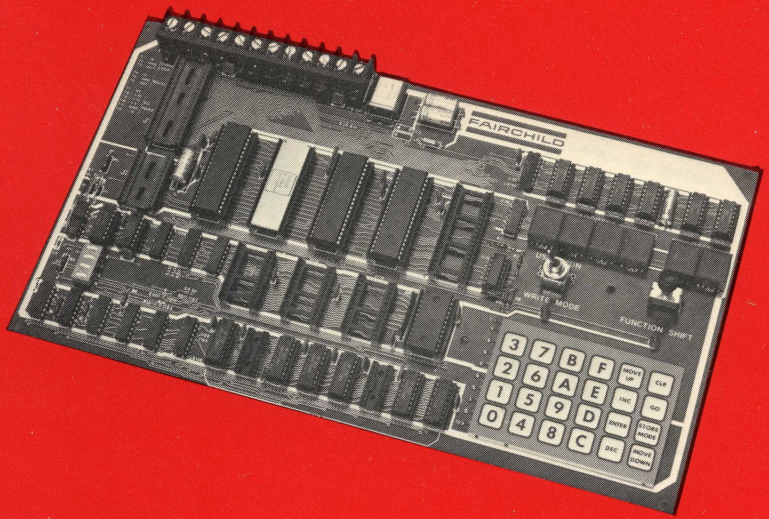
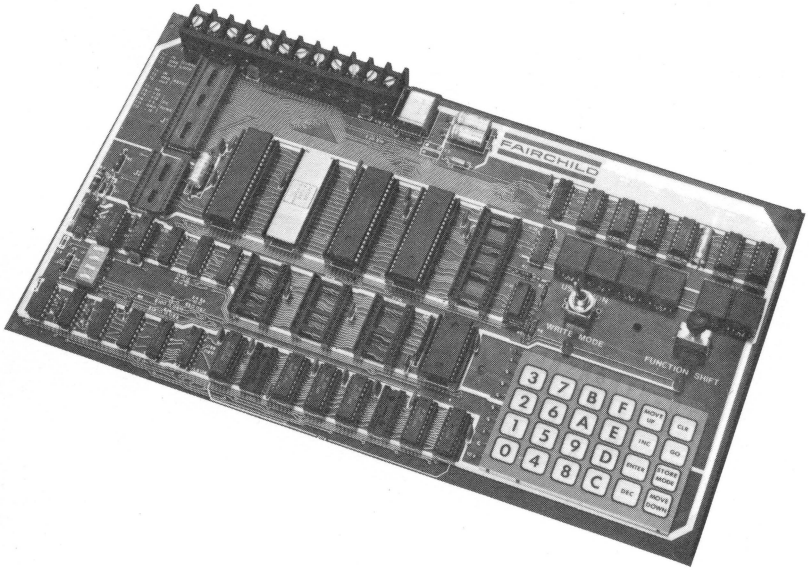
PEP USER GUIDE



FAIRCHILD

PEP USER GUIDE



FAIRCHILD

©1979 Fairchild Camera & Instrument Corporation • 464 Ellis Street • Mountain View, California 94042 • 415/962-5011 • TWX 910-379-6435

F387X PEP System User Guide
TABLE OF CONTENTS

Section 1
INTRODUCTION

1.1 Overview of the PEP System

The F387X Prototyping, Emulating, and Programming (PEP) System is a specialized, single-board microcomputer specifically designed as a tool to aid the engineer in the design and development of circuits using single-chip microcomputers. It offers the design engineer convenience and flexibility for both prototyping and emulating the industry-standard F3870, F38E70, F3872, F3876, and F3878 single-chip devices. The PEP System can also be used as a general F8 series prototyping tool.

The PEP System can serve as an emulator and prototyping tool for the design engineer, as an EPROM programmer, as a training system for the F387X family of single-chip microcomputers, or as a general-purpose microcomputer educational system. With the PEPBUG software supplied with the system, users can interactively examine and modify the registers, memory locations, and the I/O data.

Since the PEP System includes an on-board keypad and a six-digit display (a more expensive terminal is not required), the addition of only two power supplies makes the PEP System a complete stand-alone training station. A serial I/O interface is also provided to connect a video display terminal, teletypewriter, or other terminal if desired.

With the PEP System, users can program both F38E70 EPROM microcomputers and 2716-type EPROMs. To develop a program on the PEP System that emulates the desired micro-chip, the user first enters the program in RAM. He debugs his program interactively by examining memory and the system status as needed. When the program is correct, it is burned into a F38E70 or a 2716 EPROM for testing. When the prototype EPROM correctly emulates the programming of the desired micro-chip, the designer can use it as a convenient medium for transmitting single-chip ROM code to Fairchild at order time.

The features of the PEP System include:

- Full in-circuit emulation of F3870, F3872, F3876, and F3878 microcomputers using the Fairchild F8 System.
- 2K firmware monitor (PEPBUG).
- Versatile on-card keyboard for command and data entry.
- On-card address and data display using large 0.8" seven-segment digits.
- Programming socket for F38E70 single-chip EPROM microcomputers.
- Programming socket for 2716-type EPROMs.
- Crystal-controlled system clock.
- 2K bytes of executable static RAM (2114's).
- Sockets for an additional 2K bytes of static RAM (2114's).
- Sockets for 6K of executable EPROM (2716-type)
- Separate 128-byte static memory for monitor workspace.
- Flexible memory map strapping options.
- Serial I/O circuits for 20 mA current loop or EIA RS-232-C interfaces.
- LED indicators for STORE and PROMPT modes of keyboard operation.

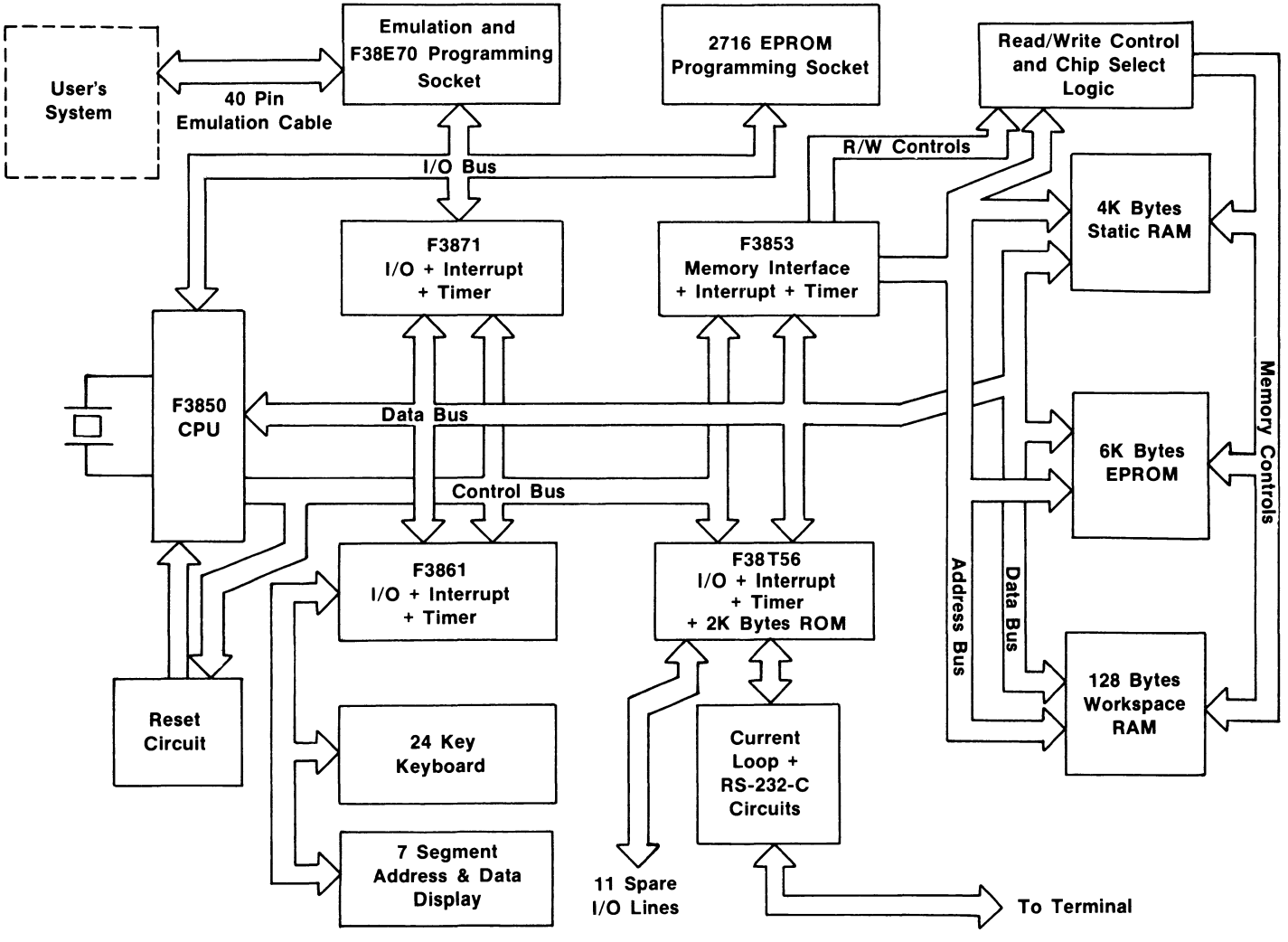


Figure 1-1 The Fairchild F387X PEP System

1.2 PEP System Configuration Supplied by Fairchild

The PEP System configuration is illustrated in the Component Diagram, Appendix A.

As shipped from the factory, the PEP System includes a F3850 Central Processing Unit (CPU) in socket U1, a F3871 Peripheral I/O Device (PIO) in socket U2, a F3861C Peripheral I/O Device in socket U3, and a F3853 Static Memory Interface (SMI) in socket U5. See Section 4.1.1 for more information about these chips.

PEPBUG, the PEP System monitor, may be supplied in one of two configurations. PEPBUG may be supplied in an F38T56 with 2K ROM and I/O capabilities. In this case, the F38T56 is in socket U4 and socket U16 is empty.

The alternative configuration is with PEPBUG in an F68316 at socket U16 (the F68316 is pin-compatible with a 2716, which could otherwise be placed in socket U16). In this case, socket U4 contains an F3861D to supply the I/O capabilities not available on the F68316. In this configuration, Fairchild ships the system with E10 connected to E12 and the E11-to-E12 connection open (see *Table 2-2* in Section 2 for PEP System connection options). This changes the U16 starting address from H'A000' to H'8000' (refer to *Table 2-4* in Section 2).

Random access memory is available at sockets U6 through U13. Four 2114 chips are supplied with the system, and there are sockets for another four 2114's. Sockets for 2716 EPROMs are available at U14 and U15; chips are not supplied in these sockets. An F6810 is supplied at socket U17; this provides the program workspace memory for the system monitor.

1.3 PEP System Philosophy

From a hardware standpoint, the primary use of the PEP System is to perform F387X emulation. Two ports from the F3850 CPU and another two ports from the F3871 Peripheral I/O device simulate the I/O capabilities of the F3870 series. Either RAM or EPROM, as needed, simulates the memory function.

The discussion of the F8 system in Section 4.1 summarizes the hardware elements for the F387X family of single-chip microcomputers. Note that, throughout this manual, references to I/O ports imply I/O addresses rather than physical lines.

From a programming standpoint, the major capability of the PEP System is that it is designed for the modular, iterative programming that is necessary in a program development environment. The PEPBUG monitor provides the ability to address specific sections of EPROM. Thus, the programmer can burn program segments into EPROM as they are developed, test them, and later add more program modules.

A variety of PEP System memory mapping options makes it convenient to evolve programs and program segments in RAM, then burn them into EPROM for testing. EPROMs can also be used for system storage, particularly if the PEP System is configured without an external terminal. In this case, programs can be stored in EPROM and debugged by downloading to RAM and making necessary corrections.

PEPBUG, the PEP System monitor, provides a powerful interactive debugging facility for use in F387X program development. An overview of PEPBUG is presented in Section 3; for detailed information on PEPBUG, refer to the *PEPBUG User's Guide*.

Section 2
SET-UP PROCEDURES

2.1 Starting System Operation
Before using the PEP System, apply DC power to the terminal strip on the upper left-hand side of the board (refer to the Physical Component Location Diagram in the Appendix).

Stand-alone operation of the system requires only two DC voltages: +5 volts and +12 volts. Make the following connections to attach the power supplies (before turning the power supplies on):

1. Connect the ground terminals of both power supplies at T11 of the PEP System terminal strip. Table 2-1 summarizes the connections that can be made to the terminal strip.

2. Connect the +5 volt terminal to T10; current required is approximately 800 mA.

3. Connect the +12 volt terminal to T9; current required is approximately 500 mA.

After the power supplies are connected, make sure that the DIP switches, SW 3 on the lower left-hand side of the board, are set to the following positions:

Switch	Position
A-B	A
C-D	C
E-F	E
H-J	H

Also set the three-position toggle switch at the right-hand side of the board, SW 2, to the center or KEYPAD position.

