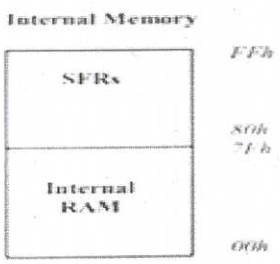
SCHEME OF VALUATION

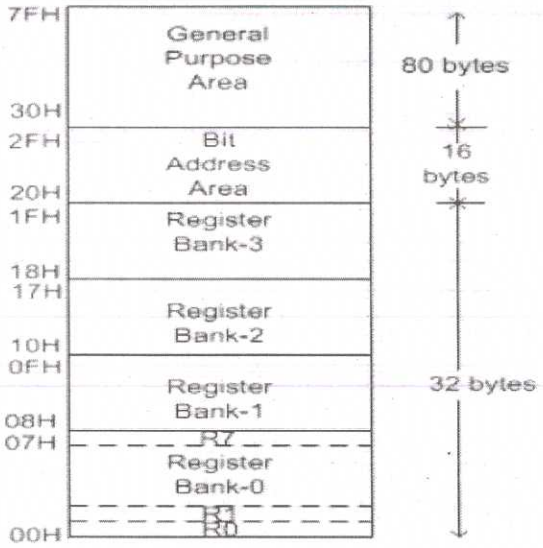
(Scoring Indictors)

Revision: 2015 **Course Code:4043**

Course Title: Microcontroller and Interfacing

Qst. No	Scoring Indicator	Split up score	Sub Total	Total (100)
I	**PART A**			
(1)	ALU is a digital circuit used to perform arithmetic and logic operations.	2		
(2)	Register indirect addressing	2		
(3)	Timer Interrupt& Serial port Interrupt(any two)	2		
(4)	The baud rate is the rate at which information is transferred in a communication channel.	2		
(5)	Interfacing a microcontroller is to connect it with various peripherals to perform various operations to obtain a desired output. .	2	5X 2= 10	10

II	***PART B***			
(1)	8-bit **CPU**. 16-bit Program Counter. 8-bit **Processor** Status **Word** (PSW) 8-bit Stack Pointer. Internal **RAM** of 128bytes. Internal **ROM** of 4Kbytes Special Function Registers (SFRs) of 128 bytes. 32 I/O pins arranged as four 8-bit **ports** (P0 - P3) Two 16-bit timer/counters : T0 and T1.(any six points)	6X1 =6		
(2)	The 8051's on-chip(RAM) memory consists of 256 memory bytes organized as follows: First 128 bytes: 00h to 1Fh, Four Register Banks(32 bytes) Each register bank consists of eight 8-bit registers R0,R1,R2,R3,R4,R5,R6 and R7 20h to 2Fh Bit Addressable RAM(16 bytes) 30 to 7Fh General Purpose RAM(80 bytes) Next 128 bytes: 80h to FFh Special Function Registers ***Or***  	6x1 =6		



(3)

1.ADD A, source

The ADD instruction add the source operand with the destination operand and modify the flags CY,PF AC etc. The destination operand is always in register A while the source operand can be a register accumulator A, immediate data or in memory.

Examples. ADD A,R1

ADD A,#30H

2.MUL AB

The MUL instruction multiply unsigned number in the accumulator with the unsigned number in the B register and store the lower byte of the result in the A reg. and upper byte of the result in the B register. The CY &OF flags are modified.

