 32</td>
</tr>
<tr>
<td>OBJ CODE</td>
<td>SOURCE</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
</tr>
<tr>
<td>CB28</td>
<td>SRA</td>
</tr>
<tr>
<td>CB2C</td>
<td>SRA</td>
</tr>
<tr>
<td>CB2D</td>
<td>SRA</td>
</tr>
<tr>
<td>CB3E</td>
<td>SRL</td>
</tr>
<tr>
<td>DDC8053E</td>
<td>SRL</td>
</tr>
<tr>
<td>FDC8053E</td>
<td>SRL</td>
</tr>
<tr>
<td>CB3F</td>
<td>SRL</td>
</tr>
<tr>
<td>CB38</td>
<td>SRL</td>
</tr>
<tr>
<td>CB39</td>
<td>SRL</td>
</tr>
<tr>
<td>CB3A</td>
<td>SRL</td>
</tr>
<tr>
<td>CB3B</td>
<td>SRL</td>
</tr>
<tr>
<td>CB3C</td>
<td>SRL</td>
</tr>
<tr>
<td>CB3D</td>
<td>SRL</td>
</tr>
<tr>
<td>96</td>
<td>SUB</td>
</tr>
<tr>
<td>DD9605</td>
<td>SUB</td>
</tr>
<tr>
<td>FD9605</td>
<td>SUB</td>
</tr>
<tr>
<td>97</td>
<td>SUB</td>
</tr>
<tr>
<td>90</td>
<td>SUB</td>
</tr>
<tr>
<td>91</td>
<td>SUB</td>
</tr>
<tr>
<td>92</td>
<td>SUB</td>
</tr>
<tr>
<td>93</td>
<td>SUB</td>
</tr>
<tr>
<td>94</td>
<td>SUB</td>
</tr>
<tr>
<td>95</td>
<td>SUB</td>
</tr>
<tr>
<td>DE20</td>
<td>SUB</td>
</tr>
<tr>
<td>AE</td>
<td>XOR</td>
</tr>
<tr>
<td>DDAE05</td>
<td>XOR</td>
</tr>
<tr>
<td>FDAE05</td>
<td>XOR</td>
</tr>
<tr>
<td>AF</td>
<td>XOR</td>
</tr>
<tr>
<td>AB</td>
<td>XOR</td>
</tr>
<tr>
<td>A9</td>
<td>XOR</td>
</tr>
<tr>
<td>AA</td>
<td>XOR</td>
</tr>
<tr>
<td>A8</td>
<td>XOR</td>
</tr>
<tr>
<td>AC</td>
<td>XOR</td>
</tr>
<tr>
<td>A0</td>
<td>XOR</td>
</tr>
<tr>
<td>EE20</td>
<td>XOR</td>
</tr>
</tbody>
</table>
Appendix F

Mini Computer Mnemonics

Mini Computer Instruction Mnemonics -- F1
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.
ADCCCL 102160  Add the complement of ACS to ACD; use complement of Carry as base for carry bit; rotate left.
ADCCCR 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; swap halves of result.
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; swap halves of result.
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.
ADDCCR 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.
ADDLL 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 ACO-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.
DOB P 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.
DOCT 064000 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.
INCCCL 101560  Place ACS + 1 in ACD; use complement of Carry as base for carry bit; rotate left.
INCCCR 101660  Place ACS + 1 in ACD; use complement of Carry as base for carry bit; rotate right.
INCCCS 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.
INCOR 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.
INCL. 101420  Place ACS + 1 in ACD; use 0 as base for carry bit.
INCL.L 101520  Place ACS + 1 in ACD; use 0 as base for carry bit; rotate left.
INCRZ 101620  Place ACS + 1 in ACD; use 0 as base for carry bit; rotate right.
INCR 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).
INREN 060177  Enable interrupt by setting Interrupt On (= NIOS CPU).
IORST 062677  Clear all 10 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.
MOVCL 101060  Move ACS to ACD; use complement of Carry as carry bit.
MOVC 101160  Move ACS to ACD; use complement of Carry as carry bit; rotate left.
MOVCLR 101260  Move ACS to ACD; use complement of Carry as carry bit; rotate right.
MOVC 101360  Move ACS to ACD; use complement of Carry as carry bit; swap halves of result.
MOV 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.
MOVZ 101300  Move ACS to ACD; use Carry as carry bit; swap halves of result.
MOVZL 101120  Move ACS to ACD; use 0 as carry bit.
MOVZL 101120  Move ACS to ACD; use 0 as carry bit; rotate left.
MOVZL 101120  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 (DOH, 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.
NEG 100400  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 Caroy 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.
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.
NIOP 060000  Clear device.
NIOP 060300  Send special pulse to device.
Start device.