Now turn the power supplies on; they may be turned on in any sequence. Press the RESET button, SW 1. The display will light up, showing 0000 in the address display and random numbers in the data display. The PROMPT light will come on, indicating that the PEP System is ready to accept commands.

Table 2-1 Terminal Connector Definitions

Terminal	Definition
T1 (Left-most)	Current Loop In
T2	Current Loop Ground
T3	Current Loop Out
T4	EIA RS-232-C In
T5	EIA RS-232-C Ground
T6	EIA RS-232-C Out
T7	(N. C.)
T8	+25 Volt Programming Supply
T9	+12 Volt Supply
T10	+5 Volt Supply
T11	Ground
T12 (Right-most)	-12 Volt Supply

To check that the PEP System is functioning, press the MEM key. The PROMPT light will go off. Then enter 2BC3 and press the C/R key. The PROMPT light will go on. Press the CHG key. The STORE light will go on and the PROMPT light off. Enter the number 6, then press the C/R key twice. The STORE light will go off and the PROMPT light on. The address display should show 2BC3 and the data display should show 06. The PEP System is now set to communicate with an external terminal at 110 baud, if desired (this set-up procedure does not interfere with stand-alone operation using the keypad).

2.2 Stand-Alone Operation Using the Keypad

As described in the previous pages, the PEP System can be used as a stand-alone system, with the on-board keypad and the seven-segment address and data displays used for input and display.

To use the PEP System with the keypad, set the three-position toggle switch, SW 2, to the KEYPAD position, and press the RESET button, SW 1. The PEPBUG system monitor now controls the operation of the keypad.

When you press the RESET button, the PROMPT light comes on. This indicates that PEPBUG is ready to accept commands. The PEPBUG commands are shown on the 24 keys of the keypad; their use is described in Section 3.

In general, the PROMPT light prompts for a new command and turns off while a command is being entered. The C/R (carriage return) key terminates a command string, whereupon the PROMPT light turns on again. The STORE light turns on while a change command is executed; it indicates that something new is being stored. The STORE light turns off when the change command is terminated (by two consecutive carriage returns).

For more information on the PEPBUG commands, see Section 3.

2.3 Connecting a Terminal to the PEP System

Instead of using the keypad for program development, the PEP System user may prefer to connect a teletypewriter, video display terminal, or other serial I/O terminal.

An external terminal offers many advantages over program development and debugging from the keypad. A major benefit of a hard-copy terminal is that it produces a history of the steps taken and the results displayed. Overall system display by the user is much more convenient.

Input from an external terminal is much less error-prone. If you type carefully and examine each line of input before entering the carriage return, you can practically eliminate keyboarding errors. Also, a debugging session achieves more results more quickly.

An advantage of a terminal with storage capabilities (paper tape read/punch or connections to other tape or disk storage) is that, when you wish to end a debugging session, you may save the modified program using the PEPBUG F command. In a subsequent debugging session you can load the modified program and continue to debug. For extensive program patches, this can save much time and effort.

The PEP System can communicate with serial I/O terminals through two interfaces: RS-232-C standard three-wire interface and 20 mA current loop three-wire interface. In both modes, an ASCII terminal must be used in all upper-case mode (the PEP System does not recognize lower-case characters).

Information transfer is normally at 110 baud. To activate this communication speed from the keypad, enter a 6 into address location H'2BC3' as described in Section 2.1. To alter the communication speed, change the value in location H'2BC3' as described in the *PEPBUG User's Guide.*

Figure 2-1 illustrates the connections for the 20 mA current loop and EIA RS-232-C interfaces. For more information, Section 4.3 describes the Theory of Operation for the serial I/O interfaces.

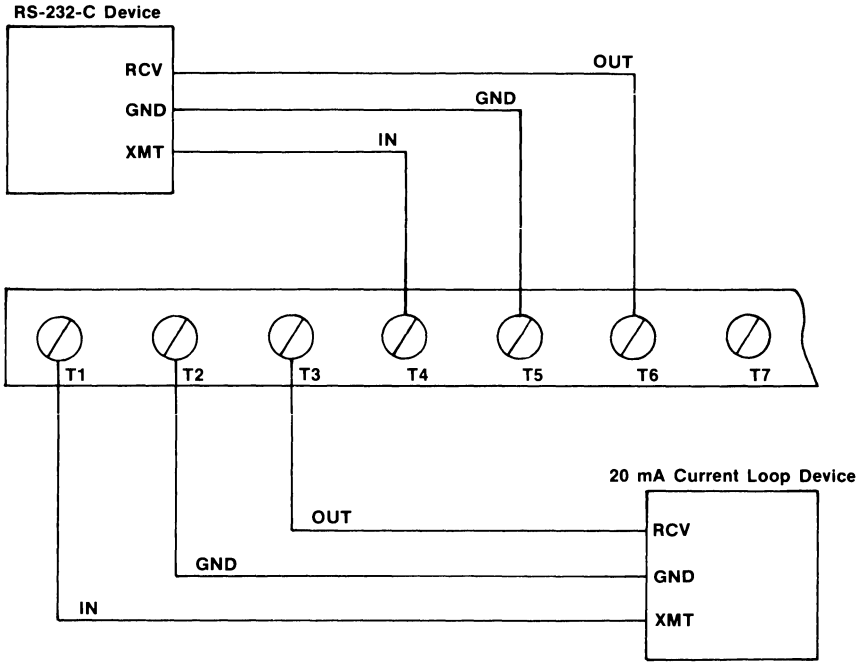


Figure 2-1 Connecting an External Terminal to the PEP System

2.3.1 20 mA Current Loop Interface

To use a terminal with a 20 mA current loop interface, make the following connections to the PEP System terminal strip:

1. Connect the transmit line of the terminal to T1.
2. Connect the signal ground line to T2.
3. Connect the receive line of the terminal to T3.
4. Set the C-D DIP switch on SW 3 to the C position.

2.3.2 RS-232-C Interface

To use a terminal with an EIA RS-232-C interface, make the following connections to the PEP System terminal strip:

1. Connect the transmit line of the terminal to T4.
2. Connect the signal ground line of the terminal to T5.
3. Connect the receive line of the terminal to T6.
4. Set the C-D DIP switch on SW 3 to the D position.
5. Connect the -12 volt power supply to T12.

Using the RS-232-C serial I/O circuit on the PEP System requires a -12 volt power supply (in addition to the required +5 volt and +12 volt power supplies) to provide the negative voltage levels used at the communications interface. Connect the -12 volt power supply at terminal T12. Current loop operation does not require the -12 volt supply.

2.3.3 Transfer of Control to Serial I/O Device

Before using an external terminal to communicate with the PEP System, make sure that the three-position toggle switch, SW 2, is in the TERMINAL position. The switch was in the KEYPAD position during system startup in order to enter the Baud Rate Counter (communication speed information). Now place the switch in the TERMINAL position and press the RESET button.

For information on using PEPBUG from an external terminal (command formats vary slightly from the keypad commands), refer to Section 3.

2.4 Loading Code from an External Device

Although you can enter program code manually into the PEP System RAM using the keypad or a terminal, it is easier to enter large programs by using the PEPBUG data entry routines and loading the program code from an external device. Loading object-formatted programs from an external device can be performed using either a serial or parallel interface.

The PEPBUG LOAD command enters formatted serial data using either the RS-232-C or the 20-mA current loop serial I/O interface. The program may be on paper tape in a teletypewriter paper tape reader. It may also be downloaded from a terminal with local mass storage. Both FAIRBUG and FORMULATOR formats are acceptable; refer to the *PEPBUG User's Guide.*

When loading serial data, be sure that the PEP System is set up for communication with an external terminal, as described in Section 2.3.

To enter formatted parallel data, the parallel device is connected at I/O port B of device U4 (see Appendix A). The usual parallel device is a high-speed paper tape reader which reads 100 to 300 characters per second. However, you may use any byte-parallel device controlled by a "hand-shaking" protocol to eliminate synchronization problems. For more information on loading programs from a parallel interface, refer to the *PEPBUG User's Guide.*

2.5 Downloading Code from Another Computer System

The PEP System user can also enter program code from another computer system which is communicating with the external terminal being used for PEP System input/output.

To download code to the PEP System from another computer system, using the serial terminal, follow this procedure:

1. Connect the signal ground line of the terminal to T2 on the PEP board for a 20 mA current loop terminal or to T5 for an RS-232-C terminal.

2. Connect the receive line of the terminal to the PEP board at T1 for a 20 mA current loop terminal or to T4 for an RS-232-C terminal.

3. Issue a LOAD command from the terminal keyboard. The PEP System is now waiting for input from the outside computer system.

4. From the terminal, issue a command to the outside computer system to dump or punch the desired program. This program will then be downloaded into the PEP System.

2.6 Microprocessor Prototyping

The system designer can use the PEP System not only as a vehicle for component emulation of single-chip devices, but also for performing general system-level emulations of the multi-chip F8 family of devices. Thus the PEP System may be used for prototyping and debugging any F8 microprocessor application which requires up to 10K bytes of memory, using the 4K bytes of static RAM and the 6K bytes of EPROM memory.

The user may also take advantage of the program debugging capabilities of the system monitor to develop the necessary application software in RAM, then use the EPROM programming capability of the PEP System to place the software into non-volatile EPROM for storage (See Section 2.8).

The prototyping flexibility of the PEP System is further enhanced because the user's program can access various firmware subroutines located in the PEPBUG system monitor. These subroutines are described in Section 3.

2.7 In-Circuit Emulation of Single-Chip Microcomputers

The F387X PEP System can emulate the F3870, F3872, F3876, and the F3878 single-chip microcomputers. The logic functions and signals accessible through the J1 socket of the PEP System are the same as the functions and signals associated with these microcomputers.

In the PEP System, the F3850 CPU supplies I/O ports 0 and 1, and the F3871 PIO supplies I/O ports 4 and 5, as well as the strobe, the timer, and the interrupt logic. Thus the PEP System performs, in real time, the required functions of timer, four I/O ports, the data strobe, the interrupt line, and the reset line.

By connecting a 40-pin connector and cable to the J1 socket, these signals can be interfaced to a user's circuit. The end of the 40-pin cable performs electrically and logically just like a dedicated microcomputer in the user's circuit.

In the PEP System, RAM and EPROM code substitute for the mask-programmed ROM code normally present in a dedicated microcomputer. This reduces costly turnaround time in software and firmware development, because the user can develop, debug, and check out program code directly either from the on-board keypad or from a serial I/O terminal.

For in-circuit emulation of these devices, one end of the 40-pin emulation cable must be placed in the J1 socket and the other end of the cable placed in the socket on the user's board where the single-chip microcomputer normally belongs. Connect pin 1 at one end of the emulation cable (the J1 socket) to pin 1 at the other end of the cable (the user's socket). Connect all the other contacts, except for pin 21, in the same manner.

Pin 21 of socket J1 is the +25 volt programming pin. Do not connect this pin to the user's system. Leave it open or ground it during emulation.

The software program to be used in the emulation must reside in the lowest page of memory. If the program resides in RAM, then this RAM must start at address 0000. This address must also be the entry point of the program, where program execution will commence after any type of reset (except when the reset is directed into the monitor).

To perform in-circuit emulation, place the three-position toggle switch, SW 2, in the USER position. Then press the RESET button, switch SW 1, to start the emulation process.

For additional information on emulation, primarily on supplying an external clock pulse for emulation, refer to Section 4.5.

2.8 EPROM Programming

The PEP System can program both the F38E70 EPROM microcomputer and the 2716-type EPROM. The PEPBUG monitor provides the ability to address various portions of EPROM. This is very convenient for modular program development, because subroutines or small segments of data can be written into the EPROM at one time and more added later. A library of subroutines can be built in this manner.

Thus a programmer can generate and debug subroutines or other program modules in RAM, move them to EPROM for execution, and repeat this as often as necessary. Finally, he can debug the main program logic in RAM and move it to EPROM as well.

To program F38E70 and 2716-type EPROM's, use the following procedure:

1. Plug the EPROM into the appropriate programming socket (J1 for the F38E70, J2 for the 2716).

2. Write H'FF' into Port 0 (2716 to read mode).

3. Write zero into Port 1 (F38E70 to read mode).

4. Connect a +25 volt power supply to T8 on the PEP System terminal strip. The voltage may vary between 24 and 26 volts, depending on the EPROM specifications.

5. Store the parameter list in six bytes of memory as follows:
   a. memory starting address at H'2BD8' and H'2BD9'
   b. EPROM starting address at H'2BDA' and H'2BDB'
   c. EPROM ending address at H'2BDC' and H'2BDD'
   d. H'FF' at H'2BDE' to perform a blank check, or zero at the same address to skip the blank check.

6. When using the keypad, press the E70 or 2716 key to begin programming. From a terminal, enter a GO H'8745' command to begin programming the F38E70 or a GO H'8741' command to begin programming the 2716.