Example

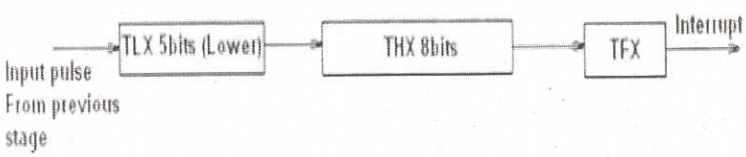
MOV A,#02 H

MOV B,#05H

MUL AB, after execution A contains 0AH&B contains 00H

3.DIV AB

The DIV instruction divide unsigned number in the

(4)	accumulator with the unsigned number in the B register and store the quotient in the A reg. and remainder in the B register. The CY &OF flags are modified MOV A,#02 H MOV B,#05H DIVAB, after execution A contains02H&B contains 01H	3X2 =6			
(5)	1. It finishes the instruction it is executing and saves the address of the next instruction (PC) on the stack and other interrupts of low priority and same priority are disabled. 2. It also saves the current status of all the interrupts internally (i.e., not on the stack). 3. Except for the serial interrupt, the corresponding interrupt flag is cleared. 4. It jumps to a fixed location in memory called the interrupt vector table that holds the address of the interrupt service routine and gets the address of the ISR from the interrupt vector table. 5. It jumps to ISR and starts to execute the interrupt service subroutine until it reaches the last instruction of the subroutine, which is RETI (return from interrupt). 6. Upon executing the RETI instruction, the microcontroller returns to the place where it was interrupted. First, it gets the program counter (PC) address from the stack by popping the top two bytes of the stack into the PC. Then it starts to execute from that address.  	6x1 =6			

6)

In this mode the timer is used as a 13-bit up counter as follows. The lower 5bits of TLX are used for the 13 bit counter. The upper 3 bits of TLX are ignored. When the counter rolls over from all 0's to all 1's TFX flag is set and an interrupt is generated. If TR1/0 bit is 1 and gate bit is 0, the counter continues counting up. If TR1/0 bit is 1 and gate bit is 1, then the operation of the counter is controlled by INTX input.

Fig-2&Exp-4 =6

(MSB) (LSB)

SMO	SM1	SM2	REN	TB8	RB8	TI	RI

SMO &SM1– Mode select bits

SM0	SM1	Mode
0	0	0
0	1	1
1	0	2
1	1	3

SM2 – enable multiprocessor communication in mode 2/3

REN – Receive enable bit

TB8 – the 9th bit that will be transmitted in mode 2/3

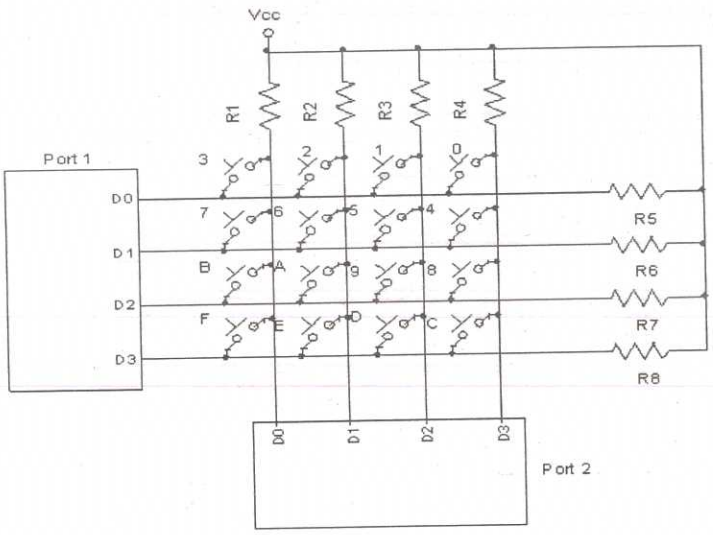
RB8 – in mode 2/3 it is the 9th bit that was received in mode 1 if SM2 =0,

TI – transmit interrupt flag

RI – receive interrupt flag

2+1+3 =6

(7)


Vcc
R1
R2
R3
R4
Port 1
D0
D1
D2
D3
3
2
1
0
7
6
5
4
B
A
9
8
F
E
D
C
R5
R6
R7
R8
D0
D1
D2
D3
Port 2

