FAMILIARISATION WITH MAT385 - ASSIGNMENTS 1 AND 2

INTRODUCTION

A typical microcomputer system comprises a microprocessor (the CPU), a selection of memory circuits and some input-output (I/O) ports. Typically the memory is made up of some read-only memory (ROM) to hold the fixed application program and some random-access (read/write) memory (RAM) for temporary data storage.

During the development of a program, however, it is advantageous if the program is initially stored in RAM so that it may be readily changed if any errors are found when the program is executed. Since a typical program (machine) instruction takes only a few microseconds to execute, it is also helpful and instructive if, during program development, the execution of the program can be controlled. In particular it is possible on the MAT385, to monitor and follow the state of the complete system at each program step so that any errors in intended operation can be identified immediately.

The MAT385 has therefore been designed to enable the user to write and store a program in memory and readily follow and monitor the state of the complete system – the microprocessor, memory and I/O ports – while the program is being executed. In addition, the MAT385 provides a variety of application examples to enable the user to understand how a microprocessor may be used to control the operation of a number of different I/O devices and see in detail exactly how this is being done.

The Basic System

The MAT385 comprises an Intel 8085 CPU – an industry standard 8 bit microprocessor – with some ROM and RAM memory and various I/O ports and associated facilities. The allocation of memory addresses (the memory map) in the system is shown in fig 1.

The system has a *monitor* program which is stored in the ROM. This program provides the user with a number of commands to enable the loading of a program into memory and the monitoring of the system status during the execution of a program. Although a number of different input and display devices may be selected for communicating with the monitor this assignment is concerned with using the 24-pushbutton keypad and the numeric LED (light emitting diode) display.

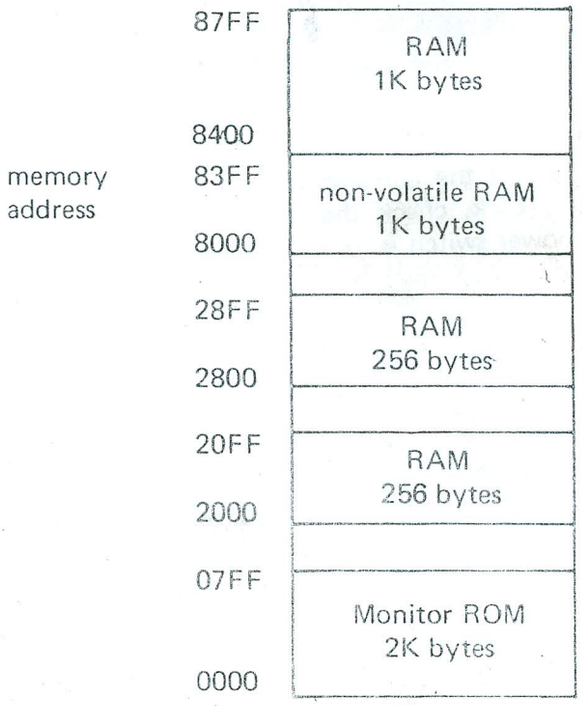


Fig 1 The MAT385 memory map

The keypad and numeric display

Input commands are selected by pressing keys on the 24-pushbutton keypad and the response of the monitor program is displayed on the 6-digit numeric display. This response is either an echo of the particular key pressed or a prompt character to indicate that further inputs are required from the user. All the displayed digits are in hexadecimal code but since each displayed character is built up from seven segments, some of the alphabetic characters are in lower-case. The displayed characters are shown in fig 2.

The six hexadecimal characters are split into two fields: the left-most four digits are the address field and the right-most two digits comprise the data or contents field. Thus a memory address, for example, is specified by up to four hexadecimal digits (16 bits) and the contents of a memory location by two hexadecimal digits (8 bits).

Whenever the monitor expects a command the display shows a dash ('–') in the most significant digit position. Similarly, when the monitor expects a parameter, either address or data, a decimal point is displayed at the right-hand edge of the appropriate field. Each monitor facility will be introduced in the assignment where it is first needed. This first assignment is concerned with the use of three commands. These are described below with detailed examples of their use.

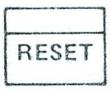
Decimal	Binary	Hex	Display	Decimal	Binary	Hex	Display
0	0000	0	0	8	1000	8	8
1	0001	1	1	9	1001	9	9
2	0010	2	2	10	1010	A	A
3	0011	3	3	11	1011	B	b
4	0100	4	4	12	1100	C	C
5	0101	5	5	13	1101	D	d
6	0110	6	6	14	1110	E	E
7	0111	7	7	15	1111	F	F

Fig 2 Table of Displayed Characters

Switching on

If the MAT385 is switched on, some part of the numeric display above the keyboard will be illuminated. If the display is completely dark, check that the MAT385 is connected to the electricity supply and switch it on. Its power switch is located on the rear panel, and lights up when the power is on.

