Experiment #7

Introduction to Flight86 Microprocessor Trainer and Application Board

7.0 Objectives:

The objective of this experiment is to introduce the Flight86 Microprocessor training kit and application board.

In this experiment, you will do the following:

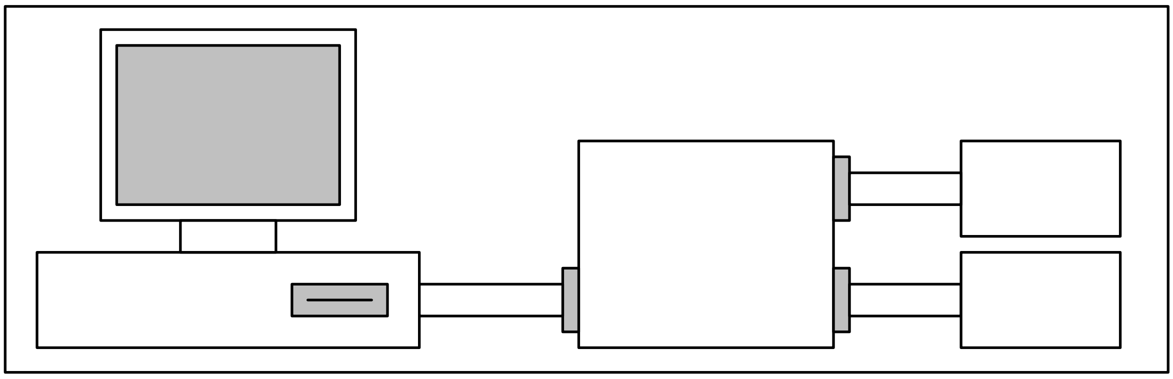
- Study the hardware specifications of the training and application boards
- Learn monitor commands to communicate with the Flight86 trainer
- Assemble, download, and test a program on the trainer board

7.1 Equipment and Software

- Flight86 Trainer and Application Board
- PC with Flight86 Monitor program

7.2 Introduction:

The Flight86 trainer system together with an application board can be used to perform interesting experiments using the 8086 on-board microprocessor. The Flight86 trainer system can be connected to a PC (through its serial port) which allows code to be assembled and debugged in a supportive software environment before being downloaded into the RAM on the board. The block diagram below shows such a setup:



Once the code is downloaded, it can be executed and tested in a system which is accessible to the user. Data may be modified on the board and the change in results can be viewed on the PC display. A monitor program stored on the board in an EEPROM is the software that allows communication between the PC and the trainer system. The monitor program allows code to be downloaded from an Intel Hex file into the controller RAM. The monitor code also enables development activity such as register and memory management and program execution to take place.

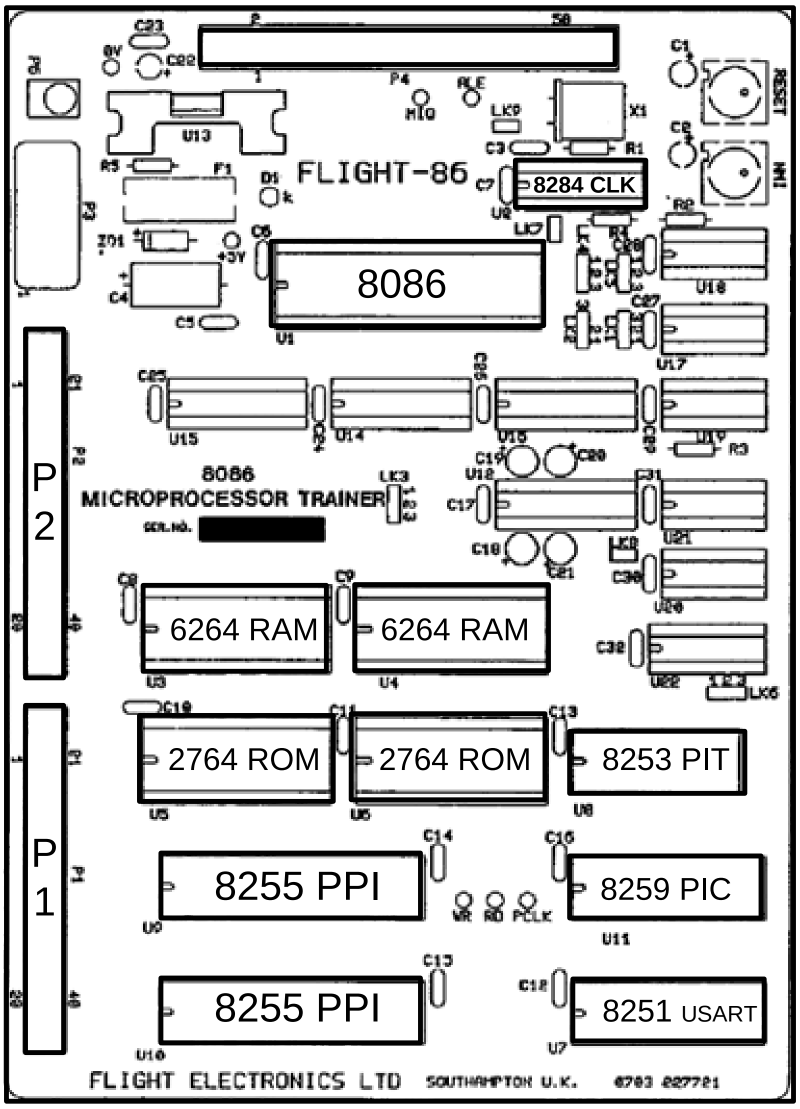
The basic components of the trainer system are described below:

Microprocessor	
CPU 8086	Operating in Min. mode
CGD 8284A	Clock Generator Device Oscillator source: 14.7456 MHz; CPU Clock, CLK: 4.9152 MHz Peripheral clock, PCLK: 2.4576 MHZ
Memory	
EPROM (2) 2764	16k byte (expandable to 64k byte)
RAM (2) 6264	16k byte (expandable to 64k byte)
On-board Peripherals	
PPI (2) 8255A	Programmable Peripheral Interface providing four 8-bit parallel ports with handshake lines
PIT 8253	Programmable Interval Timer providing three 16-bit counter/timer channels
USART 8251A	Universal Synchronous/Asynchronous Receiver/Transmitter
PIC 8259A	Programmable Interrupt Controller providing 8 levels of priority for above devices

Register	Address	Register	Address
8255 PPI (U10) connected to P1		8255 PPI (U9) connected to P2	
Port A	00h	Port A	01h
Port B	02h	Port B	03h
Port C	04h	Port C	05h
Control	06h	Control	07h
8253 PIT (U8)		8259 PIC (U11)	
Count 0	08h	ICW1, OCW2-3	10h
Count 0	0Ah	ICW2-4, OCW1	12h
Count 0	0Ch	8251 USART U7	
Mode Word	0Eh	Data	18h
		Status/Control	1Ah

Table: I/O Map

Layout of the Flight86 Trainer system:

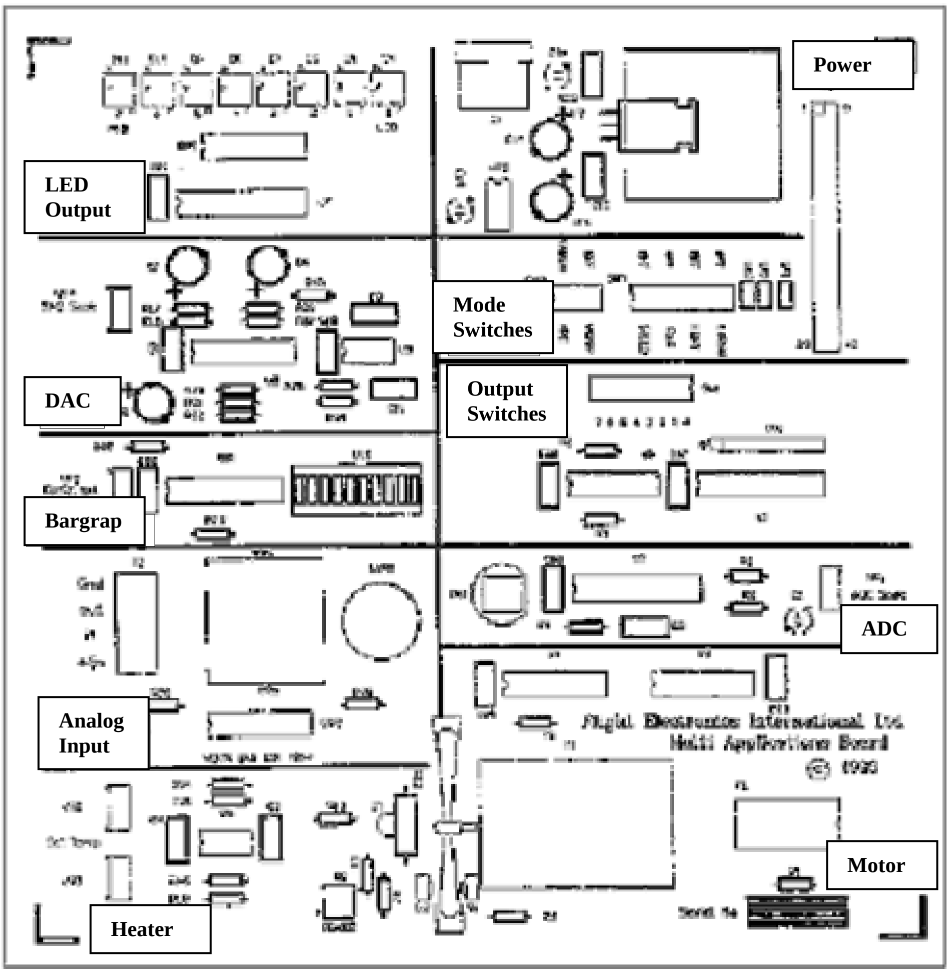


7.2.1 Application Board

The application board is useful for microprocessor interfacing from simple switch and lamp input/output through to more complex closed-loop and open-loop control systems. This board includes the following components:

- Eight digital switches
- Temperature sensor
- Optical speed/position sensor
- Light sensor
- Potentiometer
- External analogue input
- DC motor
- Eight LED's
- Bargraph
- Heater
- Analogue output

Layout of the Application Board



7.2.2 Host (PC) to Controller Board (Flight 86 Trainer) Communication

To establish Host to Controller (Trainer) Board Communication follow this set-up procedure:

1. Turn on Host PC. Ensure you have the DOS prompt showing C:\EE390\FLIGHT86>
2. Turn OFF Flight86 board power supply.
3. Connect the serial lead (cable) between the serial port (COM1) on the PC and socket P3 on the Flight86 board.
4. Turn ON Flight86 board DC power supply. [Note: the board indicates that power is actually applied by illuminating the green LED, D1.
5. Type flight86 at the DOS prompt C:\EE390\FLIGHT86>, i.e., C:\ EE390\FLIGHT86>flight86.
6. After a few seconds you should see the following messages:

```
FLIGHT86 Controller Board. Host Program Version 2.0.
Press ? and Enter for help – Waiting for controller board response …
ROM found at F000:C000 to F000:FFFF Flight Monitor ROM version 2.0
RAM found at 0000:0000 to 0000:FFFF
-
```

7. The “-“ is the host prompt.
8. You have control over the controller board once this prompt is displayed.

If you do not see the prompt, you do not have communications with the controller board.

If the above message stopped at “Waiting for controller board response …” turn the Controller board power supply OFF, wait a few seconds, and turn it ON again. You should then see the “Controller Reset” message followed by the memory test messages.

7.2.3 Host Commands

The command line is executed immediately after the Enter or Return key is pressed. Before this it may be edited using the DEL key.

All data must be specified in hexadecimal, although leading zeros may be omitted. Spaces are ignored for flexible entry, although the first character of a line must be a valid command letter.

The command line syntax uses squared brackets, [ ], to indicate parameter options. The pipe symbol, |, is used to indicate a choice of parameters, one of which must be used.

If an invalid parameter is entered, the full command syntax and help line is displayed to assist.

Command	Key	Parameters	Description
Escape	Esc		Press the Escape button to stop the current command
Reset	X		Resets the training board
Help	?	[*command letter*]	Help on the command
Quit	Q		Terminates running of the board software and returns control to the operating system
Register	R	[*register*]	Allows the user to display or change the content of a register
Memory	M	[W][*segment*:] *address*1 [*address*2]	Allows the user to display or edit one or more memory locations
Assembly	A	[[*segment*:] *address*]	Allows the user to write 8086 assembly code directly into the training board
Disassemble	Z	[[V] [*segment*:] *address*1 [*address*2]]	Reverse assembles the contents of memory
Go	G	[[*segment*:] *address*]	Allows the user to execute code that has been downloaded into RAM
Breakpoint	B	? | R | S [*segment*:] *address*	Allows the user to Display/Clear/Set break points inside his code
Single step	S	[R][[*segment*:] *address*]	Allows the user to step through code one instruction at a time
Download	:	[*drive*:*path*\] *filename*	Loads an Extended Intel Hex file from disk into the memory of the training board
Upload	U	[*drive*:*path*\] file [seg:] add1 add2	Allows a block of memory to be saved to a disk file in Extended Intel Hex format

The **Assemble** command (A) is used to enter 8086 assembly code commands, and these will be assembled and the bytes stored directly into memory. There is a short delay as the bytes of code are sent to the board.

Note: The origin address for user RAM on the FLIGHT86 system is 0050:0100.