6

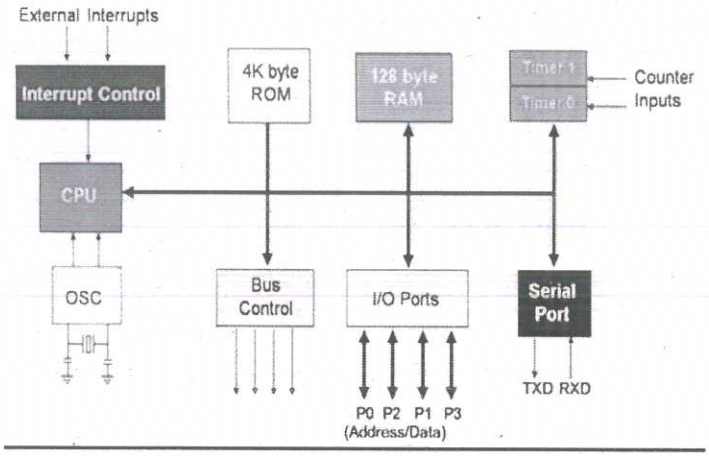
5X 6= 30

30

III

PART C

(a)



Central Processor Unit (CPU) &Oscillator

It reads program written in ROM memory and executes them. It monitors and controls all operations that are performed on the Microcontroller units. The function of the oscillator is to generate clock pulses for synchronizing the internal operations

Interrupt control &Timers

The interrupt control unit will respond to specific interrupts and start execution of functions related to interrupts. It has two 16-bit timers timer0 and timer1

On-chip ROM & On-chip RAM

ROM is used to store program codes. It has 4KB of internal ROM starting from 0000h to 0FFFh.RAM is used to store data and user variables. It has 128bytes of internal RAM starting from 00h to 7Fh

I/O Ports & Serial Port

8051 has four 8-bit i/o ports P0,P1,P2 and P3.Ita has a full duplex serial port

Fig-4+ Exp-4= 8

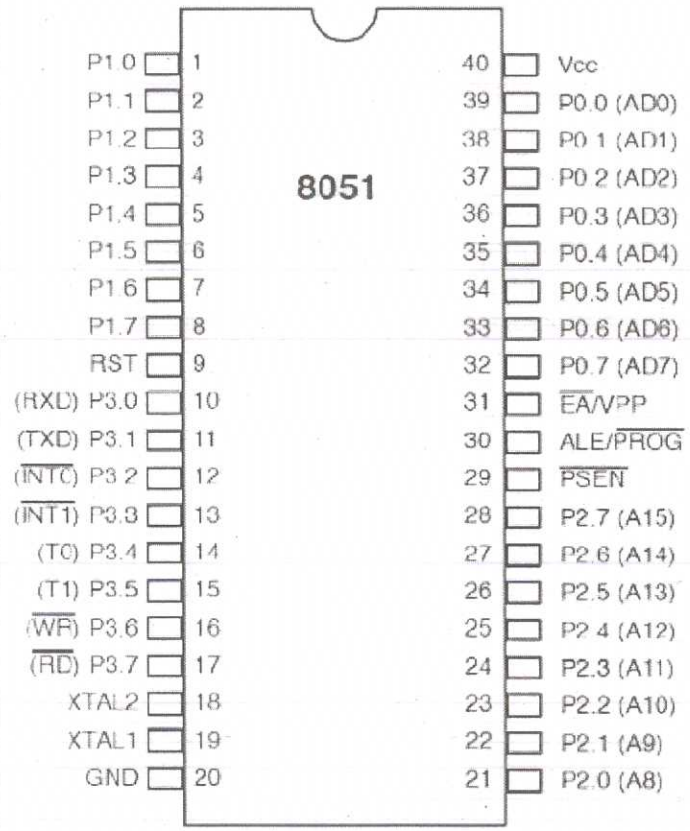
III (b)

Microprocessor	Microcontroller
Microprocessor contains ALU, General purpose registers, stack pointer, program counter, clock timing circuit, interrupt circuit	Microcontroller contains the circuitry of microprocessor, and in addition it has built in ROM, RAM, I/O Devices, Timers/Counters etc
It has many instructions to move data between memory and CPU	It has few instructions to move data between memory and CPU
Few bit handling instruction	It has many bit handling instructions
Less number of pins are multifunctional	More number of pins are multifunctional
Single memory map for data and code	Separate memory map for data and code
Access time for memory and IO are more	Less access time for built in memory and IO.
More flexible in the design point of view (any seven points)	Less flexible since the additional circuits which is residing inside the microcontroller is fixed for a particular microcontroller

7X1=7

8+7 =15

IV (a)



PSEN(Program store enable)-If external ROM is used for storing program, then a logic zero (0) appears on it every time the microcontroller reads a byte from memory.