Read console data switches into AC (= DIA -.CPU).

Skip if both carry and result are nonzero (skip function in an arithmetic or logical instruction).

Skip if either carry or result is zero (skip function in an arithmetic or logical instruction).

Skip (skip function in an arithmetic or logical instruction).

Skip if Busy is 1.

Skip if Busy is 0.

Skip if Done is 1.

Skip if Done is 0.

Skip if carry bit is 1 (skip function in an arithmetic or logical instruction).

Stores AC in location E.

Subtract ACS from ACD; use Carry as base for carry bit.

Subtract ACS from ACD; use complement of Carry as base for carry bit.

Subtract ACS from ACD; use complement of Carry as base for carry bit; rotate left.

Subtract ACS from ADC; use complement of Carry as base for carry bit.

Subtract ACS from ACD; use 0 as base for carry bit; rotate left.

Subtract ACS from ACD; use 0 as base for carry bit.

Subtract ACS from ACD; use 0 as base for carry bit; rotate right.

Subtract ACS from ACD; use 0 as base for carry bit; swap halves of result.

Skip if carry is 0 (skip function in an arithmetic or logical instruction).

Skip if result is zero (skip function in an arithmetic or logical instruction).
When this character appears in an instruction, the assembler places a 1 in bit 5 to produce indirect addressing.

When this character appears with a 15-bit address, the assembler places a 1 in bit 0, making the address indirect.

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 Programmer

This section was included only for interest sake

Software listing - - - - - - - - - - - - G1
Block diagram - - - - - - - - - - - - G16
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
START: EQU START-4 ;EPROM START
START: EQU START-2 ;RAM START

HBUFF: EQU 10FOH
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 030AH ;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 0F81H ;HL TO CURSOR POSI
BRKEY: EQU 001EH ;SHIFT/BREAK PRESSE
PRINT: EQU 0012H ;PRINT ASCII
CURSOR: EQU 1171H
HEX: EQU 03F9H ;ASCII TO NIBBLE
CONTROL: EQU 0DDCH ;OUTPUT CONTROLL
GETCHAR: EQU 0983H ;SCAN KEYBOARD
NORMAL: EQU 0BCEH ;CONVERT TO ASCII
SPECIAL: EQU 0B99H ;CONV TO SCRN CHA
BELL: EQU 003EH ;SOUND BELL

;START OF MAIN PGM
61
62 6000 CD0900 MAIN: CALL NL  ;***EPROM PROGRAMER***
63 6003 111362 LD DE,M6
64 6006 CD1500 CALL MSG
65 6009 CD9090 CALL NL
66 600C CD1E00 BREAK: CALL BKEY  ;SHIFT/BREAK?
67 600F CA0012 JP Z,1200H  ;GOTO ZEN
68 6012 CD8309 CALL GETCHAR
69 6015 FE10 CP 16  ;P
70 6017 2827 JR Z,PROGRAM  ;PROGRAM EPROM
71 6019 FE03 CP 3  ;C
72 601B CAD560 JP Z,CHECK  ;CHECK EPROM CLEAN
73 601E FE16 CP 22  ;V
74 6020 CAFA60 JP Z,VERIFY  ;CHECK PROGRAMED OK
75 6023 FE04 CP 4  ;D
76 6025 CA0C61 JP Z,DUPPLICATE  ;COPY EPROM'S
77 6028 CD0012 JP Z,READ  ;READ EPROM TO SCREEN
78 602A CA6561 JP Z,PRINT  ;SCREEN OR PRINTER
79 602D FE14 CP 20  ;T
80 602F CA3460 JR BREAK
81 6032 1BDB ;MUST READ BEFORE PRINT
82 83 6034 CD0900 PRINT: CALL NL  ;CR+LF
84 6037 11CE62 LD DE,M19
85 603A CD1500 CALL MSG
86 603D CD0165 JP PRINT
87 88 89 6040 CD0900 PROGRAM: CALL NL
90 6043 112962 LD DE,M7
91 6046 CD1500 CALL MSG
92 6049 CD9361 CALL STAEP
93 604C CD1763 CALL GETNUM
94 604F 22FC5F LD (START),HL  ;SAVE EPROM START
95 6052 CD8F61 CALL STARA
96 6055 CD1763 CALL GETNUM
97 6058 22FE5F LD (STARTR),HL  ;SAVE RAM START
98 605B CD0561 CALL STORA
99 605E CD1763 CALL GETNUM
100 6061 23 INC HL  ;STOP+1
101 6062 22FA5F LD (STOPR),HL  ;SAVE RAM STOP
102 6065 CDE863 CALL INIT  ;SET UP ALL OUTPUT
103 6068 CD0900 CALL NL
104 606B 11D562 LD DE,M20
105 606E CD1500 CALL MSG
106 6071 CD0900 CALL NL  ;LINE FEED
107 6074 CD8309 CALL GETCHAR
108 6077 FE01 CP 1  ;A
109 6079 CABB60 JP Z,PG16
110 607C FE02 CP 2  ;B
111 607E CAB160 JP Z,PG32
112 6081 117F62 LD DE,M14  ;ERROR
113 6084 CD1500 CALL MSG
114 6087 CD0900 CALL NL
115 608A C30160 JP MAIN
116 117 118
119 608D CD0900 PG16: CALL NL
120 6090 11F562 LD DE,M21
121 6093 CD1500  CALL MSG
122 6096 116762  LD DE,M17
123 6099 CD1500  CALL MSG
124 609C CD8309  CALL GETCHAR
125 609D CD0900  CALL NL
126 60A2 115B62  LD DE,M13
127 60A5 CD1500  CALL MSG
128 60A8 CD8309  CALL GETCHAR  ;WAIT FOR ANY KEY
129 60AB CD4764  CALL PGMR
130 60AE C30060  JP MAIN