The **Disassemble** command (Z) allows the contents of memory to be reverse assembled to 8086 mnemonic codes. If the V option is specified, then the ASCII codes for the disassembled bytes are displayed alongside each line. For example,

```
- Z V 100 130
    displays the disassembled code between the addresses specified with ASCII codes
```

The **Download** command (D) loads an Extended Intel Hex file from disk and puts it into the appropriate memory locations. The original file may have been generated using an assembler, compiler or the Upload command. For example,

```
- : C:\ EE390\FLIGHT86\program.hex
    downloads file program.hex from drive C subdirectory FLIGHT86
```

The **Upload** command (U) allows a block of memory to be saved to a disk file. The data is saved in the Extended Intel Hex format. If the file extension is not specified, then it defaults ti .HX. For example,

```
- U C:\ EE390\FLIGHT86\program.hex 0050:0100 150
    saves contents of block 0050:0100 to 0050:0150 to file program.hex on drive C
    subdirectory FLIGHT86
```

Note: This command is useful for saving your program to be shown to your lab instructor later while you continue to work on another program. So, the next time you don't have to assemble the program again but simply download it using the Download command ":".

7.2.4 I/O Interfacing using 8255 PPI (Parallel Peripheral Interface)

The FLIGHT86 controller board has two 8255 PPI chips that provide ports for I/O interfacing. To be able to use the 8255 for I/O interfacing, a control byte has to be written to the Control register of the 8255 so that any of the ports A, B or C can be set up as input or output ports. The format of the Control byte is as shown below:

b7	b6	b5	b4	b3	b2	b1	b0
1	Mode bits for port A, and port C_{Upper}		Port A	Port C_{Upper}	Mode bit for port B, and port C_{Lower}	Port B	Port C_{Lower}

Port bit:
0 → output
1 → input

b6	b5	Mode	b2	Mode
0	0	Mode 0 - Simple I/O	0	Mode 0: Simple I/O
0	1	Mode 1 - Strobed I/O	1	Mode 1: Strobed I/O
1	x	Mode 2*- Bidirectional I/O		

* - Only port A can be configured for Mode 2 whereas the other ports (B, and C) can be used either for input or output only.

Example:

b7	b6	b5	b4	b3	b2	b1	b0	Configuration
1	0	0	0	1	0	0	1	Mode 0; ports A and B: **output**; port C: **input**
1	0	0	0	0	0	1	0	Mode 0; ports A and C: **output**; port B: **input**

The Flight86 controller board communicates with the Application board using the 8255 PPI. The addresses of these ports are given below:

Register	Address	Register	Address
8255 PPI (U10) connected to P1		8255 PPI (U9) connected to P2	
Port A	00h	Port A	01h
Port B	02h	Port B	03h
Port C	04h	Port C	05h
Control	06h	Control	07h

Note that the Flight86 controller board has two 8255 devices one connected to port P1 and the other to port P2 through which it can connect to the Application board. The addresses of the 8255 registers depend on which port is used.

Output Port:

The Controller board output port connects to **Port B** on the Applications Board, and the state of the 8 lines will always be displayed on the 8 colored LED’s.

By means of on board mode switches, this port can be used to control the motor (forward and reverse) and/or the heater.

When not in use for these functions, the output port can be used to drive the Digital to Analogue Converter (D/A).

Input Port:

The processor input port connects to **Port A** on the Applications Board, and by selection via mode switches can be used to read the 8 bit DIL switch, or the output of the Analogue to Digital Converter (A/D), or the output of the D/A comparator, and/or the output of the speed sensing infra-red detector.

7.3 Pre-lab:

1. Read all the above sections of this experiment.
2. Read the topic on 8255 PPI from your text book.
3. Read about the IN and OUT instructions from your text book.

7.4 Lab Work:

Part 1: Familiarization with the Flight86 trainer system

The lab instructor will give a demonstration and introduce you to the Flight86 Controller board, the monitor program, and the Application board.

Part 2: Initialization

Before the Flight86 Controller board can communicate with the Application board, the 8255 on the Controller board must be initialized. Telling the 8255 how to perform is known as INITIALIZATION, so the first program we run after power up or reset, must be one which initializes the 8255.

```
MOV AL, 99        ; set up Port A IN, Port B OUT, Port C IN
OUT 07, AL        ; and output this word  to control Port
MOV AL,0          ; data zero
OUT 03,AL         ; output to Port B to ensure all LEDs off
INT 5             ; Return to Monitor program and prompt
```

Note: This program without the last line INT 5 will be required at the start of every program you write for the Flight86 system to ensure the 8255 is set up before doing any thing else.