ALE(Address Latch Enable)-Prior to reading from external memory, the microcontroller puts the lower address byte (A0-A7) on P0 and activates the ALE output '1' to latch the address of multiplexed address data bus.

EA(External program memory enable). By applying logic zero to this pin, P2 and P3 are used for data and address transmission with no regard to whether there is internal memory or not.

Fig-7 + Ex-3 =10

IV (b)

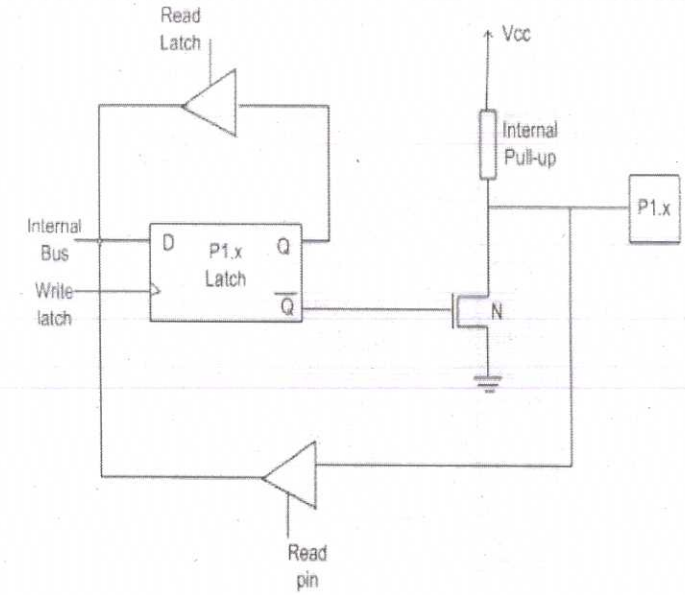


Fig-5

10+5=15

V (a)

1. Immediate Addressing mode

Data is immediately available in the instruction.

For example -ADD A, #77; Adds 77 (decimal) to A and stores in A.ADD A, #4DH; Adds 4D (hexadecimal) to A and stores in A

2. Direct Addressing

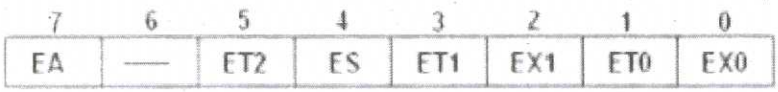
The address of the data is available in the instruction.

For example -MOV A, 088H; Moves content of SFR TCON (address 088H)to A

3. Register Indirect Addressing

The address of data is available in the R0 or R1 registers as specified in the instruction.

For example -MOV A, @R0 moves content of address pointed by R0 to A

	Register Addressing: This way of addressing accesses from the current register bank (R0-R7) and registers A,B,DPTR etc. Data is available in the register specified in the instruction. The register bank is decided by 2 bits of Processor Status Word (PSW). For example-ADD A, R0; Adds content of R0 to A and stores in A	4X2=8		
V (b)	 X0 ⟶ $\overline{INT0}$ interrupt (External) enable bit ET0 ⟶ Timer-0 interrupt enable bit EX1 ⟶ $\overline{INT1}$ interrupt (External) enable bit ET1 ⟶ Timer-1 interrupt enable bit ES ⟶ Serial port interrupt enable bit ET2 ⟶ Timer-2 interrupt enable bit EA ⟶ Enable/Disable all Setting '1' ⟶ Enable the corresponding interrupt Setting '0' ⟶ Disable the corresponding interrupt	3+4=7	8+7 =15	

VI (a)	MOV DPTR,#4500H
	MOV R0,#00H
	MOV R1,#00H			

	MOV R7,#08H
	MOVX A,@DPTR
	Back:RRC A
	JC Next
	INC R0
	SJMP Loop
	Next: INC R1
	Loop: DJNZ R7 Back
	INC DPTR
	MOV A,R1
	MOVX @DPTR,A
	INC DPTR
	MOV A,R0
	MOVX @DPTR,A
	Last: SJMP Last
		Progra-m-7		

VI (b)	Each interrupt source can be programmed to have one of the two priority levels by setting (high priority) or clearing (low priority) a bit in the IP (Interrupt Priority) Register . A low priority interrupt can itself be interrupted by a high priority interrupt, but not by another low priority interrupt. If two interrupts of different priority levels are received			

simultaneously, the request of higher priority level is served. If the requests of the same priority level are received simultaneously, an internal polling sequence determines which request is to be serviced. Thus, within each priority level, there is a second priority level determined by the polling sequence, as follows.