7. If the parameter list specified a blank check, the system will blank check before the programming process.

8. If an error is encountered during the programming process, control returns directly to the PEPBUG monitor.

9. When the programming is complete, control returns to the PEPBUG monitor.

10. Check registers as appropriate to determine if the programming is correct; refer to the *PEPBUG User's Guide* for detailed information on examining register contents. (Section 3 of this manual gives an overview of PEPBUG.)

11. To perform a low-voltage verification of the programming, remove the 25-volt power supply from T8 and replace it with a 5-volt supply. Then repeat the above programming procedure starting at Step 6, after changing memory location H'2BDE' to zero to bypass the blank check.

12. When programming and verification are complete, remove the EPROM from its programming socket and insert it into the appropriate memory mapping socket.

2.9 User Options

The F387X PEP System provides a group of alternative connections to determine which of the different configurations on the circuit board is operating at a given time. The circuit board options include memory address decoding logic, clocking options, and keypad/terminal options.

Depending on which microcomputer the user is emulating and which stage of program development is in progress, different configurations of the RAM and EPROM memory are appropriate. Most of the options for the PEP System involve reconfiguring memory. Depending on the application of the PEP System, these options may be used often, infrequently, or not at all.

A DIP switch, SW 3, enables the user to select different operation configurations. The C or D position of the switch selects RS-232-C or 20 mA current loop serial I/O logic, as described in Section 2.3. See also Section 4.3 for a discussion of the serial I/O Theory of Operation. The other DIP switch settings, A-B, E-F, and H-J, enable the user to select several different memory configurations, especially when used in combination with other connection options.

In addition to the configurations you can produce with the DIP switch, the PEP System also provides some connection options which reconfigure memory. These connections consist of two or three adjacent plated-through holes for each connection. Located at various well-marked locations around the PC board, the connection holes are labeled with numbers preceded by the letter E (e.g., E1, E2, E3). These connection options are defined in *Tables 2-2* and *2-3*.

The configuration most frequently selected is installed by Fairchild as default connections, using traces on the plated board rather than discrete wire jumpers. These default connections are not permanent. They may be removed and, if desired, later replaced with a discrete wire jumper. *Table 2-4* describes the status of the connection options on the F387X PEP System when it is shipped.

To remove a default connection, carefully cut the trace between its two solder pads with a small, sharp knife. To implement a connection, add a wire jumper between two plated-through solder pads.

In any pair of alternate connections (see *Tables 2-2* and *2-3*), only one connection should be in place at a time. If both connections are in place when power is applied, the system will not operate properly. Damage to the system is also possible.

Table 2-2 Jumper Connection Options for Memory Mapping*

Connection	Significance
E1 to E2	**Default** connection which implements the standard memory configuration of 2K-byte memory blocks within each 4K-byte page (see A of Figure 2-2).
E1 to E3	Optional connection which reverses the order of 2K-byte memory blocks within each 4K-byte page. If the E1 to E2 connection configuration is A B C D, then the E1 to E3 configuration is B A D C.
E4 to E5 and E6 to E7	**Default** connections which implement the standard memory configuration of contiguous 4K-byte blocks of RAM and EPROM.
E4 to E7 and E5 to E6	Produces memory configuration which interleaves 2K-byte blocks of RAM and EPROM. If the default memory configuration is A B C D, connecting E4 to E7 and E5 to E6 produces the configuration A C B D.
E8 to E9	Enables the chips in sockets U10 and U11.
E11 to E12	**Default** connection which causes the PROM to serve as the following four sets of addresses: A000-A7FF, A800-AFFF, B000-B7FF, B800-BFFF.
E10 to E12	When this connection is made, the PROM serves as the following four sets of addresses: 8000-87FF, 8800-8FFF, 9000-97FF, 9800-9FFF.
E13 to E14	Optional connection which places the F6810 128-byte RAM used for system monitor workspace into address locations 0B80 to 0BFF.
E14 to E15	**Default** connection which places the F6810 128-byte RAM into address locations 2B80 to 2BFF.

*For more information on these memory mapping options, refer to Section 4.2.

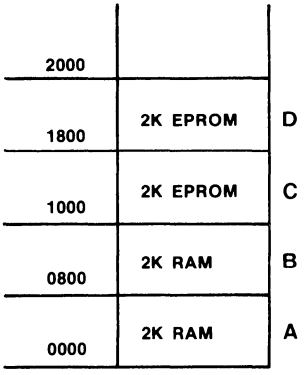
Table 2-3 Other Jumper Connection Options

Connection	Significance
E16 to E17	This connection ties the Vcc supply (pin 40) of socket J1 to the +5 volt power supply to provide power transfer during emulation.
E18 to E19 and E20 to E21	When these connections are made and the crystal is disconnected, external clocking or off-board crystal arrangements may be used.

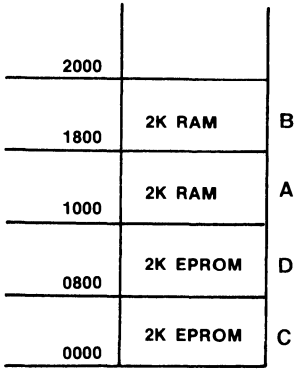
Table 2-4 Default Jumper Connections

E1 to E2	Connected
E1 to E3	Not Connected
E4 to E5	Connected
E6 to E7	Connected
E8 to E9	Not Connected
E10 to E12	Not Connected*
E11 to E12	Connected*
E13 to E14	Not Connected
E14 to E15	Connected
E16 to E17	Not Connected
E18 to E19	Not Connected
E20 to E21	Not Connected

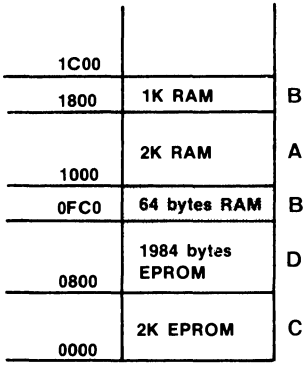
*If PEPBUG is supplied on a 68316 in socket U16 (see Section 1.2), the system is shipped with E10 connected to E12 and the E11 to E12 connection open.



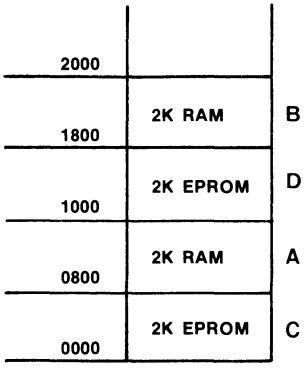
A) Switches E, H and A set.
Default connections:
E1 to E2, E3 open,
E4 to E5, E6 to E7.



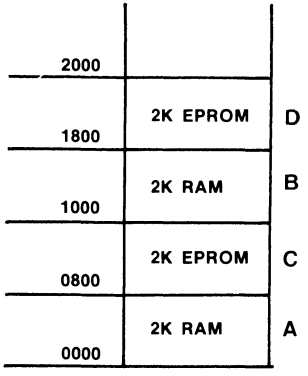
B) Switches F, H and A set.
Default connections:
E1 to E2, E3 open,
E4 to E5, E6 to E7.



C) Switches F, J, and B set.
Default connections:
E1 to E2, E3 open,
E4 to E5, E6 to E7.



D) Switches E. H and A set.
Jumper connections:
E1 to E3, E2 open,
E4 to E7, E5 to E6.



E) Switches E, H and A set.
Jumper connections:
E1 to E2, E3 open,
E4 to E7, E5 to E6.

Figure 2-2 Examples of Memory Configurations and Associated Jumper Connections

2.9.1 Memory Configuration Options

Figure 2-2 illustrates 8K bytes of the PEP System memory address space, showing the alphabetic notation used in the rest of this manual to illustrate memory configuration options. Illustrations A, B, and C of Figure 2-2 show the memory configurations you can select simply by setting the DIP switch, making no other alterations to the PEP System as it is shipped from the factory.

Illustration A shows the most typical configuration for a 4K program. It is particularly useful when you are downloading the program from a high-speed device and plan to store all instructions in RAM. Illustration B shows a configuration that would be more useful in program debugging and EPROM programming.

The configurations in Illustrations D and E of Figure 2-2 require changing the default plated connections as well as setting the DIP switches. The configuration in Illustration D is useful for extensive program development, when a large program is still evolving. The upper 2K of RAM can be used for new program development. Some programmers may prefer the configuration in Illustration E, which is convenient for building a subroutine library.

Note that Illustration C shows the emulation mode for the F3876 with a program in EPROM, while Illustration D is the emulation configuration for the F3872. These configurations provide 64 bytes of RAM as the last 64 bytes of the first 4096 bytes of memory. For more information on F3872 and F3876 emulations, see Section 4.2.4.

For a detailed discussion of memory mapping and the memory configurations available to the PEP System User, refer to Section 4.2.

2.9.2 Other User Options

If you wish to supply +5 volt power to the PEP System from another system (for example, from a prototype board containing a power supply), it can be supplied through the emulation cable. Connecting E16 to E17 ties the Vcc supply (pin 40) of socket J1 to the +5 volt power supply of the PEP System. During emulation, this allows +5 volt power to be supplied between the PEP System and the user's circuit via the emulation cable.

The PEP System normally operates with an on-board 2.000 MHz crystal clock. To use external clocking, disconnect the crystal and make connections E18 to E19 and E20 to E21. For more discussion on using an external clock reference, see Section 4.8.

Section 3
THE PEP SYSTEM MONITOR

3.1 PEPBUG Overview

PEPBUG, the PEP System monitor, controls the operation of the PEP System and provides a convenient, powerful debugging facility to aid in the development of F387X and F8 programs.

PEPBUG is an interactive system. The programmer enters PEPBUG commands either from the on-board PEP System keypad (see *Figure 3-1*) or from an external terminal. With these commands, PEPBUG can perform the following program development functions:

Display or alter memory locations
Display or alter scratchpad registers
Display or alter I/O ports (addresses)
Display or alter accumulator, ISAR, status (W register)
Display or alter PC0, PC1, DC0, DC1
Load formatted paper tape (FAIR-BUG or Formulator format)
Punch formatted paper tape (Formulator format)
Punch paper tape in PROM format (8-bit binary format)
Entry from keyboard or by program instruction
I/O subroutines available to user
Set breakpoints and enter PEPBUG from an executing program
Memory block move
Hexadecimal arithmetic
Program F38E70 or F2716 EPROM

The program storage unit containing PEPBUG is assigned memory addresses H'8000' to H'87FF', with the entry point at '8080'. The port assignments are as follows:

I/O port A	Port 28
I/O port B	Port 29
Local Interrupt Control	Port 2A
Timer	Port 2B

The interrupt address vector for the timer is H'0300' and for an external interrupt is H'0380'. PEPBUG also uses RAM at memory locations H'2B80' to H'2BE5' as a workspace area.

When you start up the PEP System normally (see Section 2), the PEPBUG monitor is in control, and you can begin issuing PEPBUG commands immediately. PEPBUG can also be explicitly entered in several ways. To enter PEPBUG from another software module, execute a program instruction such as PI'8080' or JMP'8080'.

PEPBUG will save the state of the system upon entry and will restore it upon return to the user's program. The interrupt is disabled by FAIR-BUG and may be re-enabled by the user if desired.

3.2 PEPBUG Commands

The PEPBUG commands, with brief descriptions of their format and use, are listed in *Table 3-1*. For detailed descriptions of the PEPBUG commands, refer to the *PEPBUG User's Guide*.

When the keypad is in use for I/O, the PROMPT light indicates when PEPBUG is ready to accept commands. The PEPBUG commands are shown on the 24 keys of the keypad. After a command key is pressed, the PROMPT light goes off. PEPBUG then recognizes further keystrokes as hexadecimal digits until the C/R key is pressed to terminate the command string.

On an external terminal, the first letter of the command name represents the command, after which the user enters the command string and carriage return normally. A few of the PEPBUG commands are different when entered from an external terminal from their format when entered from the keypad; these differences are listed in *Table 3-1*.

A carriage return (or pressing the C/R key on the keypad) terminates all commands except the change and delete commands. This normal use of the carriage return is not shown in the command formats in *Table 3-1*; only atypical uses of the carriage return are indicated. That is, the command formats assume that all command strings end with a single carriage return unless otherwise specified.

When the CHG key is pressed, the STORE light goes on to indicate that the contents of memory are changing. The STORE light remains on until the user enters two consecutive carriage returns to terminate the change command.

Table 3-1 Summary of PEPBUG Commands

Keypad Key	Terminal Command	Command Format	Function
C/R	Carriage return		Carriage return. Terminates a command string, except for change and delete commands.
DEL	[	DEL or [with no carriage return.	Delete. Cancels any command string which has not yet been terminated with a C/R.
CHG	C	Cxx(C/R)(C/R)	Change. Changes the previously displayed memory location, port, or register to xx. Note: terminate this command string with *two* C/R's.
		Cxx(C/R)yy(C/R) . . . (C/R)(C/R)	Changes the sequential registers or memory locations to xx, yy .... Terminate the command string with two C/R's.
		Cxxxx(C/R)(C/R)	Changes the previously displayed PC or DC to xxxx. Terminate the command string with two C/R's.
PORT	P	Pxx	Displays the data at port xx.