Procedure

1. Start up the Flight86 board as described in section **1.2 Host (PC) to Controller Board Communication (Flight86 Trainer)** obtaining the sign on message, followed by the '-' prompt. (Ask your lab instructor to show you how).
2. Make sure the ribbon cable connects port P2 on the Flight86 board to the port on the Application board.
3. Turn ON the Application board power supply.
4. Now enter **A 0050:0100** at the '-' prompt. This starts the line assembler at the desired address. The monitor program responds by echoing the address 0050:0100. Now enter the program as shown above. Press the Enter or Return key after each line of code. The monitor program goes to the next address automatically. The screen will look like:

```
0050:0100 MOV AL, 99
0050:0102 OUT 07, AL
0050:0104 MOV AL,0
0050:0106 OUT 03,AL
0050:0108 INT 5
0050:010A
```

5. Press the ESC key at the last address 0050:010A to return to the '-' prompt.
6. Enter **Z 0050:0100** at the prompt. The code just entered will be listed on the screen and can be checked for accuracy.
7. Now enter **G 0050:0100** at the prompt and press the Enter key. The program will now run, initialize the 8255, and return to the prompt. Any LEDs that were lit on the application board will now turn off.
8. Enter **U C:\EE390\FLIGHT86\init.hex 0050:0100 10A** at the prompt to upload the above program from the Controller board memory to a file init.hex on drive C subdirectory FLIGHT86.
9. Now press the RESET button on the Flight86 Controller board.
10. Enter the following command at the prompt to download init.hex to the Controller board memory.

: C:\ EE390\FLIGHT86\init.hex

11. Now enter **G 0050:0100** at the prompt to run the program again.

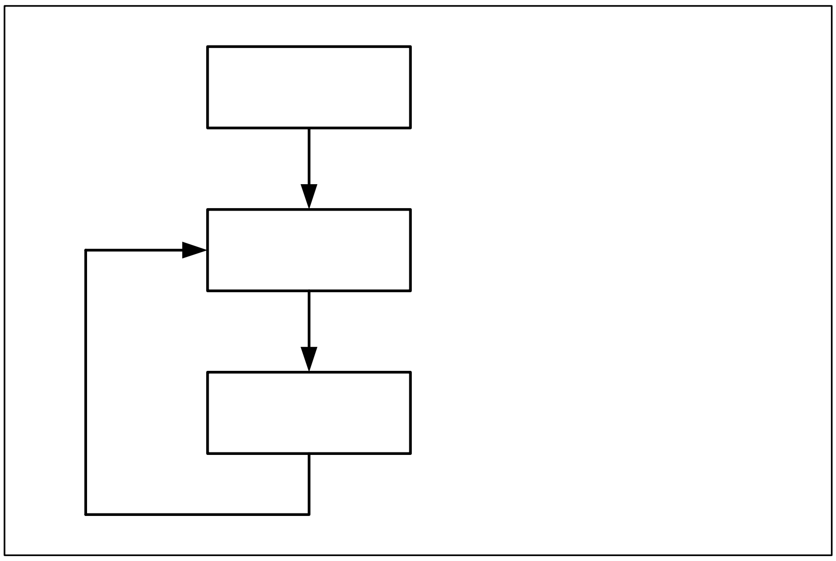
Note that you don't have to enter the whole program again when the controller board is reset for some reason, if you have saved it to a file using the Upload command. This is particularly useful when writing large programs.

Part 3: SWITCHES and LEDs

Write a program that will read the state of the 8 bit DIL switch and output data to the 8 colored LEDs. If a switch is ON, then an LED in the corresponding position will turn ON.

The Application board has an 8 bit DIL switch and 8 colored LEDs. For example, if switch 2 is set, then LED 2 (the green LED on the right side) will turn ON, and so on.

The program logic is illustrated in the following flowchart:



The switches are connected to Port A (address 01), and the LEDs are connected to Port B (address 03) when mode switch SW2A is pushed up to SWITCH position. All the other mode switches must remain in the OFF position.

Procedure

1. Follow steps 1, 2, and 3 of the procedure of Part 2 above.
2. Include the initialization code (without the last line) at the beginning of your program.
3. Assemble your program at 0050:0100.
4. Disassemble your program using the Z command to check for accuracy.
5. Run your program using the G command.
6. Change the settings of the DIL switch and test your program.
7. Check your program again if it does not work.
8. If your program works correctly, then repeat steps 8, 9, 10, and 11 of the procedure of Part 1 above, this time name the file as SW_to_LEDS.hex. The end address to be specified with the Upload command will be the address after the last instruction of your program.
9. Alter the program above so that when a switch is set, the respective LED goes off. Then, repeat steps 3 through 8 above.