Interrupt	Priority
PX0(External interrupt INT0)	Highest
PT0(Timer0)	.
PX1(External interrupt INT1)	.
PT1(Timer1)	.
PS(Serial port interrupt)	Lowest

Bit Name	U	U	U	PS	PT1	PX1	PT0	PX0

PS - Serial Port Interrupt priority bit

PT1 - Timer 1 interrupt priority

PX1 - External Interrupt INT1 priority

PT0 - Timer 0 Interrupt Priority

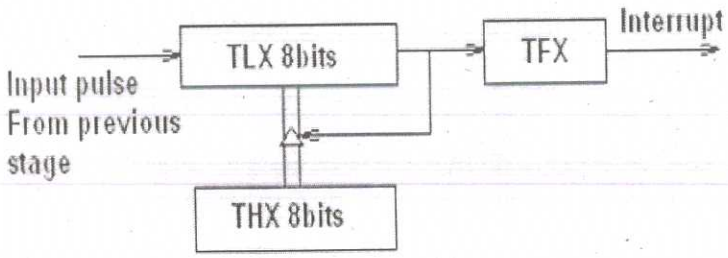
PX0 - External Interrupt INT0 Priority

Expl-4
+Prio-2
+IPR-2

7+8
=15

VII (a)

Timer Mode-2: (Auto-Reload Mode)



This is a 8 bit counter/timer operation. Counting is performed in TLX while THX stores a constant value. In this mode when the timer overflows i.e. TLX becomes FFH, it is fed with the value stored in THX. For example if we load THX with 50H then the timer in mode 2 will count from 50H to FFH. After that 50H is again reloaded. This mode is useful in applications like fixed time sampling.

Fig-3+ Exp-4= 7

VII (b)

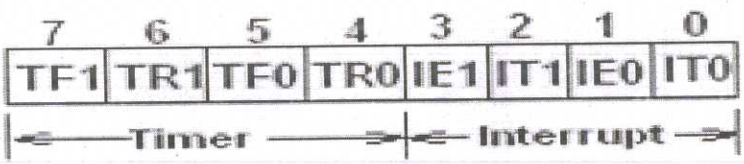


Fig 8.2 TCON Register

TF1: Timer1 overflow flag. It is set when timer rolls from all 1s to 0s.It is cleared when processor vectors to execute ISR located at address 001BH.

TR1: Timer1 run control bit. Set to 1 to start the timer /counter

TF0: Timer0 overflow flag. (Similar to TF1)

VIII (a)	TR0: Timer 0 run control bit. IE1: Interrupt1 edge flag. Set by hardware when an external interrupt edge is detected. It is cleared when interrupt is processed. IE0: Interrupt0 edge flag. (Similar to IE1) IT1: Interrupt1 type control bit. Set/ cleared by software to specify falling edge / low level triggered external interrupt. IT0:Interrupt0 type control bit. (Similar to IT1) **Serial Data Mode 0 (Baud Rate Fixed)** In this mode, the serial port works like a shift register and the data transmission works synchronously with a clock frequency of f_{osc} /12. Serial data is received and transmitted through RXD. 8 bits are transmitted/ received aty a time. Pin TXD outputs the shift clock pulses of frequency f_{osc} /12, which is connected to the external circuitry for synchronization. **Serial Data Mode 1 (baud rate is variable)** In mode-1, the serial port functions as a standard Universal Asynchronous Receiver Transmitter (UART) mode. 10 bits are transmitted through TXD or received through RXD. The 10 bits consist of one start bit (which is usually '0'), 8 data bits (LSB is sent first/received first), and a stop bit (which is usually '1'). Once received, the stop bit goes into RB8 in the special function register SCON. The **baud**	2+6=8	7+8 =15	

rate is variable.