131

132

133 60B1 CD0900  PG32:  CALL NL
134 60B4 110663  LD DE,M22
135 60B7 CD1500  CALL MSG
136 60BA 116762  LD DE,M17
137 60BD CD1500  CALL MSG
138 60C0 CD8309  CALL GETCHAR
139 60C3 CD0900  CALL NL
140 60C6 115B62  LD DE,M13
141 60C9 CD1500  CALL MSG
142 60CC CD8309  CALL GETCHAR
143 60CF CD7C64  CALL PGM32
144 60D2 C30060  JP MAIN

145

146

147

148

149 60D5 CD0900  CHECK:  CALL NL
150 60D8 113162  LD DE,M8
151 60DB CD1500  CALL MSG
152 60DE CD9361  CALL STAE
153 60E1 CD1763  CALL GETNUM
154 60E4 22FC5F  LD (STARTE),HL ;SAVE EPROM START
155 60E7 CD0900  CALL STAE
156 60EA CD1763  CALL GETNUM
157 60ED 23  INC HL  ;STOP+1
158 60EE 22FB5F  LD (STOPE),HL ;SAVE EPROM STOP
159 60F1 CD0C64  CALL INIT2 ;SET UP FOR READ
160 60F4 CD8B65  CALL CHK  ;DO CHECK FOR CLEAN ROM
161 60F7 C30060  JP MAIN

162

163

164  ;MUST HAVE PROGRAMED
165 60FA CD0900  VERIFY:  CALL NL
166 60FD 113762  LD DE,M9
167 6100 CD1500  CALL MSG
168 6103 CD0C64  CALL INIT2
169 6106 CD8865  CALL VERF
170 6109 C30060  JP MAIN

171

172

173 610C CD0900  DUPLICATE:  CALL NL
174 610F 113E62  LD DE,M10
175 6112 CD1500  CALL MSG
176 6115 CD9361  CALL STAE
177 6118 CD1763  CALL GETNUM
178 611B 22FC5F  LD (STARTE),HL ;EPROM START
179 611E CD9661  CALL STAE
180 6121 CD1763  CALL GETNUM

G3
181 6124 23  
182 6125 22F85F  
183 6126 CD8F61  
184 6128 CD1763  
185 612E 22FE5F  
186 6131 D5  
187 6132 E5  
188 6133 37  
189 6134 3F  
190 6135 2AF85F  
191 6136 EDSBFC5F  
192 613C ED52  
193 613E EDSBFE5F  
194 6142 ED5A  
195 6144 22F85F  
196 6147 E1  
197 6148 D1  
198 6149 CD0C64  
199 6154 CD0965  
200 6152 D5  
201 6153 C09000  
202 6156 CD1500  
203 6159 C01763  
204 615C C00900  
205 615D C01500  
206 615F CD4764  
207 6162 C30060  
208 6169 C08F61  
209 616E 22F85F  
210 616F C01763  
211 6170 23  
212 6174 22FC5F  
213 6177 CD0961  
214 617A CD1763  
215 617E 22F85F  
216 6180 C09000  
217 6181 C00900  
218 6188 C01500  
219 618B CD4764  
220 618D C01763  
221 6190 C30060  
222 6193 C09000  
223 6196 CD0961  
224 619C CD1500  
225 619D CD0961  
226 619F CD1500  
227 61A0 C9  
228 61A2 CD0961  
229 61A9 CD0900  
230 61A9 CD0900  
231 61AF CD1500  
232 61B0 C01763  
233 61B1 22FE5F  
234 61B4 CD1500  
235 61B8 CD1500  
236 61B9 C9  
237 61BE CD0961  
238 61C0 CD4764  
239 61C4 CD1763  
240 61C5 CD1500