Table 3-1 Summary of PEPBUG Commands (cont.)

Keypad Key	Terminal Command	Command Format	Function
REG	R	Rxx	Displays the contents of register xx.
		Rxx-yy	Displays the contents of the block of registers that includes registers xx to yy.
MEM	M	Mxxxx	Displays the contents of memory location xxxx.
		Mxxxx-yyyy	Displays the contents of the block of memory that includes addresses xxxx-yyyy.
NEXT	N	N	Displays the contents of the next register, port, or memory location.
PREV	B	B	Displays the contents of the previous register, port, or memory location.
—	E	E	Examine; displays the contents of the last addressed register, memory location, or port. This command can only be used from an external terminal; there is no corresponding keypad command.
W	W	W	Displays the contents of the W or status register (refer to the *PEPBUG User's Guide* for the significance of each bit in the status register).
ACC	A	A	Displays the contents of the accumulator.
IS	I	I	Displays the contents of the Indirect Scratchpad Address Register (ISAR).
PC	P	P0	Displays the contents of the Program Counter (PC0).
		P1	Displays the contents of the Stack Register (PC1).

Table 3-1 Summary of PEPBUG Commands (cont.)

Keypad Key	Terminal Command	Command Format	Function
DC	D	D0	Displays the contents of Data Counter 0 (DC0).
		D1	Displays the contents of Data Counter 1 (DC1).
MOVE	V	Mdddd Vssss-eeee	Moves a block of memory. A memory display command specifying the first byte of the destination (dddd) must precede the move command. The move command then provides the starting address (ssss) and the ending address (eeee) of the block of memory whose contents are to be moved.
FIND	K	Mxxxx Kyy	Finds the first byte which matches yy, starting the search through memory at the last displayed address xxxx.
HEX	X	Xaaaa + bbbb = (no C/R) or Xaaaa + bbbb(C/R)	Adds value bbbb to value aaaa. On an external terminal, the equal sign can be used to replace the normal carriage return terminating the command string.
		Xaaaa - bbbb = (no C/R) or Xaaaa - bbbb(C/R)	Subtracts value bbbb from value aaaa. On an external terminal, the equal sign can be used to replace the normal carriage return terminating the command string.
LOAD	L	L	Loads a formatted object program such as a teletypewriter paper tape from the RS-232-C device.
—	H	H	Loads a formatted object program from a high-speed paper tape. There is no corresponding keypad instruction, but a GO H'84BB' command executes the high-speed load routine.

Table 3-1 Summary of PEPBUG Commands (cont.)

Keypad Key	Terminal Command	Command Format	Function
GO	G	G	Goes to address PC0 and executes, after restoring all registers (but not the accumulator).
		Gaaaa	Changes PC0 to address aaaa, then goes to aaaa and continues execution.
BRK	U	Uaaaa	Sets a breakpoint sequence at address aaaa. Refer to the *PEPBUG User's Guide* for details.
B. CLR	T	T	Clears the breakpoint and restores the previous user instructions to the breakpoint address.
E70	GO H'8754'	—	Programs the F38E70, as described in Section 2.7.
2716	GO H'8741'	—	Programs the F2716, as described in Section 2.7.
—	J	Jxxxx-yyyy-z	Punches binary paper tape in PROM format, with xxxx the starting page address. yyyy is the ending page address + 1 (the next page address). z is the number of bytes per block; 0 specifies 256, 1 specifies 512. There is no corresponding keypad command.
—	F	Fxxxx-yyyy	Punches paper tape in FORMULATOR format, in 16-byte segments which include the starting address xxxx and the ending address yyyy.

3.3 PEPBUG Subroutines

PEPBUG offers the PEP System user additional prototyping tools with a group of firmware subroutines which the user's program can access. The following subroutines are located in the PEPBUG System monitor; they are described in detail in the *PEPBUG User's Guide*.

SCAN — Scans the on-board keypad for key depressions, enabling the user program to employ the keypad as an input device.

DISPX — Displays information at the on-board address and data displays.

TTX — Inputs one byte of data in a bit-serial manner, including the associated start and stop bits.

TTYX — Outputs one byte of data in a bit-serial manner, including the associated start and stop bits.

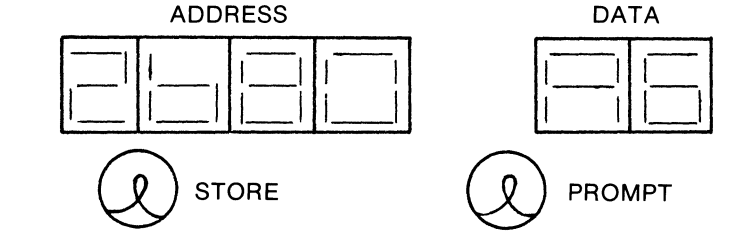
TCRX — Outputs a carriage return, line feed, and 250 ms delay, using the TTYX subroutine.

PINP — Inputs one byte of data in a byte-parallel manner (from a parallel input device).

F0P1 — Outputs one or two hexadecimal digits in ASCII format from a memory location.

F0P2 — Outputs one or two hexadecimal digits in ASCII format from register QL.

BYTE — Inputs two ASCII digits from a parallel or serial input device and converts them to one hexadecimal byte.



BRK C	B.CLR D	LOAD E	GO F	-	+
E70 8	HEX 9	MOVE A	FIND B	PORT	CHG
2716 4	IS 5	W 6	ACC 7	REG	DEL
PC 0	DC 1	PREV 2	NEXT 3	MEM	C/R

Figure 3-1 PEP System Keyboard

Section 4
THEORY OF OPERATION

4.1 Review of the F8 System

The following pages provide a brief review of F8 system concepts to give the reader a more thorough understanding of the internal operations of the PEP System circuitry. For complete F8 tutorial documentation, refer to the *F8 User's Guide*.

4.1.1 F8 Microprocessor Family

F3850 Central Processing Unit (CPU)

The F3850 Central Processing Unit (CPU) provides processing capabilities and two of the four I/O ports required for emulation of F387X family devices. The CPU is an 8-bit arithmetic device with 70 instructions. It contains a 64-byte RAM, an instruction register, an accumulator, two parallel I/O ports, an interrupt control, power-on reset, and clock generation logic. The CPU provides control lines to the other members of the family.

F3856 and F38T56 Program Storage Units (PSU's)

The F3856 Program Storage Unit (PSU) is not just a conventional read-only memory. In addition to containing 2048 bytes of mask-programmable ROM for program and constant storage, the F3856 includes the addressing logic for memory referencing, a program counter, an indirect address register (the data counter), and a stack register.

A complete vectored interrupt is provided, including an external interrupt line to alert the central processor. The F3856 contains all of the logic necessary to request, acknowledge, and reset the interrupt. The 8-bit programmable timer is especially useful in generating real-time delays. The F3856 has an additional 16 TTL-compatible, bi-directional, fully latched I/O lines.

The F38T56 contains the F3871-type timer and strobe logic instead of the F3856 timer logic.

F3853 Static Memory Interface (SMI)

The F3853 Static Memory Interface (SMI) provides a 16-bit address bus and read and write controls, enabling the CPU to access external memories consisting of static RAMs, EPROMs, and PROMs such as the F2102 RAM, the F2708 EPROM, and F93448 PROM.

The SMI also contains a program counter, data counter, an auxiliary data counter, stack register, incrementer/adder, a programmable timer, and an 8-bit data bus for communication with external memory. The F3853 may be used with the CPU alone or in conjunction with other F8 devices.

F3861 Peripheral I/O Device (PIO)
The F3861 Peripheral I/O Device (PIO) is an expansion unit for I/O ports, interrupts, and timers. It contains two 8-bit I/O ports, one interrupt control, and one programmable timer. Depending on the application requirements, multiple PIOs can be added to the system to expand the functions at low cost.

F3871 Peripheral I/O Device (PIO)
The F3871 Peripheral I/O Device (PIO) is an expansion unit for I/O ports, interrupts, and timers. It contains two 8-bit I/O ports, one interrupt control, and one programmable timer. The versatile timer/interrupt circuit provides a timer with resolution of 1.0us at 2.0 MHz, measures external pulse widths, or counts external pulses. Depending on the application requirements, multiple PIOs can be added to the system to expand the functions at low cost.

4.1.2 The F8 System Bus

The separate components in the F8 System communicate with each other via the F8 System bus. This system is composed of the following elements:

1. Data Bus: 8 lines, D0 through D7
2. Control Bus: 5 lines, ROMC0 through ROMC4
3. Clocks: 2 lines, Φ and WRITE
4. Interrupt Line: $\overline{\text{INT REQ}}$

Data Bus (D0 through D7)
These eight bi-directional lines transfer data, addresses, and input/output signals between various devices in the F8 system, including external memory. Only one device in the system can drive signals on the Data Bus at any given time.

Control Bus (ROMC0 through ROMC4)
These five lines transmit the 32 possible commands from the 3850 CPU to the rest of the devices in the system. At the beginning of each cycle, the CPU decodes the instruction and sends commands to the Control Bus. The Control Bus lines then transmit to each device in the system the operation to be performed by the device during the current cycle.

Clocks (Φ and WRITE)
These two clocks carry all of the required timing information from the 3850 CPU to the rest of the devices in the system.

The Φ signal is generated by the CPU and is used to synchronize the entire microprocessor system. The WRITE clock has a pulse-width of one Φ period. The WRITE clock period is either four Φ periods for short instruction cycles or six Φ periods for long instruction cycles. All of the registers and flip-flops in the F8 devices are either updated or refreshed during the HIGH state of the WRITE clock.

Interrupt Lines $\overline{\text{PRI IN}}$, $\overline{\text{PRI OUT}}$, and $\overline{\text{INT REQ}}$
There are three lines which control the distributed interrupt structure of the F8 system. The $\overline{\text{PRI IN}}$ and $\overline{\text{PRI OUT}}$ lines connect the various interrupt circuits into a priority daisy chain; this allows only one F8 device at a time to interrupt the CPU. The $\overline{\text{INT REQ}}$ line is a wire-ORed signal from the other F8 devices to the 3850 CPU requesting an interrupt.

4.1.3 The Input/Output Structure

The input/output structure of the F8 system is distributed throughout the system. The CPU, the PSU, and the PIOs each possess two bi-directional 8-bit ports. Each I/O port has a unique 8-bit I/O address and can be accessed by the CPU for either input or output.

In the output mode, each bit is LOW when true and latches the output data until the CPU writes new data to the port. A zero written to the port indicates a HIGH output, and a zero read from the port indicates that a HIGH TTL level was present. Normal operation is to clear the I/O port before the input of data. If the port is used for input only, the clear need be done only once.

Input Data Convention

The F8 ports invert data on both output and input. Thus an external HIGH is interpreted as an internal zero and an external LOW line is read as an internal one. Since the ports invert data so that a one is HIGH when it comes out of the system, the user program should write a zero (which produces a HIGH TTL output) when you want a HIGH signal. Thus there is no need for external inverters, since software can invert data as needed.

Although there are several types of output options available for F8 ports, the standard (internal) pull-up configuration is normally used with the PEP System. In this configuration, an internal pull-up is provided to insure that the CPU senses unused I/O pins as logical zeros.

The Interrupt Structure

The interrupt structure of the F8 system is distributed among the various devices in the system. Each PSU, PIO, and SMI has its own independent interrupt structure, with an external interrupt input, an interrupt address vector, and an internal timer interrupt.

When one of the F8 chips is enabled and it senses a LOW condition on its $\overline{\text{EXT INT}}$ pin, if there is no higher priority interrupt in the system, the device sends a LOW signal on the wire-ORed $\overline{\text{INT REQ}}$ line to the CPU. The CPU then directs program execution to the appropriate service routine.

Interrupt priority is established by $\overline{\text{PRI IN}}$ and $\overline{\text{PRI OUT}}$ lines which connect the various interrupt circuits into a priority "daisy chain". This daisy chain arrangement allows only one F8 device at a time to interrupt the CPU.

However, in the F387X PEP System, only the interrupt request line from the F3871 PIO is connected to the CPU. The interrupt circuits associated with the other sockets are not connected. A hardware modification can make these circuits available to the user if desired (this will also require software considerations).

4.2 The Memory

Several options on the PC board enable the user to reconfigure RAM and EPROM memory locations in the address space of the PEP System.

Figure 2-2 in Section 2 illustrates the PEP System memory address space, showing the alphabetic notation used throughout this manual to illustrate memory configuration options. Using this notation, *Table 4-1* summarizes the memory configurations available on the PEP System and shows what connections produce those configurations. The default connections are noted.