Monitor Reset



The initial application of power, or pressing the 'RESET' key, automatically generates a signal which forces the monitor program always to restart from the same point. The monitor outputs the message '–80 85' across the address and data fields of the numeric display. The monitor is now ready to accept commands.

The Substitute Memory command

The 'substitute memory' command allows the user to examine the contents of successive memory locations and, if required, to modify the contents of any RAM locations. As will be seen, this is particularly useful for entering a user's program into RAM.

To *examine* the contents of a memory location, the user must first press the 'substitute memory' key. The monitor responds by using a 'decimal point' to indicate that it is waiting for a 4-digit hexadecimal address to be entered. This address is then entered using the 16 hexadecimal keys (0 – F). Note that if an incorrect digit is entered it can be corrected by entering further digits; only the last four digits entered and displayed are taken to be the required address. The address is terminated by depressing the 'next' key – [, / NEXT] – and the monitor responds by displaying the 8-bit (2 hexadecimal digit) contents of the specified location in the data field. The contents of successive memory locations may then be examined simply by repeatedly depressing the 'next' key. Each time this key is depressed the displayed address is incremented by one and the current contents of this new location is displayed in the data field.

At the right of the data field there is also a 'decimal point' displayed, indicating that a new entry can be keyed into the data field. Thus to *modify* the contents of a memory location the user can first examine them, and, when the current contents are displayed, the new data (two hexadecimal digits) can be entered on the keypad. The monitor responds by displaying the new keyed-in contents in the data field but only writes this into the addressed memory location when the 'next' key is depressed. Whenever the contents of a memory location are modified, the monitor checks that the new contents have in fact been changed correctly. If, for example, the addressed memory location was in ROM and therefore could not be altered, the monitor alerts the user by displaying an error message (–Err) in the address field.

Finally, when all the required memory locations have been examined and possibly modified, the 'substitute memory' command is terminated by pressing the execute – [. / EXEC] – key in place of the 'next' key. The monitor responds by displaying a dash (–) and waits for the next command.

Some examples of the use of the 'substitute memory' command are given below.

Substitute Memory – Example 1 Examining memory

The monitor program is stored in read-only memory starting at memory address 0000 (hex). Since this program is in ROM and therefore fixed, it is possible to examine some of the program instructions using the substitute memory command. Follow the sequence of steps below to verify that the contents of the four memory locations starting at address 000F (hex) are as shown:

Key	Address	Data	Comments
SUBST MEM			Start of substitute memory command.
F	000F .		Since required address is location 000F only one key need be depressed.
NEXT	000F	F5.	The fixed contents of memory location 000F are F5 (hex)
NEXT	0010	E1 .	The fixed contents of memory location 0010 are E1 (hex)
NEXT	0011	22 .	The fixed contents of memory location 0011 are 22 (hex)

, NEXT	0012	ED .	The fixed contents of memory location 0012 are ED (hex)
. EXEC	—		Terminates the command and returns to the monitor

Substitute Memory — Example 2 Loading a program

The substitute memory command is used to enter a program into RAM. Since basic programming is covered in later assignments a small program has been pre-written to allow the loading of a program.

Normal user RAM in the MAT385 starts at memory address 2800 (hex) and consequently the following program is entered starting at location 2800. The aim of the program is simply to write the binary pattern FF (hex) into memory location 280F but the program will not be executed for the moment.

Key	*Address*	*Data*	*Comments*
SUBST MEM			Start of substitute memory command.
2	0002		
8	0028		Memory address 2800 (hex) entered
0	0280		
0	2800		
, NEXT	2800	xx .	xx (hex) denotes the current (unpredictable) contents of location 2800
3	2800	03 .	
E	2800	3E .	
, NEXT	2801	xx .	3E (hex) entered into location 2800 (hex) and address incremented by monitor

F	2801	0F .	
F	2801	FF .	
NEXT	2802	xx .	FF entered into location 2801 and address incremented by monitor
3	2802	03 .	
2	2802	32 .	
NEXT	2803	xx .	32 entered into location 2802 and address incremented by monitor
0	2803	00 .	
F	2803	0F .	
NEXT	2804	xx .	0F entered into location 2803 and address incremented by monitor
2	2804	02 .	
8	2804	28 .	
NEXT	2805	xx .	20 entered into location 2804 and address incremented by monitor
C	2805	0C .	
F	2805	CF .	
EXEC			CF entered into location 2805 and control returned to the monitor