; EPROM STOP
; RAM START

; READ INTO RAM
; SCAN KEYBOARD
; STA ADD EPROM
; STOP ADD EPROM

; STOEP:
; STARP:
; STARA:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
; STARE:
241 61B5 CD1500 CALL MSG
242 61B8 110162 LD DE, M4
243 61BB CD1500 CALL MSG
244 61BE C9 RET
245 61BF CD0900 CALL MSG
246 61E2 11EB61 LD DE, M1
247 61E5 CD1500 CALL MSG
248 61E8 11FB61 LD DE, M3
249 61EB CD1500 CALL MSG
250 61E4 110B62 LD DE, M5
251 61E1 CD1500 CALL MSG
252 61E4 C9 RET
253 61E5 CD0900 CALL MSG
254 61E8 11FB61 LD DE, M3
255 61E1 CD1500 CALL MSG
256 61E4 110B62 LD DE, M5
257 61E7 CD1500 CALL MSG
258 61E4 110B62 LD DE, M5
259 61E7 CD1500 CALL MSG
260 61E4 C9 RET
261 61E5 53544152 M1: DB 'START ', CR
262 61E5 542000 DB 'STOP ', CR
263 61E5 41444552 M3: DB 'ADDRESS ', CR
264 61E5 53544F50 M2: DB '(EPROM): ', CR
265 61E5 2852414D M5: DB '(RAM): ', CR
266 61E5 29203A0D DB '***EPROM'
267 61E5 50524F47 M4: DB 'PROGRAMER***', CR
268 61E5 52414D45 M5: DB 'READ', CR
269 61E5 46590D DB 'ERROR.NOT HEX', CR
270 61E5 5800 CR
271 61E5 434F4E4E M13: DB 'CONNECT 25 V'
272 61E5 4E59204E M17: DB ' PRESS ANY KEY '
GETNUM: LD HL,0000H ;CLEAR HL TO 0
296 631A CD1E00 CALL BRKEY ;SHIFT/BREAK ?
297 631D CA0060 JP Z,MAIN
298 6320 CDB309 CALL GETCHAR ;GET MSN
299 6323 CDCE0B CALL NORMAL ;CONVERT TO ASCII
300 6326 47 LD B,A
301 6327 CD1200 CALL PRTN
302 632A 78 LD A,B
303 632B CDF903 CALL HEX ;CONVERT TO HEX
CALL C,ERROR
AND OFH ;MASK
RLC A
RLC A
RLC A
RLC A
LD H,A
CALL BRKEY ;SHIFT/BREAK ?
JP Z,MAIN
CALL GETCHAR ;GET NMSN
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR ;SLSN
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCHAR
CALL NORMAL
LD B,A
CALL PRNT
LD A,B
CALL GETCH
364 ;
365 ;
366 ;8255 ROUTINES
367 ;
368 63A8 E5  V250N:  PUSH HL
369 63A9 210EE0  LD HL,EPROMC
370 63AC CB6E  SET 5,(HL)  ;25 VOLT ON
371 63AE E1  POP HL
372 63AF C9  RET
373 ;
374 ;
375 63B0 E5  V25OFF:  PUSH HL
376 63B1 210EE0  LD HL,EPROMC
377 63B4 CB6E  RES 5,(HL)  ;25 VOLT OFF
378 63B6 E1  POP HL
379 63B7 C9  RET
380 ;
381 ;
382 63B8 E5  CE1OFF:  PUSH HL
383 63B9 210EE0  LD HL,EPROMC
384 63BC CB6E  SET 7,(HL)  ;CE1 OFF (HI)
385 63BE E1  POP HL
386 63BF C9  RET
387 ;
388 ;
389 63C0 E5  CE1ON:  PUSH HL  ;CE1 ON (LO)