Also note that, as shown in Table 4-1, A in the alphabetic notation represents the RAM chips at U6, U7, U8, and U9. B represents RAM at U10, U11, U12, and U13. C represents EPROM at U14, and D represents EPROM at U15.

Table 4-1 Alternative Memory Configurations on the PEP System

E1 to E2 (default)	E1 to E3 (optional)	DIP Switch E	DIP Switch F	E4 to E5 and E6 to E7 (default)	E4 to E7 and E6 to E5 (optional)	RAM				EPROM	
						A		B		C	D
						CSA U6:U7	CSB U8:U9	CSC U10:U11	CSD U12:U13	U14	U15
X		X				0000-03FF	0400-07FF	0800-0BFF	0C00-0FFF	1000-17FF	1800-1FFF
	X	X		X		0800-0BFF	0C00-0FFF	0000-03FF	0400-07FF	1800-1FFF	1000-17FF
X			X	X		1000-13FF	1400-17FF	1800-1BFF	1C00-1FFF	0000-07FF	0800-0FFF
	X		X	X		1800-1BFF	1C00-1FFF	1000-13FF	1400-17FF	0800-0FFF	0000-07FF
X		X			X	0000-03FF	0400-07FF	1000-13FF	1400-17FF	0800-0FFF	1800-1FFF
	X	X			X	1000-13FF	1400-17FF	0000-03FF	0400-07FF	1800-1FFF	0800-0FFF
X			X		X	0800-0BFF	0C00-0FFF	1800-1BFF	1C00-1FFF	0000-07FF	1000-17FF
	X		X		X	1800-1BFF	1C00-1FFF	0800-0BFF	0C00-0FFF	1000-17FF	0000-07FF

Note: Sockets U10:U11 are not active unless E8 and E9 are connected.
Sockets U12:U13 are not active unless DIP switch H-J is in the H position.
Socket U15 is not active unless DIP switch A-B is in the A position.

Of the possible memory configurations for the PEP System, certain options are of particular interest to the system designer performing emulation. The following operations involving memory configuration are discussed in more detail in the rest of this section:

1. Reversing the order of 2K-byte memory blocks in each 4K-byte page.
2. Interleaving 2K-byte blocks of RAM and PROM.
3. Locating RAM or EPROM in the base page (addresses 0000 to 0FFF).
4. Emulating F3872 and F3876 microcomputers (which require special considerations to emulate their 64 bytes of on-board RAM).

4.2.1 Reversing the Order of 2K-Byte Memory Blocks

The PEP System user can reverse the order of the two 2K-byte memory blocks in each of the two 4K-byte pages. For example, the memory configuration expressed as A B C D can be changed to B A D C. Similarly, the memory configuration C D A B can be changed to D C B A. This option is useful in conjunction with other connection options that reconfigure memory.

To implement this reversal:

1. Connect E1 to E3, and
2. Break the trace connecting E1 to E2 (the default connection).

4.2.2 Interleaving 2K-Byte Blocks of RAM and PROM

The PEP System RAM and EPROM may be either contiguous or interleaved in 2K-byte increments, whichever configuration is more useful for program development. The default connections of E4 to E5 and E6 to E7 (as shipped from the factory) configure the memory space into blocks of up to 4K bytes of contiguous RAM and up to 4K bytes of contiguous EPROM (shown in A and B of *Figure 2-2*, Section 2).

Reversing these connections configures memory space with interleaved RAM and EPROM in 2K-byte blocks (shown in *D* and *E* of *Figure 2-2*, Section 2). For example, if the system memory is configured A B C D, connecting E4 to E7 and E5 to E6 produces an A C B D configuration. Similarly, the configuration C D A B becomes C A D B as a result of the same connection.

The option to interleave RAM and EPROM may be useful in developing programs for 4K microcomputers when one half or more of the program is already correct. The known-correct portion of the program can be placed in EPROM for easier use.

4.2.3 Locating RAM or EPROM in the Base Page

F387X single-chip microcomputer emulations execute programs starting at the first or lowest page in memory (address 0000). This is because an external or "power-on" reset sends program control to address 0000.

Since some devices in the F387X family of microcomputers provide up to 4K bytes of memory, the PEP System can have either 4K bytes of RAM or 4K bytes of EPROM at addresses H'0000' to H'0FFF'.

If the DIP switch is in position E, the 4K bytes of RAM are located at addresses 0000 to 0FFF, and up to 4K bytes of EPROM memory reside in the address range 1000 to 1FFF. This is represented as A B C D (shown in *A* of *Figure 2-2*, Section 2).

If the DIP switch is in position F, the RAM and EPROM exchange places. That is, the 4K bytes of EPROM reside in the address range 0000 to 0FFF, and the 4K bytes of RAM are located from 1000 to 1FFF. The configuration becomes C D A B (shown in *B* of *Figure 2-2*, Section 2).

4.3.2 20mA Current Loop Mode

In 20 mA current loop mode, an FCD 850 (U44) optical isolator buffers the complemented serial output data from pin 19 of U4. The isolator optically couples the signal to the current loop output terminal T3, but electrically isolates the current loop from the PEP System logic.

When the CPU is not outputting data to the serial I/O device (that is, when the line is in the "idle" or "marking" state), pin 2 of the FCD 850 (U44) is held at logical 0. Current then flows from the output of the isolator (pin 4 of U44), through the current loop, out line T3 to the user's I/O device, and back again through the ground return line T2.

When the I/O device closes the connection between the PEP System current loop in line T1 and the ground return line T2, current flows from the input of another FCD 850 optical isolator (pin 2 of U45). Closing a current path of ground through the I/O device causes the output of the isolator (pin 4 of U45) to be pulled to +5 volts.

U41 inverts this TTL logic HIGH and presents the signal to pin 6 of U4, where it can be read by the CPU.

4.4 Programming Sockets

4.4.1 Socket Connections

The F387X PEP System can program F38E70 EPROM microcomputers and 2716 2K-byte EPROM's each from a separate socket. The 40-pin J1 socket programs F38E70's, and the 24-pin J2 socket programs 2716's.

Ports 0 and 1 of the F8 CPU supply in parallel to sockets J1 and J2 the eleven address bits required to uniquely access all 2048 bytes of memory in each device. *Tables 4-2* and *Table 4-3* list the exact address definitions. Port 5 supplies to sockets J1 and J2 the data to be entered into the memory of the chip being programmed.

4.4.2 Data Verification

The contents of a programmed address location can be verified by reading that location. Verification is handled differently for F38E70's and 2716's.

For 38E70's in the J1 socket, be sure to verify that data is present at port 4 throughout the programming process, even though the data was entered via port 5. Port 4 can be used to verify a location immediately after programming that location.

To verify data for 2716's in the J2 socket, drop the output enable line (bit 7 of Port 0) to a logical 0, and read the memory contents of Port 5.

Table 4-2 Programming Socket J1 For Programming F38E70 EPROM Microcomputers

Signal		J1 Pin Number	Active Level
Address			
A10	Bit 3 of Port 1	34	HIGH
A9	Bit 2 of Port 1	35	HIGH
A8	Bit 1 of Port 1	36	HIGH
A7	Bit 0 of Port 1	37	HIGH
A6	Bit 0 of Port 0	3	HIGH
A5	Bit 1 of Port 0	4	HIGH
A4	Bit 6 of Port 0	17	HIGH
A3	Bit 5 of Port 0	18	HIGH
A2	Bit 4 of Port 0	19	HIGH
A1	Bit 4 of Port 1	22	HIGH
A0	Bit 5 of Port 1	23	HIGH
Data In			
D7	Bit 7 of Port 5	26	LOW
D6	Bit 6 of Port 5	27	LOW
D5	Bit 5 of Port 5	28	LOW
D4	Bit 4 of Port 5	29	LOW
D3	Bit 3 of Port 5	30	LOW
D2	Bit 2 of Port 5	31	LOW
D1	Bit 1 of Port 5	32	LOW
D0	Bit 0 of Port 5	33	LOW
Data Out			
D7	Bit 7 of Port 4	15	HIGH
D6	Bit 6 of Port 4	14	HIGH
D5	Bit 5 of Port 4	13	HIGH
D4	Bit 4 of Port 4	12	HIGH
D3	Bit 3 of Port 4	11	HIGH
D2	Bit 2 of Port 4	10	HIGH
D1	Bit 1 of Port 4	9	HIGH
D0	Bit 0 of Port 4	8	HIGH
Program Strobe			
PROG	Bit 6 of Port 1	24	LOW

The TEST pin (pin 21) must be set at +25V throughout the programming operation.

Table 4-3 Programming Socket J2 For Programming 2716 2K EPROMs

Signal		J2 Pin Number	Active Level
Addresses			
A10	Bit 3 of Port 1	19	HIGH
A9	Bit 2 of Port 1	22	HIGH
A8	Bit 1 of Port 1	23	HIGH
A7	Bit 0 of Port 1	1	HIGH
A6	Bit 0 of Port 0	2	HIGH
A5	Bit 1 of Port 0	3	HIGH
A4	Bit 6 of Port 0	4	HIGH
A3	Bit 5 of Port 0	5	HIGH
A2	Bit 4 of Port 0	6	HIGH
A1	Bit 4 of Port 1	7	HIGH
A0	Bit 5 of Port 1	8	HIGH
Data I/O			
D7	Bit 7 of Port 5	17	HIGH
D6	Bit 6 of Port 5	16	HIGH
D5	Bit 5 of Port 5	15	HIGH
D4	Bit 4 of Port 5	14	HIGH
D3	Bit 3 of Port 5	13	HIGH
D2	Bit 2 of Port 5	11	HIGH
D1	Bit 1 of Port 5	10	HIGH
D0	Bit 0 of Port 5	9	HIGH
Programming Controls			
PGM	Bit 6 of Port 1	18	HIGH
OE	Bit 7 of Port 0	20	LOW

4.5 Emulation Options

4.5.1 +5 Volt Power Supply

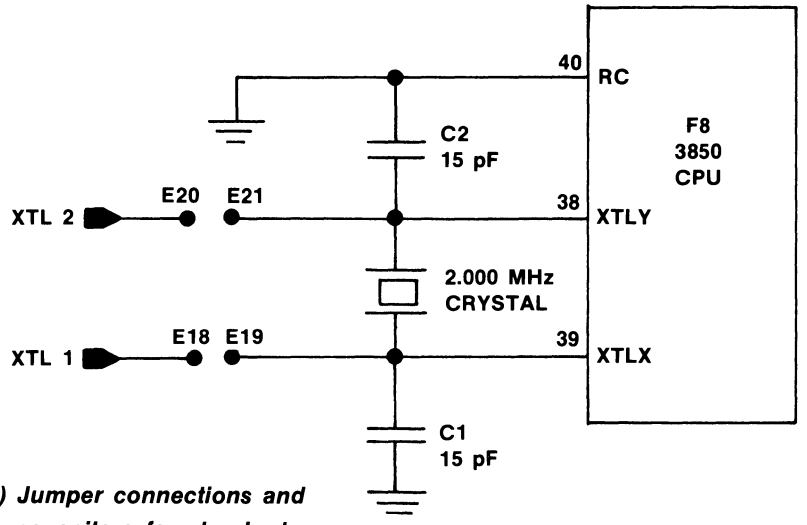
Connecting E16 to E17 ties the Vcc supply (pin 40) of socket J1 to the +5 volt power supply of the PEP System. During emulation, this allows +5 volt power to be supplied between the PEP System and the user's circuit via the emulation cable.

When the PEP System is shipped, the connection from E16 to E17 is left open. The connection must be made to program the F38E70 EPROM microcomputers in the J1 socket.

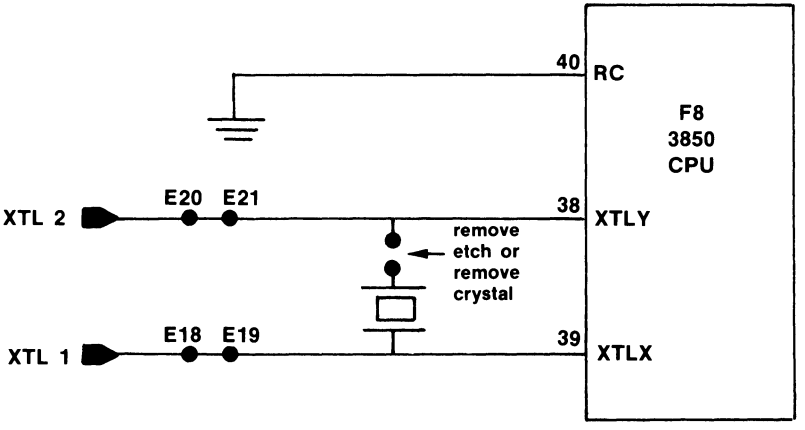
4.5.2 External Clock Reference

The PEP System normally operates with an on-board 2.000 MHz crystal clock. Pins 1 and 2 of J2 are the XTLX and XTLY clock lines respectively. When the PEP System operates with its own 2.000 MHz clock, pins 1 and 2 of J2 float.