Exercise 1

Use the technique outlined in Example 1 to confirm that memory locations 2800 to 2805 now contain the following data:

Address	Data
2800	3E
2801	FF
2802	32
2803	0F
2804	28
2805	CF

Substitute Memory – Example 3

In addition to providing the facility for examining and modifying the contents of specific memory locations, when in the modify mode the substitute memory command checks that any new data entered by the user has in fact been loaded before stepping on to the next consecutive location. This can be demonstrated readily by trying to modify the contents of a memory location which is in ROM:

Key	*Address*	*Data*	*Comments*
SUBST MEM			Start of substitute memory command
0	0000		
NEXT	0000	3E .	Location 0000 currently contains 3E
F	0000	0F .	
F	0000	FF.	New value (FF) entered and displayed by monitor
NEXT	–Err		Error message output by monitor because new contents is not equal to FF since location 0000 is in ROM and therefore cannot be changed, i.e remains at 3E

The GO Command

Once a program has been written, coded and entered using the substitute memory command, the next step is to execute or run the program. All microprocessors contain within them a number of registers one of which – the program counter (PC) – is used to keep track of where within a program the microprocessor has reached during execution. In order to execute a program, therefore, it is necessary to initialise the program counter to contain the memory address which contains the first instruction of the program.

When the 'go' key is pressed, therefore, the monitor responds by displaying the four hex digit current contents of the program counter in the address field along with a decimal point at the right edge of the field to indicate that the contents may be changed if desired. The 'go' command is terminated by pressing the execute key.

Thus to execute a program which is already stored in memory starting at address 2800, for example, the 'go' key is first pressed followed by the four-digit start address 2800 and this is then terminated by pressing the 'execute' key which causes the monitor to run the program. Once a program is running the monitor only regains control of the system either if the reset key is pressed or if certain instructions are executed. This will be described in a later assignment. Some examples of the use of the 'go' command are now given.

Go Command – Example 1

The aim of the program which was loaded into memory in Example 2 using the 'substitute memory' command was to write the binary pattern FF (hex) into memory location 280F. It is now possible to run this program and because of the last instruction of this program, control will automatically return to the monitor program on completion. You can thus inspect memory location 280F to check that this has in fact happened.

The following command sequence first uses the 'substitute memory' command to check that the program entered in Example 2 is still in memory starting at location 2800, executes this program using the 'go' command, and finally uses the 'substitute memory' command to inspect the contents of memory location 280F:

Key	*Address*	*Data*	*Comments*
SUBST MEM			Substitute memory entered
2	0002 .		
8	0028 .		
0	0280 .		Memory address 2800 entered
0	2800 .		
, NEXT	2800	3E .	
, NEXT	2801	FF .	

, NEXT	2802	32 .	Memory locations 2800–2805 still contain program entered in Example 2
, NEXT	2803	0F .	
, NEXT	2804	28 .	
, NEXT	2805	CF .	
EXEC			Substitute memory terminated
GO	xxxx .	xx	Go command entered
2	0002 .		
8	0028 .		
C	0280 .		Program start address (2800) entered
0	2800 .		
EXEC	80	85	Program execution started
SUBST MEM			Substitute memory entered
2	0002 .		
8	0028 .		Memory address 280F entered
0	0280 .		
F	280F .		
, NEXT	280F	FF .	Contents confirmed
. EXEC			Substitute memory terminated

Note: If this program is to be run again it is first necessary to change the contents of memory location 280F using the 'substitute memory' command.

2.1 Introduction

The majority of machine instructions available with a microprocessor operate on or affect the state of various internal registers which make up the micrprocessor. Before a user can write a program for a particular microprocessor, therefore, it is first necessary to acquire a knowledge of those processor registers which are accessible or are affected by the machine language instruction set. The main registers for the Intel 8085, for example, are as follows:

A	
B	C
D	E
H	L
F	IM
Stack Pointer SP	
Program Counter PC	

The letters have the following meanings:

A	8-bit accumulator
B, C, D, E	Four 8-bit general purpose registers
F	8-bit flags register (modified by ALU operations)
IM	8-bit Interrupt control register
HL	Two 8-bit registers which are normally used to form a 16-bit memory pointer
SP	Stack pointer – a 16-bit memory address which always points to the top of a system stack
PC	Program counter – a 16-bit memory address which always points to the next sequential instruction to be executed.