390 63C1 210EE0  LD HL,EPROMC
391 63C4 CB6E  RES 7,(HL)
392 63C6 E1  POP HL
393 63C7 C9  RET
394 ;
395 ;
396 63C8 E5  CE0OFF:  PUSH HL  ;CE0 OFF (HI)
397 63C9 210EE0  LD HL,EPROMC
398 63CC CB6E  SET 6,(HL)
399 63CE E1  POP HL
400 63CF C9  RET
401 ;
402 ;
403 63D0 E5  CE0ON:  PUSH HL  ;CE0 ON (LO)
404 63D1 210EE0  LD HL,EPROMC
405 63D4 CB66  RES 6,(HL)
406 63D6 E1  POP HL
407 63D7 C9  RET
408 ;
409 ;
410 63D8 E5  OEOFF:  PUSH HL  ;OE OFF (HI)
411 63D9 210EE0  LD HL,EPROMC
412 63DC CB66  SET 4,(HL)
413 63DE E1  POP HL
414 63DF C9  RET
415 ;
416 ;
417 63E0 E5  OEON:  PUSH HL  ;OE ON (LO)
418 63E1 210EE0  LD HL,EPROMC
419 63E4 CB66  RES 4,(HL)
420 63E6 E1  POP HL
421 63E7 C9  RET
422 ;
423 ;
; INITIALIZE 8255
424 63E8 E5  INIT:  PUSH HL
425 63E9 F5  PUSH AF
426 63EA 210FE0  LD HL,EPROM
427 63ED 3E80  LD A,80H
428 63EF 77  LD (HL),A ; ALL IO TO OUTPUT
429 63F0 2B  DEC HL
430 63F1 CBF6  SET 6,(HL) ; CE0 OFF (HI)
431 63F3 CBFE  SET 7,(HL) ; CE1 OFF (HI)
432 63F5 CBE6  SET 4,(HL) ; OE OFF (HI)
433 63F7 CBAE  RES 5,(HL) ; 25 VOLT OFF (LO)
434 63F9 CB86  RES 0,(HL)
435 63FB CB96  RES 1,(HL)
436 63FD CB9E  RES 2,(HL)
437 63FF CB9E  RES 3,(HL)
438 6401 3E00  LD A,00
439 6403 320DE0  LD (EPROMB),A
440 6406 320CE0  LD (EPROMA),A
441 6409 F1  POP AF
442 640A E1  POP HL
443 640B C9  RET
444
445
446
447
448 640C E5  INIT2:  PUSH HL
449 640D F5  PUSH AF
450 640E 210FE0  LD HL,EPROM
451 6411 3E90  LD A,90H
452 6413 77  LD (HL),A ; PORT A TO INPUT
453 6414 2B  DEC HL
454 6415 CBF6  SET 6,(HL) ; CE0 OFF (HI)
455 6417 CBFE  SET 7,(HL) ; CE1 OFF (HI)
456 6419 CBE6  SET 4,(HL) ; OE OFF (HI)
457 641B CBAE  RES 5,(HL) ; 25 VOLT OFF
458 641D CB86  RES 0,(HL)
459 641F CB8E  RES 1,(HL)
460 6421 CB96  RES 2,(HL)
461 6423 CB9E  RES 3,(HL)
462 6425 3E00  LD A,00
463 6427 320DE0  LD (EPROMB),A
464 642A F1  POP AF
465 642B E1  POP HL
466 642C C9  RET
467
468
469
470
471 ; COMPARE DE TO BC
472 ; Z=0 IF EQUAL
473
474 642D D5  COMPARE:  PUSH DE
475 642E C5  PUSH BC
476 642F 7A  LD A,D ; COMPARE D+B
477 6430 B8  CP B
478 6431 C23664  JP NZ,HOME
479 6434 7B  LD A,E ; COMPARE E+C
480 6435 B9  CP C
481 6436 C1  HOME:  POP BC
482 6437 D1  POP DE
483 6438 C9  RET
PAGE 10