When the PEP System is shipped, connections E18 to E19 and E20 to E21 are left open. This default configuration is shown in *A* of *Figure 4-1*. The modified configuration shown in *B* of *Figure 4-1* is required for external clocking of off-board crystal arrangements using the 12-inch, 40-pin jumper cable supplied with the system.



(a) Jumper connections and capacitors for standard use



(b) Jumper connections for use with non-standard frequency or clock reference

Figure 4-1 PEP System Configurations for Standard and Non-Standard Clock References

To operate at a different frequency or to synchronize the PEP System to an external clock reference, make one of the following two modifications:

1. A metal trace connects the on-board 2.000 MHz crystal to pin 38 of the 3850 CPU. Cut the metal trace at the two solder pads just above capacitor C2 on the lower right-hand side of the crystal. Use jumpers to connect E18 to E19 and E20 to E21. Connecting E18 to E19 connects XTLX to XTL1, and connecting E20 to E21 connects XTLY to XTL2.
2. Remove the crystal from the PEP System.

These modifications connect the clock in the user's circuit to pins 38 and 39 of the 3850 CPU. The user's system may also drive the XTLY line by supplying an external clock pulse if desired. When a TV crystal is used which supplies 3.58 MHz rather than 4.0 MHz, the user must adjust the programming to accommodate the different timing values.

Note: The PEP System cannot emulate the self-oscillating mode option or the internal clock "divide-by-two" circuits available on F387X microcomputers.

For further details on F8 clocking considerations, refer to the *F8 User's Guide* and the F3850 Data Sheet.

4.6 Keyboard and Display Circuits

A 24-key calculator-style keypad, with four rows and six columns of keys, provides an on-board method for entering data and commands to the system. The PEPBUG monitor periodically interrogates the keypad for key closures, that is, to determine when a key is pressed.

When a key is pressed, the row line associated with the key is at logical zero. When no keys are depressed, the four row lines float and the internal pullup resistors of the I/O port pull the row lines up to a logical 1.

To interrogate the keypad for key closures, the PEPBUG outputs a logical 0 to one of the six column lines and then senses the four row lines. If a key in that column is pressed, the logical 0 on the column line is connected to the row line of the row containing the depressed key. This procedure is repeated for each of the six columns until a logical zero is found on a row line, indicating a key closure.

When a key closure is found, PEPBUG "debounces" the key closure and locks out further keypad scanning until the key is released. There is no key rollover.

The same six lines that drive the keypad column lines select the display driver being written to. Each display driver (9368) is a binary-to-seven-segment decoder and latch. Display refreshing is not necessary with this decoder and latch.

4.7 Reset Circuit

4.7.1 External Reset Signal

The reset circuitry enables the PEP System user to exit current program execution and begin new program execution either at memory address 0000, where the user program normally resides, or at address H'8080', which is the entry to the debug monitor (PEPBUG).

When the external reset ($\overline{\text{EXT RESET}}$) pin is taken LOW, the 3850 CPU outputs H'00' on the data bus and H'08' on the ROMC control bus to all F8 devices. The F8 devices in the system copy the contents of the Program Counter (PC0) into PC1 and load the data bus into both halves of PC0. Thus, the next instruction fetched is from address location 0000.

4.7.2 USER/KEYPAD/TERMINAL Switch

When the USER/KEYPAD/TERMINAL switch, SW 2, is in the KEYPAD or TERMINAL position, a H'08' value on the ROMC control bus causes the enable pin of a three-state buffer to be driven HIGH. This electrically disconnects the D7 (most significant) bit of the data bus from its source in the CPU. The D7 bit is modified and becomes the D7X bit.

The D7X bit is the buffered form of D7. The D7X logical value differs from the D7 logical value only when switch SW 2 is in the KEYPAD or TERMINAL position. D7X goes to all F8 devices in the system except the 3850 CPU.

A pull-up resistor pulls the D7X bit HIGH and causes the data bus to look like an H'80' instead of a H'00'. The H'80' is loaded into both halves of the program counter, PC0. Since H'8080' is the entry point address of the monitor, program execution continues under monitor control.

In the USER position, SW 2 disables the "reset-into-monitor" circuit. An external reset sends program execution to address H'0000'.

When not at external reset, the D7 bit of the data bus functions normally. The Data Bus Drive signal, $\overline{\text{DBDR}}$, selectively enables one or the other of the back-to-back three-state buffers to control D7-bit data direction. Refer also to the schematic of the reset circuit, *Figure 4-2*.

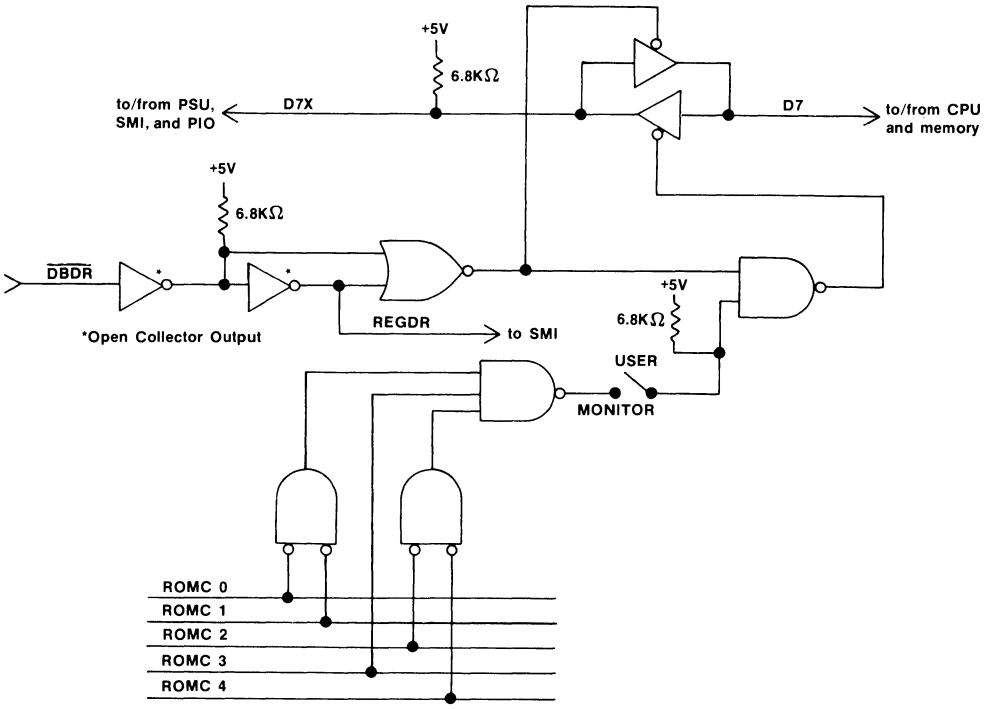


Figure 4-2 Schematic of the Reset Circuit

This switching capability makes it easy to execute program code from either RAM or EPROM memory starting at address 0000, and to switch back and forth between RAM and EPROM if necessary.

4.2.4 Emulating F3872 and F3876 Microcomputers

Emulating the F3872 or F3876 on the PEP System may require special configurations, because in these microcomputers, the last 64 bytes in the 4096-byte address range are on-chip RAM rather than mask-programmed ROM.

When PEP System emulation programs are stored in RAM in the base page (0000 to 0FFF), no changes from the normal configuration are necessary. When the emulation programs are stored in EPROM in the base page, the user must enable the PEP System 64-byte decode logic.

The 64-byte memory address decode logic detects an attempt to access memory in the range between H'0FC0' and H'0FFF' (decimal 4032 to 4095). When the CPU attempts to access the last 64 bytes of memory (locations H'0FC0' to H'0FFF'), the PEP System chip-select logic inhibits the second EPROM (in socket U15) from responding to these memory accesses and instead enables the static RAM 2114's (in sockets U12 and U13).

In this configuration, 1K of RAM actually overlaps 1K of the D block of EPROM, as shown in Illustration *C* of *Figure 2-2* in Section 2.

To enable the 64-byte decode logic, place the A-B DIP switch in position B and the H-J DIP switch in position J. For all other operations, including 4K microcomputer emulations with program code resident in RAM, place the A-B DIP switch in position A and the H-J DIP switch in position H.

4.3 Serial I/O Circuits

The F387X PEP System can communicate with a user's serial I/O device either in RS-232-C mode or in 20 mA current loop mode. To implement the serial I/O connections, refer to Section 2.3.

4.3.1 RS-232-C Mode

In RS-232-C mode on the PEP System, serialized data is output from pin 19 to U4, through an inverting NAND gate, to one section of a μA1488 (U42) quad RS-232-C line driver. (The other three sections of the line driver are not used.)

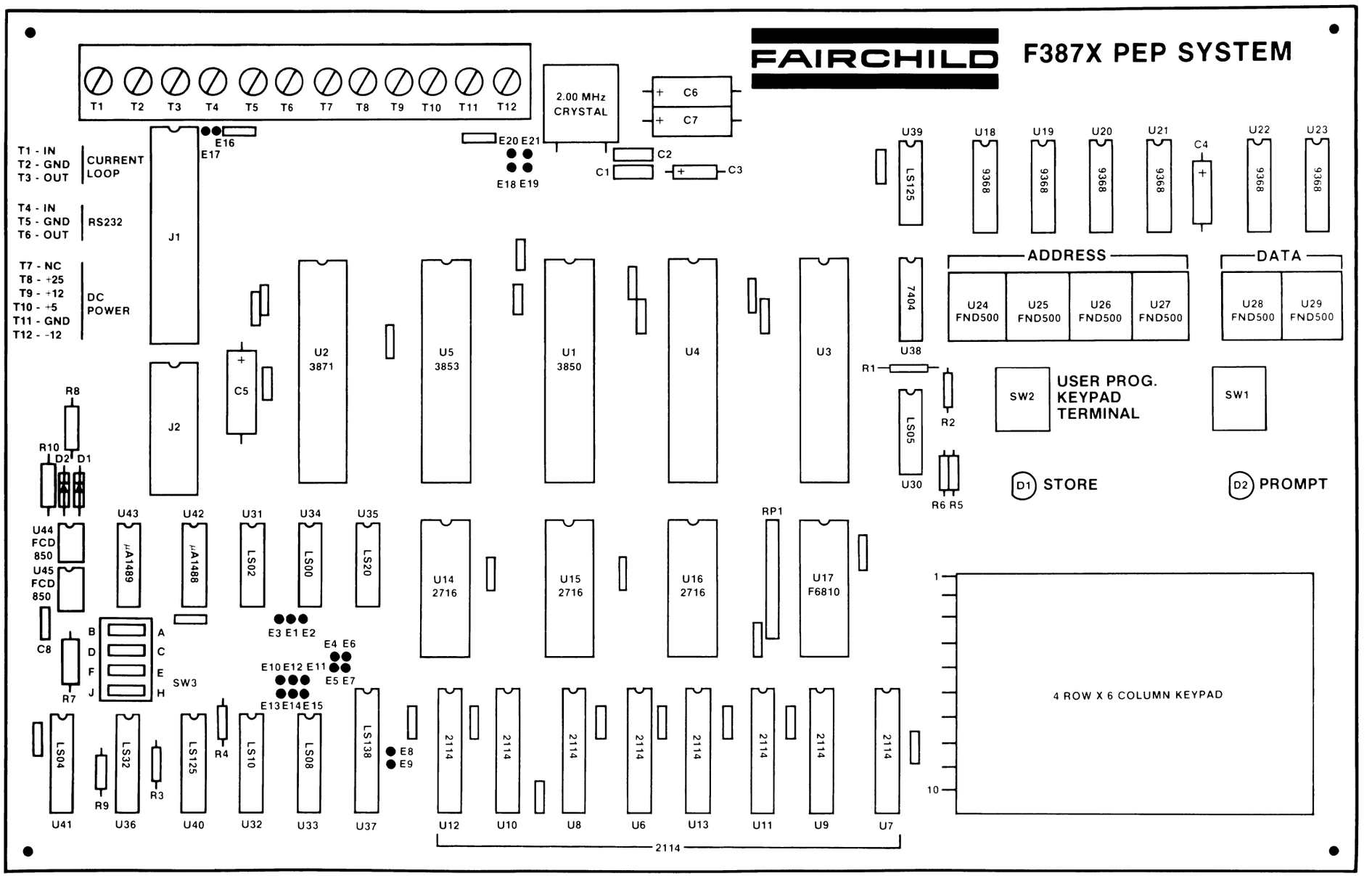
The μA1488 converts the positive logic TTL data being transmitted to a negative logic output signal, which meets the electrical interface specifications of the EIA RS-232-C standard for digital data communications. This output signal emerges at terminal T6 of the terminal strip, situated at the upper left corner of the PC board.

Serial data received by the PEP System enters via terminal T4 of the terminal group. One section of a μA1489 (U43) quad line receiver translates the received RS-232-C negative logic data to TTL-compatible positive logic levels. (The other three sections of the line receiver are not used). At TTL levels, the U41 inverts the positive logic input data and presents the data to pin 2 of U4, where it can be read by the CPU.