Serial Data Mode 2 (Baud Rate Fixed)

In this mode 11 bits are transmitted through TXD or received through RXD. The various bits are as follows: a start bit (usually '0'), 8 data bits (LSB first), a programmable 9 th (TB8 or RB8)bit and a stop bit (usually '1'). While transmitting, the 9 th data bit (TB8 in SCON) can be assigned the value '0' or '1'. On reception of the data, the 9 th bit goes into RB8 in 'SCON', while the stop bit is ignored. The baud rate is programmable to either 1/32 or 1/64 of the oscillator frequency.

Serial Data Mode 3 (baud rate is variable)

In this mode 11 bits are transmitted through TXD or received through RXD.The various bits are: a start bit (usually '0'), 8 data bits (LSB first), a programmable 9 th bit and a stop bit (usually '1'). Mode-3 is same as mode-2, except the fact that the baud rate in mode-3 is variable (i.e., just as in mode-1).

3X4= 12

VIII (b)

D7	D6	D5	D4	D3	D2	D1	D0
SMOD	x	x	x	GF1	GF0	PD	IDL

SMOD – Serial mode bit

GF1 and GF0 are General purpose flags

PD is the power down bit.

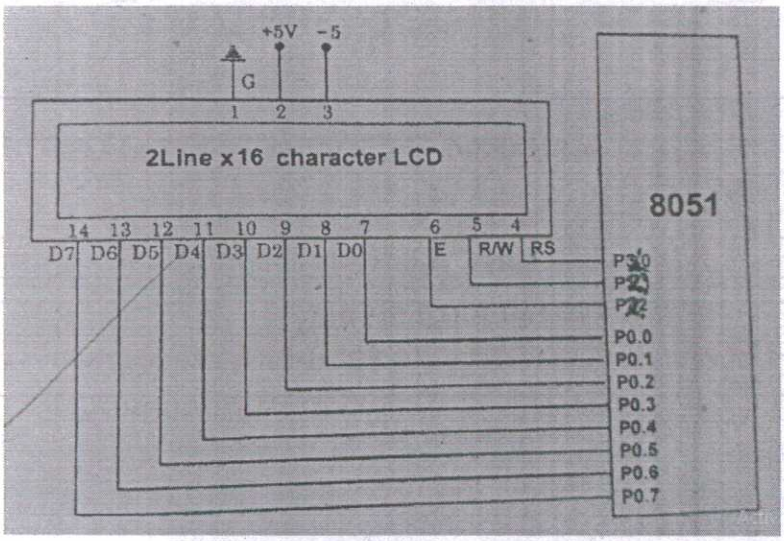