484  ;
485  ;
486  ;WAIT LOOP
487  ;
488  6439 E5  WAIT:  PUSH HL
489  643A 2119E4  LD HL,TIME
490  643D 2C  LOOP:  INC L
491  643E C23D64  JP NZ,LOOP
492  6441 24  INC H
493  6442 C23D64  JP NZ,LOOP
494  6445 E1  POP HL
495  6446 C9  RET
496  ;
497  ;PROGRAM LOOP
498  ;
499  500  6447 ED5BFESF  PGMR:  LD DE,(STARTR)
501  644B ED48FA3F  LD BC,(STOPR)
502  644F 2AF35F  LD HL,(STARTE)
503  6452 CD5E64  LOOP2:  CALL PGM
504  6455 23  INC HL
505  6456 13  INC DE
506  6457 CD2D64  CALL COMPARE
507  645A C25264  JP NZ,LOOP2
508  645D C9  RET ;RETURN TO MAIN PGM
509  ;
510  ;
511  ;PROGRAM A LOCATION
512  ;
513  645E 7D  PGM:  LD A,L
514  645F 320DE0  LD (EPROMB),A ;ADDRESS LO
515  6462 7C  LD A,H
516  6463 CBE7  SET 4,A
517  6465 CBBF  RES 7,A
518  6467 CBEE  SET 5,A ;25V ON
519  6469 CBFF  SET 6,A ;CEO HI
520  646B 320EE0  LD (EPROMA),A
521  646E 1A  LD A,(DE)
522  646F 320CEO  LD (EPROMA),A ;DATA
523  6472 CD8663  CALL CE1OFF ;CE1=HI
524  6475 CD3964  CALL WAIT ;WAIT 50MILISEC
525  6478 CDC063  CALL CE1ON ;CE1=LO
526  647B C9  RET
527  ;
528  ;
529  647C ED5BFESF  PGM32:  LD DE,(STARTR)
530  6480 ED4BF5AF  LD BC,(STOPR)
531  6484 2AF55F  LD HL,(STARTE)
532  6487 CD9364  PLOOP2:  CALL P32
533  648A 23  INC HL
534  648B 13  INC DE
535  648C CD2D64  CALL COMPARE
536  648F 2B7664  JP NZ,PLOOP2
537  6492 C9  RET
538  ;
539  ;
540  ;PROGRAM A LOCATION
541  ;
542  6493 7D  P32:  LD A,L
543  6494 320DE0  LD (EPROMB),A ;ADDRESS LO
544 6497 7C  LD A,H
545 6498 CBE7  SET 4,A
546 649A CBFF  SET 7,A  ;CE1 HI
547 649C CBEF  SET 5,A  ;25V ON
548 649E CBFF  SET 6,A  ;CE0 HI
549 64A0 320EE0  LD (EPROMC),A  ;DATA
550 64A3 1A  LD A,(DE)
551 64A4 320CE0  LD (EPROMA),A
552 64A7 CDC063  CALL CE10N  ;CE1 LO
553 64A8 CD3964  CALL WAIT  ;WAIT 50 MILLISEC
554 64AD CDB863  CALL CE1OFF  ;CE1 HI
555 64B0 C9  RET
556
557
558
559 ;CHECK FOR CLEAN ROM
560
561 64B1 ED5BFC5F  CHK:  LD DE,(STARTE)
562 64B5 ED4BF85F  LD BC,(STOPE)
563 64B9 CDC664  LOOP3:  CALL CK  ;DO THE CHECK
564 64BC 13  INC DE
565 64BD CD2064  CALL COMPARE  ;CHECK IF OVER
566 64C0 C2BB64  JP NZ,LOOP3
567 64C3 C30060  JP MAIN
568
569
570 64C6 7B  CK:  LD A,E
571 64C7 320DE0  LD (EPROMB),A  ;ADDRESS LO
572 64CA 7A  LD A,0
573 64CB CBA7  RES 4,A  ;OE=LO
574 64CD CBFF  SET 7,A  ;CE1=HI
575 64CF CBAF  RES 5,A  ;25V OFF
576 64D0 CBF7  SET 6,A  ;CE0=HI
577 64D3 320EE0  LD (EPROMC),A
578 64D6 CDC063  CALL CE10N
579 64D9 3A0E0  LD A,(EPROMA)  ;GET DATA
580 64DC CD8863  CALL CE1OFF
581 64DF FEFF  CP OFFH
582 64E1 C4E664  CALL NZ,ERROR2
583 64E4 C9  RET
584
585
586 64E5 ED53F45F  ERROR2:  LD (ADOR),DE
587 64E9 32F65F  LD (DAT),A
588 64EC CD0900  CALL NL
589 64EF D5  PUSH DE
590 64F0 117F62  LD DE,M14