Further explanation and use of these registers will be given as the various machine instructions are introduced in subsequent assignments. The monitor program within the MAT385 provides commands both for examining the contents of each of the above registers and for modifying their contents during actual program execution. This assignment describes these commands and also gives some examples to illustrate their use.

2.2 The Examine Register Command

The 'examine register' command allows the user to display and, if required, modify the contents of each of the above processor registers. The command is entered by pressing the 'examine register' key. The monitor responds by clearing both the address and data fields and displaying a decimal point at the right hand edge of the address field to indicate to the user it is waiting for a

processor register to be specified. The various register names are denoted by abbreviated legends on the keys of the keypad and their full meaning is as shown in the following table:

Key Code	Register
A	A – register (accumulator)
B	B – register
C	C – register
D	D – register
E	E – register
F	Flags register
H / 8	H – register
L / 9	L – register
SPH / 4	most significant byte of stack pointer
SPL / 5	least significant byte of stack pointer
PCH / 6	most significant byte of program counter
PCL / 7	least significant byte of program counter

Table 2.1 Abbreviated Register Names and Display Sequence

If a register key is pressed, the abbreviated name of the selected register will appear in the address field and the current contents of the register will appear in the data field. A decimal point will also be displayed at the right-hand edge of the data field to indicate that the contents of the selected register may be modified if this is required. The contents of successive registers in the sequence shown in Table 2.1 may then be *examined* by repeatedly depressing the 'next' key. Each time this key is depressed the abbreviated name of the next register in sequence will be displayed in the address field and its current contents in the data field.

To *modify* the contents of a register the same procedure as above is followed except that after the current contents of the register have been displayed, the user enters the required new data (2 hexadecimal digits) on the keypad. The monitor responds by displaying the new keyed-in contents in the data field and writes this into the selected register when the 'next' key is pressed.

Finally, when all the required registers have been examined and possibly modified, the 'examine register' command is terminated either by pressing the 'execute' key in place of the 'next' key or by pressing the 'next' key whilst the contents of register PCL are being displayed. The monitor then responds by displaying a dash (–) and waits for the next command.

Some examples of the use of the 'examine register' command are now given.

Example 2.1 Register Display

The following command sequence uses the 'examine register' command to display the contents of all the 8085 registers in sequence starting with the A-register:

Key	*Address*	*Data*	*Comments*
EXAM REG			Start of Examine Register command
A	A	xx .	xx (hex) denotes the current (unpredictable) contents of the A-register
, NEXT	b	xx .	contents of B-register
, NEXT	C	xx .	contents of C-register
, NEXT	d	xx .	contents of D-register
, NEXT	E	xx .	contents of E-register
, NEXT	F	xx .	contents of Flag register
, NEXT	I	xx .	contents of Interrupt register
, NEXT	H	xx .	contents of H-register
, NEXT	L	xx .	contents of L-register
, NEXT	SPH	xx .	Stack Pointer High
, NEXT	SPL	xx .	Stack Pointer Low
, NEXT	PCH	xx .	Program Counter High
, NEXT	PCL	xx .	Program Counter Low
, NEXT or . EXEC			Terminates the Examine Register Command and monitor awaits next command

Example 2.2 Modifying the Contents of a Register

The following command sequence uses the 'examine register' command to first modify the contents of the A-register to contain F7 (hex) and then to modify the contents of registers D and E to contain 20 (hex) and C8 (hex) respectively.

Key	*Address*	*Data*	*Comments*
EXAM REG			Start of Examine Register Command
A	A	xx .	Current contents of A-register
F	A	0F .	New contents F7 (hex) entered
7	A	F7 .	
, NEXT	b	xx .	Current contents of B and C
, NEXT	c	xx .	
, NEXT	d	xx .	Current contents of D-register
2	d	02 .	New contents of D entered,
0	d	20 .	20 (hex)
, NEXT	E	xx .	Current contents of E-register
C	E	0C .	New contents of E
8	E	C8 .	entered, C8 (hex)
. EXEC			Terminates Examine Register command and monitor awaits next command

Exercise 2.1

First follow the command sequence given in Example 2.1 to verify the new contents of registers A, D and E and then use Example 2.2 to modify the contents of registers B and C to contain E8 (hex) and D3 (hex) respectively. Finally verify the new contents again using the sequence given in Example 2.1.