Terminal T5 is a common ground return for both the transmit and receive lines of the RS-232-C interface.

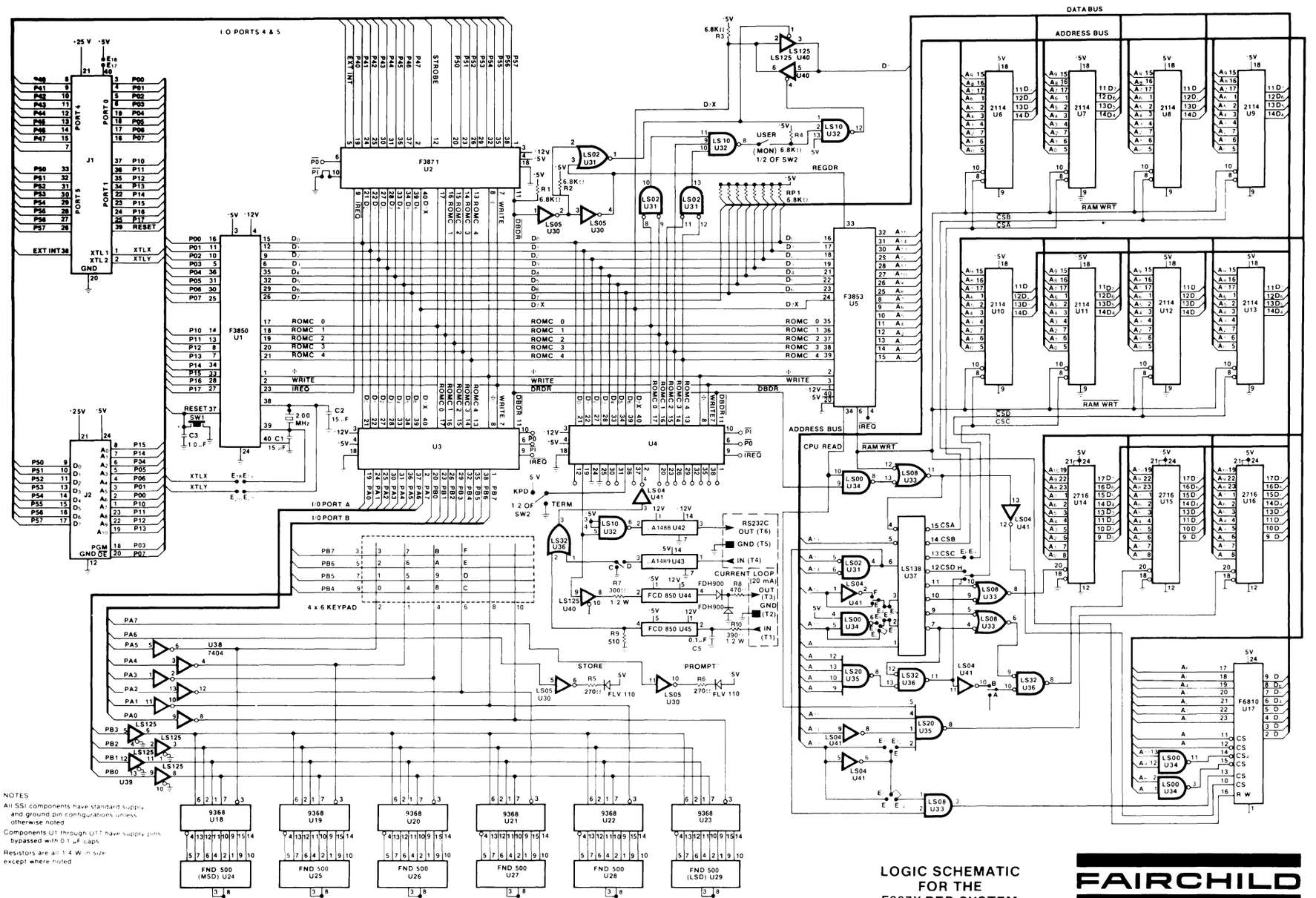
Appendix A

PEP System Component Diagram



Appendix B

PEP System Logic Schematic



LOGIC SCHEMATIC FOR THE F387X PEP SYSTEM

FAIRCHILD

Appendix C

F387X PEP System Parts List

Component	Quantity	Part Number	Description
U1	1	3850PC	F8 CPU
U2	1	3871EPC	F8 PIO
U3	0	—	—
U4	1	3856APC (SL90113)	F8 PSU (KDBUG)
U5	1	3853PC	F8 SMI
U6-U13	4	2114PC	Static RAM (1K x 4, 450 ns)
U14-U16	0	2716 (supplied by user)	2K x 8 EPROM
U17	1	F6810P	Static RAM (128 x 8)
U18-U23	6	9368PC	Decoder/Latch/Driver
U24-U29	6	FND 500	7-Segment LED Display
U30	1	74LS05PC	Hex Inverter (O.C.)
U31	1	74LS02PC	Quad 2-Input NOR
U32	1	74LS10PC	Triple 3-Input NAND
U33	1	74LS08PC	Quad 2-Input AND
U34	1	74LS00PC	Quad 2-Input NAND
U35	1	74LS20PC	Dual 4-Input NAND
U36	1	74LS32PC	Quad 2-Input OR
U37	1	74LS138PC	One-of-8 Decoder
U38	1	7404PC	Hex Inverter
U39-U40	2	74LS125PC	Quad 3-State Buffer
U41	1	74LS04PC	Hex Inverter
U42	1	uA1488PC	Quad RS-232-C Line Driver
U43	1	uA1489PC	Quad RS-232-C Line Receiver
U44-U45	2	FCD 850	Darlington Opto Isolator
D1-D2	2	FDH 900	Diodes
Q1-Q2	2	FLV 110	Red LED Lamps
RP1	1	Bourns 4310R-101-682	6.8K ohm, 10 lead SIP resistor pack
R1-R4	4	AB CB6821 (RC07GF 682K)	6.8K ohm, ¼ W, 10% resistor
R5-R6	2	AB CB2711 (RC07GF 271K)	270 ohm, ¼ W, 10% resistor
R7	1	AB EB3011 (RC20GF 301K)	300 ohm, ½ W, 10% resistor
R8	1	AB EB4711 (RC20GF 471K)	470 ohm, ½ W, 10% resistor
R9	1	AB CB5111 (RC07GF 511K)	510 ohm, ¼ W, 10% resistor
R10	1	AB EB3911 (RC20GF 391K)	390 ohm, ½ W, 10% resistor
C1-C2	2	AVX MC505A150K or AVX 3439-050A-150K	15 pF, 50V capacitor
C3	1	Sprague 150D-105X-0015A2	1.0 uF, 15V, tubular, tantalum electrolytic capacitor
C4-C7	4	Sprague 150D-226X-0035R2	22 uF, 35V, tubular, tantalum electrolytic capacitor
C8-C36	29	AVX Skycaps 3430-050E-104M	0.1 uF, 50V, 20% epoxy coated ceramic capacitor
XTAL	1	CTS Knights MP 020	2.000 MHz crystal, fund. AT Cut
SW1	1	C&K 8125V3	SPDT pushbutton switch
SW2	1	C&K7211MD9V30B	DPDT toggle switch

Component	Quantity	Part Number	Description
SW3	1	Grayhill 76C04	Quad SPDT DIP switch
KBD	1	T.I. 11KS-131	Keypad, 4 rows x 6 columns
—	1	Cellotape PEPSKB001	Overlay for keypad
J1	1	RN ICN 406-S2G	40 pin burn-in socket, 20 mil gold
J2	1	RN ICN 246-S2G	24 pin burn-in socket, 20 mil gold
S1-S5	5	RN ICL 406-S7T or Augat 240-AG-39D	40 pin socket
S6-S13	8	RN ICL 183-S6T or Augat 218-AG-39D	18 pin socket
S14-S17	4	RN ICN 246-S4T or Augat 224-AG-39D	24 pin socket
CABLE	1	CA D40-P02-24-1-TT-015	40 pin emulation cable
—	1	RDI (Reed) 6PCV-12-000	12 connector terminal strip
PCB	1	—	Printed circuit board
—	6	H.H. Smith #2193	Rubber support feet
—	6	Machine screw, #6-32 x ¼" Straight slot, binder head	
—	6	Hex nut, #6-32, standard pattern	

Appendix D

F8 Microprocessor Family Block Diagrams

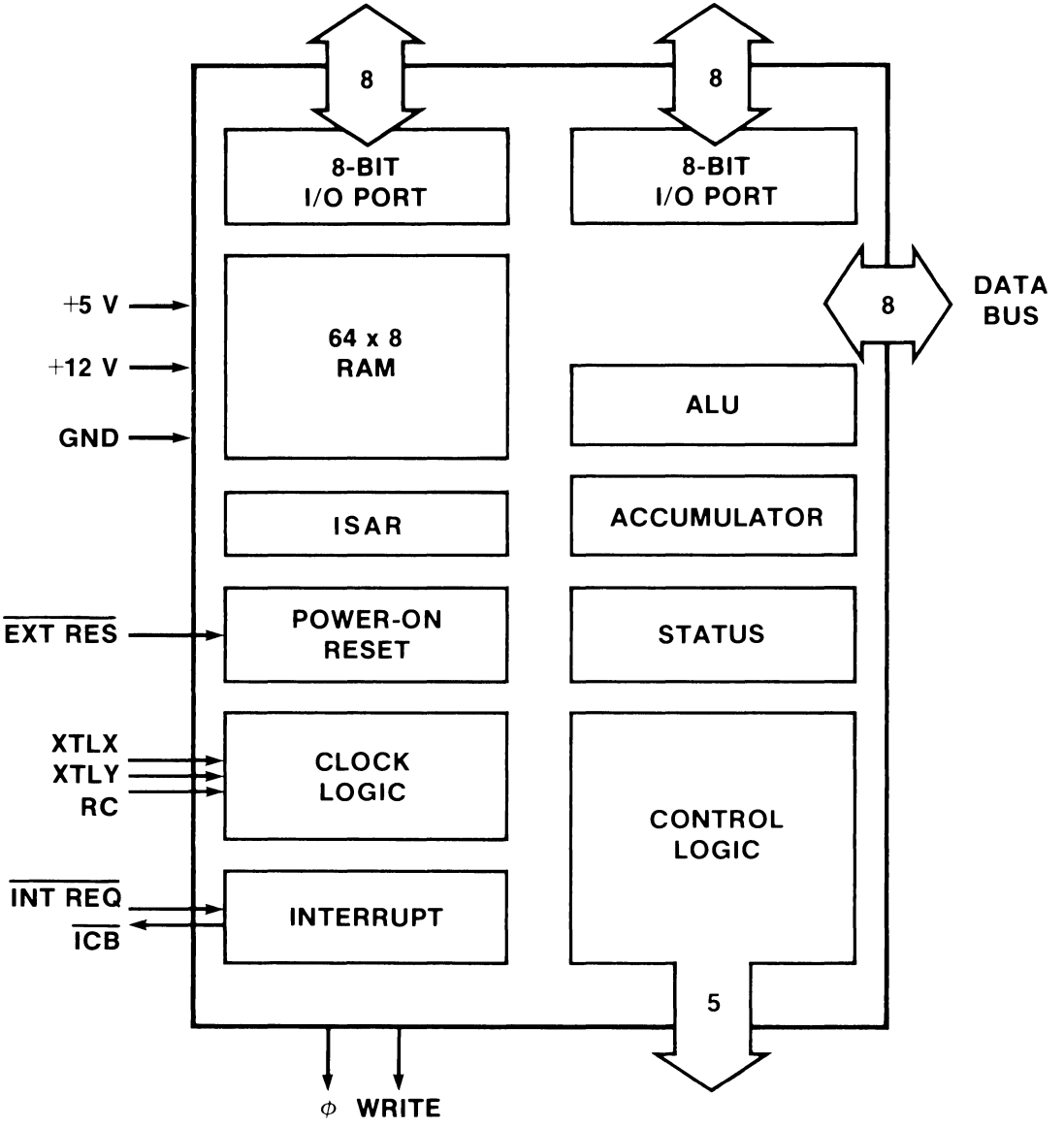


Figure D-1 F3850 Central Processing Unit (CPU)