591 64F3 CD1500  CALL MSG
592 64F6 D1  POP DE
593 64F7 2AF45F  LD HL,(ADDR)  ;GET ADDRESS
594 64FA CD8A03  CALL PRTHL
595 64FD CD0C00  CALL PRNLS
596 6500 3AF65F  LD A,(DAT)  ;GET DATA
597 6503 CDC303  CALL PRTHX
598 6506 C9  RET
599
600
601
602 6507 CD0900  READE:  CALL NL  ;CR+LF
603 650A 119F62  LD DE,M16
604 6500 CD1500 CALL MSG
605 6510 CDB309 CALL GETCHAR
606 6513 FE20 CP 32 0
607 6515 CA5965 JP Z,REDO ;ROM0
608 6518 FE21 CP 33 1
609 651A CA2765 JP Z,RED1 ;ROM1
610 651D 117F62 LD DE,M14 ;ERROR
611 6520 CD1500 CALL MSG
612 6523 CD3E00 CALL BELL
613 6526 C9 RET
614
615
616 6527 ED5BF5F RED1: LD DE,(STARTE) ;GET DATA
617 652B ED4BF5F LD BC,(STOP) ;GET DATA
618 652F 2A9E5F LD HL,(STARTR)
619 6532 CD3E65 LOOPR: CALL RECI
620 6535 23 INC HL
621 6536 13 INC DE
622 6537 CD2664 CALL COMPARE
623 653A C23265 JP NZ,LOOPR
624 653D C9 RET
625
626 653E 78 REC1: LD A,E
627 653F 320DE0 LD (EPROMB),A
628 6542 7A LD A,D
629 6543 C8A7 RES 4,A
630 6545 CBFF SET 7,A
631 6547 CBAF RES 5,A
632 6549 CBF7 SET 6,A
633 654B 320EE0 LD (EPROMC),A
634 654E CDC063 CALL CE1ON
635 6551 3A0CE0 LD A,(EPROMA)
636 6554 CD8863 CALL CE1OFF
637 6557 77 LD (HL),A
638 6558 C9 RET
639
640
641
642
643 6559 ED5BF5F RED0: LD DE,(STARTE) ;GET DATA
644 655D ED4BF5F LD BC,(STOP) ;GET DATA
645 6561 2A9E5F LD HL,(STARTR)
646 6564 CD7065 LOOPQ: CALL RECO ;READ ROM
647 6567 23 INC HL
648 6568 13 INC DE
649 6569 CD2064 CALL COMPARE
650 656C C26465 JP NZ,LOOPQ
651 656F C9 RET
652
653
654 6570 78 RECO: LD A,E
655 6571 320DE0 LD (EPROMB),A
656 6574 7A LD A,D
657 6575 C8A7 RES 4,A
658 6577 CBFF SET 7,A
659 6579 CBAF RES 5,A
660 657B CBF7 SET 6,A
661 657D 320EE0 LD (EPROMC),A
662 6580 CDC063 CALL CE1ON
663 6583 3A0CE0 LD A,(EPROMA)
664 6586 CDC863 CALL CE00FF
665 6589 77 LD (HL),A
666 658A C9 RET
667
668
669
670
671 658B ED5BF65F VERF: LD DE,(STARTR) ;SET UP ADDRESS
672 658F ED4BF62F LD BC,(STOPR)
673 6593 2AF65F LD HL,(STARTE)
674 6596 D5 LOOPV: PUSH DE
675 6597 54 LD D,H
676 6598 5D LD E,L ;LD DE,HL
677 6599 CD1E65 CALL REC1 ;READ FROM EPROM
678 659C D1 POP DE
679 659D 32F25F LD (COUNT),A ;TEMP STORAGE
680 65A0 1A LD A,(DE) ;GET DATA FROM RAM
681 65A1 E5 PUSH HL
682 65A2 21F25F LD HL,COUNT ;COMPARE TO COUNT
683 65A5 BE CP (HL)
684 65A6 E1 POP HL
685 65A7 C4B365 CALL NZ,ERP ;ERROR
686 65AA 13 INC DE
687 65AB 23 INC HL
688 65AC CD2D64 CALL COMPARE
689 65AF C8 RET Z ;ALL DONE
690 65B0 C39665 JP LOOPV ;LOOP BACK
691
692 65B3 D5 ERP: PUSH DE
693 65B4 CD0900 CALL NL
694 65B7 117F62 LD DE,M14
695 65BA CD1500 CALL MSG
696 65BD CD3E00 CALL BELL ;SOUND BELL
697 65C0 D1 POP DE
698 65C1 E5 PUSH HL
699 65C2 CD0C00 CALL PRNTS
700 65C5 CDAB03 CALL PRTHL
701 65C8 CD0C00 CALL PRNTS
702 65CB 1A LD A,(DE)