3 Single Step Command

SINGLE STEP

When a program has been written and loaded into memory using the 'substitute memory' command, under normal operation it is executed using the 'go' command as was described in the previous assignment. If the program contains errors, however, and does not perform the required task, it then becomes necessary to find the erroneous instructions so that they may be corrected. This can often be a particularly time consuming and, without any software aids, a very difficult task.

The MAT385, therefore, incorporates the monitor command 'single step'. This is a software aid which enables a user to examine the state of the complete system as each program instruction is executed. Thus any program instructions which do not produce the required effect may be readily identified. Since the state of the system may be monitored as each single machine instruction is executed, the 'single step' command is particularly useful for investigating the action of individual machine instructions. This command is therefore used extensively during subsequent assignments.

To enter the single step mode the 'single step' key is pressed and the monitor responds by displaying the current contents of the program counter (PCH and PCL) in the address field of the display together with a decimal point at the right-hand edge of the field to indicate this may be changed if required. The data field contains the contents of the memory location pointed to by the program counter.

Once in the single step mode, to step through a program starting at a particular memory address, the user simply enters the four digit start address of the program (this will be displayed in the address field as the new contents of the program counter) and presses the 'next' key. Pressing the 'next' key causes the processor to execute the first instruction but instead of continuing execution, the program is suspended and control returned to the monitor. The monitor responds by displaying the new incremented contents of the program counter in the address field (the address of the next instruction to be executed) and the contents of this address (the instruction code) in the data field.

Since control is now with the monitor, before proceeding the user may, if required, examine the contents of the various processor registers and/or memory locations to verify the correct operation of the previous instruction. This is accomplished by first pressing the 'execute' key to temporarily suspend the single step mode and then pressing either the 'substitute memory' key (introduced in the previous assignment) to examine the contents of a memory location or the 'examine register' key to examine the contents of a specific processor register.

The user may then return to the single step mode by pressing the 'single step' key as before. Subsequent pressing of the 'next' key then continues program execution one instruction at a time. The use of the 'single step' command is illustrated by the following examples.

Example 2.3 Register Data Transfer

The derivation of the following program will be described in detail in the next assignment but it is introduced here to illustrate the use of the 'single step' command. The hexadecimal code for the program and the abbreviated meaning of each instruction (group of bytes) is as follows:

Hexadecimal Code	Action
3E 2F	Load the A-Register with the value 2F (hex)
06 8A	Load the B-Register with the value 8A (hex)
0E E8	Load the C-Register with the value E8 (hex)

The action of the program is to load the three processor registers A, B and C with specified values and consequently a combination of the 'single step' and 'examine register' commands can be used to verify that the specific value has been loaded into the appropriate register as each instruction is executed. The program code must first be loaded into memory starting at address 2800 (hex) using the 'substitute memory' command and then the following command sequence can be used to single step through the program.

Key	*Address*	*Data*	*Comments*
SINGLE STEP	xxxx .	xx	Enter Single Step mode
2	0002 .		
8	0028 .		Start address entered
0	0280 .		
0	2800 .		
, NEXT	2802 .	06	First instruction executed
. EXEC	—		Suspend single step mode
EXAM REG	.		Enter examine register mode
A	A	2F .	2F (hex) entered into A
. EXEC	—		Terminate examine register mode
SINGLE STEP	2802 .	06	Re-enter single step mode
, NEXT	2804 .	0E	Next instruction executed
. EXEC	—		Suspend single step mode
EXAM REG	.		Re-enter examine register mode
B	B	8A	8A (hex) entered into B

EXEC	—		Terminate examine register mode
SINGLE STEP	2804 .	0E	Re-enter single step mode
NEXT	2806 .	CF	Next instruction executed
EXEC	—		Suspend single step mode
EXAM REG			Re-enter examine register mode
C	C	E8	E8 (hex) entered into C
EXEC	—		Terminates examine register mode

Exercise 2.4

The following program loads the seven processor registers A, B, C, D, E, H and L in turn with the following hexadecimal data:

A = EF B = C8 C = 73
D = 26 E = 75 H = 1D
L = B4

Load the program code into memory starting at address 2800 (hex) using the 'substitute memory' command and then use the 'single step' command to step through the program one instruction at a time. Inspect the contents of the appropriate register using the 'examine register' command as each instruction is executed to verify program execution.

Program Code

3E EF 06 C8 0E 73 16 26 1E 75 26 1D 2E B4

2.4 SUMMARY

This assignment has described those registers within the microprocessor which are accessible or affected by the microprocessor instruction set. The 'examine register' and 'single step' commands have been introduced to show how the contents of these registers may be examined and, if necessary, modified during actual program execution.