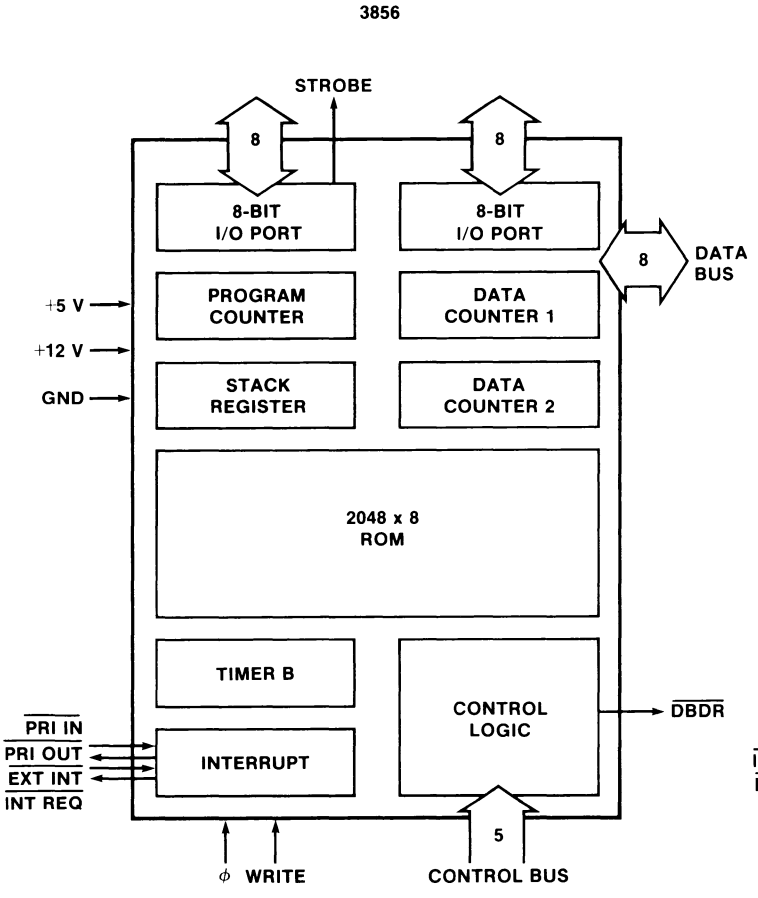


Figure D-2 F3856 Program Storage Unit (PSU)

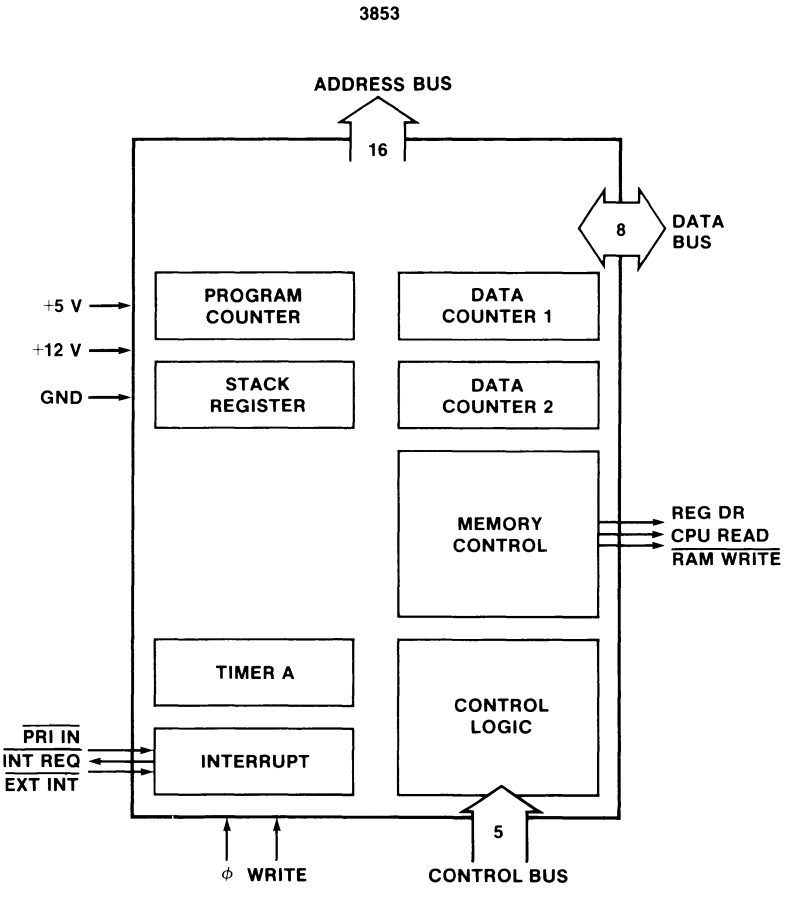


Figure D-3 F3853 Static Memory Interface (SMI)

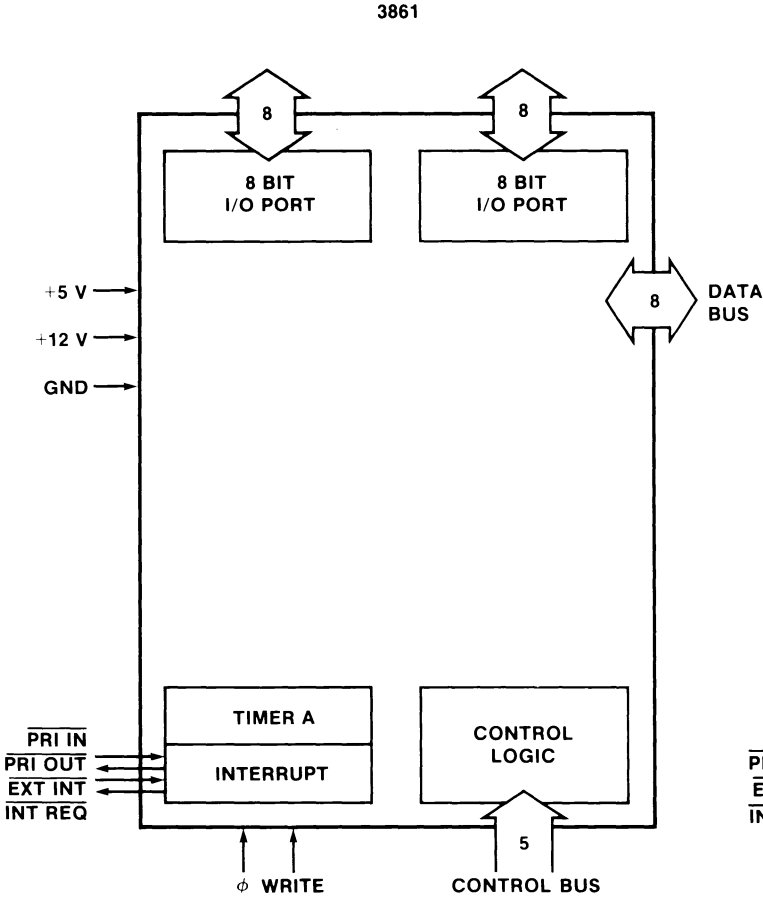


Figure D-4 F3861 Peripheral I/O Device (PIO)

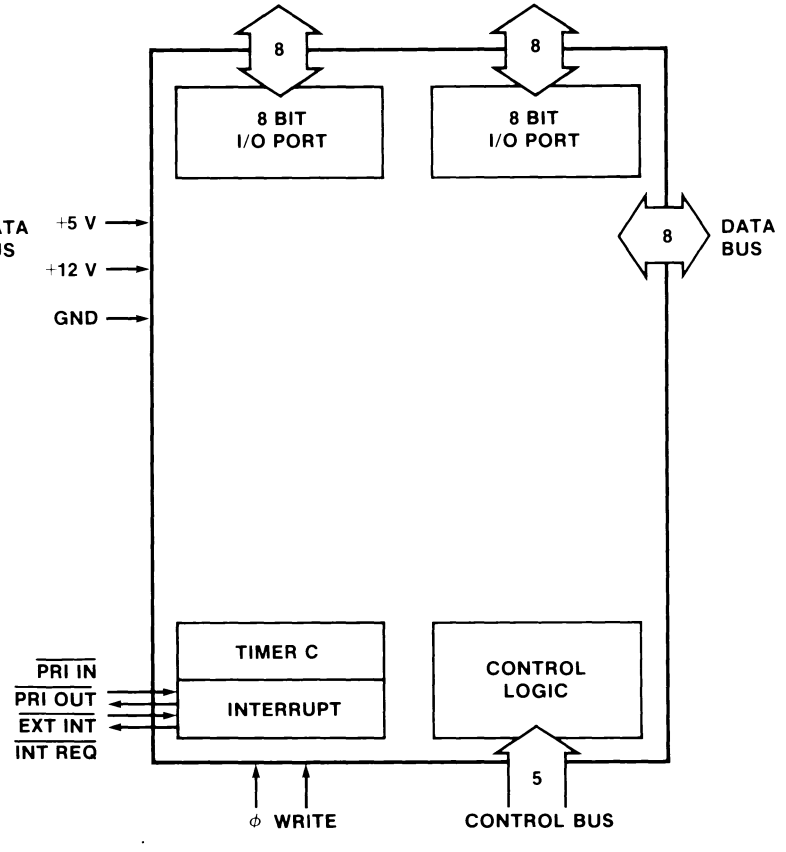


Figure D-5 F3871 Peripheral I/O Device (PIO)

FAIRCHILD

Fairchild cannot assume responsibility for use of any circuitry described other than circuitry embodied in a Fairchild product.
Manufactured under one of the following U.S. Patents: 2981877, 3015048, 3064167, 3108359, 3117260; other patents pending.
No other circuit patent licenses are implied.
Fairchild reserves the right to make changes in the circuitry or specifications at any time without notice.
Printed in U.S.A./332-12-0007-059/5K May 1979

FAIRCHILD
A Schlumberger Company

PEP User Guide
PEPBUG User Guide
SL32131 Addendum

May 1984

Microprocessor Product

PORT ADDRESSES

The PEP 38 board supplied with this addendum uses port addresses other than those listed in the user guides. In every instance, change port 28 to port 24 and port 29 to port 25.

MEMORY STARTING AND ENDING ADDRESSES

It is no longer necessary to set the memory starting address or the EPROM starting and ending addresses. The routine always programs the memory from location H'00' to H'FF' into the EPROM or the F38E70. However, memory location H'2BDE' should be loaded with H'00' for no-blank check, or with H'FF' for blank check.

The starting address for the routine is H'8741' for a 2716-type EPROM and H'8745' for an F38E70 microcomputer. Port 0 must be set to H'FF' and port 1 to H'00' before the programming voltage (precisely +24 Vdc) is applied.

Note that before the programming voltage is applied, all ports of the F38E70 output a low signal; port 0 may give a false reading of H'XX' even though the port is actually outputting H'00'.

PROGRAMMING PROCEDURE FOR THE 2716-TYPE EPROM AND THE F38E70 MICROCOMPUTER

Caution

Failure to follow this procedure in the given order may damage the F38E70 microcomputer.

1. Ensure that the V_{CC} jumper for the F38E70 socket (J1) on the PEP 38 board is in place (i.e., linking the two small pads close to pin 40 of the socket).

2. Connect a V_{PP} power supply to the PEP 38 board and adjust it for the voltage called for on the F38E70 Data Sheet.

3. Turn off the V_{PP} power supply. Ensure that the power supply output is 0 V before proceeding to the next step. Turn off V_{CC}.

4. Insert the F38E70 into socket J1 (socket J2 if programming a 2716-type EPROM).

Fairchild Camera and Instrument Corporation, Microprocessor Division
450 National Avenue, Mt. View, California 94042 (415) 962-3882

©1983 Fairchild Printed in U.S.A. April 1983
Fairchild reserves the right to make changes in circuitry or specifications at any time without notice.

5. If the data to be programmed is in an EPROM, insert the EPROM into the socket provided on the PEP board (refer to the user guides for location of proper socket).

6. If the data to be programmed is not in an EPROM, transfer it into RAM.

7. Configure the DIP switch (SW3) so that the data to be programmed into the F38E70 microcomputer or the 2716-type EPROM starts at location H'00'.

8. In memory location H'2BDE', store H'00' for no-blank check or H'FF' for blank check.

9. Output H'FF' to port 0 and H'00' to port 1.

10. Turn on V_{pp}.

11. For an F38E70, enter "G 8745" if a terminal is used or press the "E70" key if the PEP 38 board keypad is used.

    For a 2716-type EPROM, enter "G 8741" at the terminal or press the "2716" key on the PEP 38 board keypad.

12. After about 2 minutes, the monitor outputs the prompt character

    ?

    Examine register 0 to determine if the programming was successful:

    R0 = H'00' Programmed and checked okay
    R0 = H'01' Programming error
    R0 = H'02' Failed blank check

13. Turn off the 24 V power supply first. Ensure that the power supply output is 0 V before turning off the PEP 38 board power source.

DISPX ROUTINE

The "DISPX" routine is deleted from the version of PEPBUG in the SL32131 ROM.

BAUD RATE

The RS-232C port on the PEP 38 board also operates at 2400 baud. To set the port for 2400 baud, enter F6 in memory location H'2BC3'; complete communication interface according to the user guide.

Fairchild cannot assume responsibility for use of any circuitry described other than circuitry embodied in a Fairchild product. No other circuit patent licenses are implied.

Manufactured under one of the following U.S. Patents: 2981877, 3015048, 3064167, 3108359, 3117260; other patents pending.