703 65CC CDC303 CALL PRTHX
704 65CF E1 POP HL
705 65D0 C9 RET
706
707
708
709 65D1 CD0900 PRONT: CALL NL ;CR+LF
710 65D4 118662 LD DE,M15 ;'VIDEO OR PRINTER`
711 65D7 CD1500 CALL MSG
712 65DA CD3B09 CALL GETCHAR ;SCAN KEYBOARD
713 65DD FE16 CP 22 ;V
714 65DF CAF665 JP Z,VIDEO
715 65E2 FE05 CP 5 ;E
716 65E4 CA3B66 JP Z,PRINTER
717 65E7 117F62 LD DE,M14 ;ERROR
718 65E9 CD1500 CALL MSG
719 65ED CD3E00 CALL BELL
720 65F0 CD0900 CALL NL
721 65F3 C30060 JP MAIN
722
723
724  65F6  E05BF5F  VIDEO:  LD  DE,(STARTE)
725  65FA  ED4BF5F  LD  BC,(STOPE)
726  65FE  2AF5F  LD  HL,(STARTR)
727  6601  3EF8  LOOPH:  LD  A,OF8H
728  6603  32F25F  LD  (COUNT),A  ;COUNTER
729  6606  CD1E00  CALL  BRKEY  ;CHECK FOR SHIFT/BREAK
730  6609  CA0060  JP  Z,MAIN
731  660C  CD0900  CALL  NL
732  660E  E5  PUSH  HL
733  6610  62  LD  H,D
734  6611  68  LD  L,E
735  6612  CDBA03  CALL  PRTHL
736  6615  E1  POP  HL
737  6616  CD0C00  CALL  PRNTH;SPACE
738  6619  7E  DATA:  LD  A,(HL)
739  661A  CDC303  CALL  PRTHX
740  661D  CD0C00  CALL  PRNTS  ;SPACE
741  6620  13  INC  DE
742  6621  23  INC  HL
743  6622  CD2D64  CALL  COMPARE
744  6625  CA0060  JP  Z,MAIN
745  6628  3AF25F  LD  A,(COUNT)
746  662B  3C  INC  A
747  662C  32F25F  LD  (COUNT),A
748  662F  C21966  JP  NZ,DATA
749  6632  CD0900  CALL  NL
750  6635  C30166  JP  LOOPH
751  
752  
753  
754  6638  E05BF5F  PRINTER:  LD  DE,(STARTE)
755  663C  ED4BF5F  LD  BC,(STOPE)
756  6640  2AF5F  LD  HL,(STARTR)
757  6643  D5  PUSH  DE
758  6644  CD0900  CALL  NL
759  6647  118C62  LD  DE,M18
760  664A  CD1500  CALL  MSG
761  664D  116762  LD  DE,M17
762  6650  CD1500  CALL  MSG
763  6653  CDB309  CALL  GETCHAR
764  6656  D1  POP  DE
765  6657  3EF0  LOOPX:  LD  A,OF0H
766  6659  32F25F  LD  (COUNT),A  ;CHECK FOR SHIFT/BREAK
767  665C  CD1E00  CALL  BRKEY
768  665F  CA0060  JP  Z,MAIN
769  6662  3E20  LD  A,,' '
770  6664  CDC166  CALL  ACPRNNT
771  6667  7A  LD  A,D
772  6668  CDB866  CALL  ROTATE
773  666B  CDAF66  CALL  CONPRNT
774  666E  7A  LD  A,D
775  666F  CDAF66  CALL  CONPRNT
776  6672  7B  LD  A,E
777  6673  CDB866  CALL  ROTATE
778  6676  CDAF66  CALL  CONPRNT
779  6679  7B  LD  A,E
780  667A  CDAF66  CALL  CONPRNT
781  667D  3E20  LOOPY:  LD  A,,' '
782  667F  CDC166  CALL  ACPRNNT
783  6682  7E  LD  A,(HL)
CALL ROTATE
CALL CONPRINT
LD A,(HL)
CALL CONPRINT
INC DE
INC HL
CALL COMPARE
JP Z,CRLF
LD (COUNT),A
JP NZ,LOOPY
LD A,Cr
CALL ACPRNT
JP LOOPX
LD A,(COUNT)
INC A
LO (COUNTl,A
JP -LOOPV
LD A,CR
CALL ACPRNT
JP MAIN
AND OFH ;MASK IN LOWER NIBB
CALL ASC
CALL ACPRNT
RET
PUSH HL
PUSH BC
LD HL,OE014H
BIT 1,B
JP NZ,LOOPS
LD (HL),A
POP BC
POP HL
RET
END
EPROM Programer Circuit

To be used with a Sharp MZ80K PC