IDL activate the idle mode

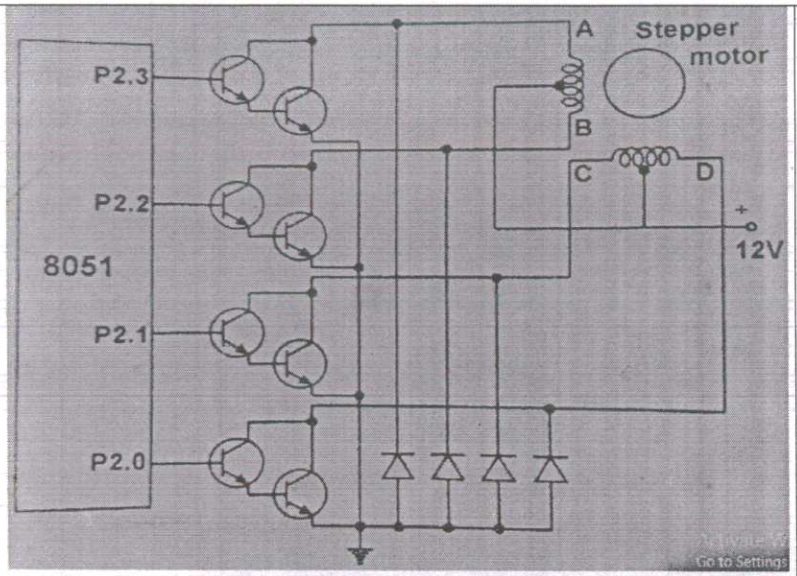
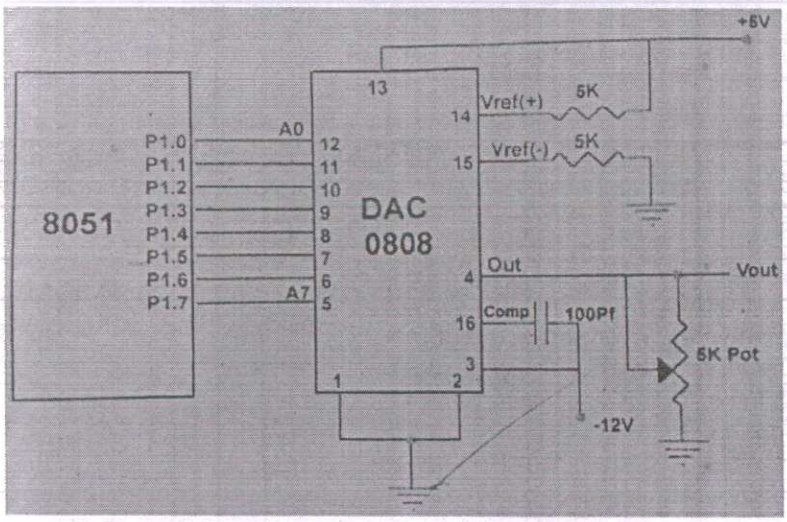
3

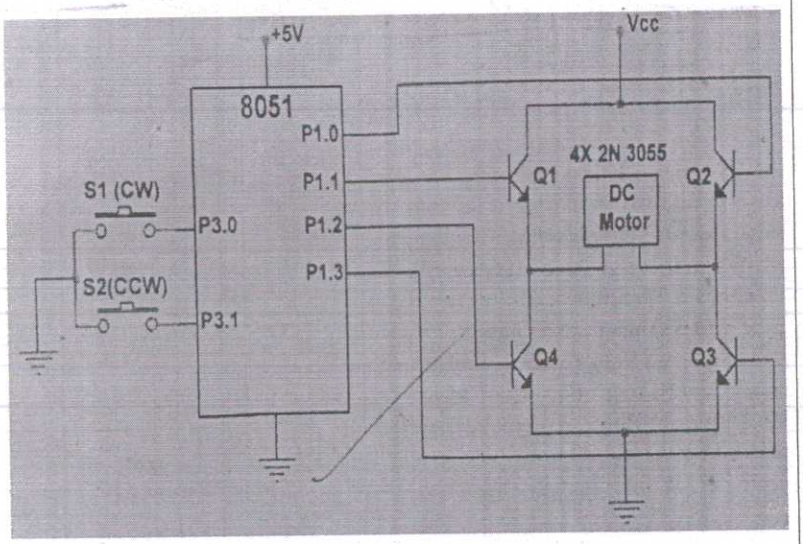
12+ 3= 15

IX (a)

The data bus consists of 4or 8 lines. In the case of 8-bit data bus the lines are referred to as DB0, DB1, DB2, DB3, DB4,

	DB5, DB6 and DB7.There are three control lines E, RS and RW. E-(Enable) A high to low pulse to this pin enable LCD to latch the data present at the data pins. RS(Register select).When RS=1,the data register is selected and when RS=0,the command register is selected RW(Read/Write).When RW=1,the program is reading the LCD and when RW=0,the information on the data bus is being written to the LCD  	3+Fig.5 =8		
IX (b)	The centre tap of the winding is connected to 12V dc supply. The four leads of the stator windings are controlled by four bits of the port2.The four Darlington pairs are used as current drivers to drive the windings.				

X (a)	  Here port 1 of the microcontroller is used to provide digital input to DAC .The Vref(+) and Vref(-) terminals of DAC are connected to +5V and ground through resistors of 5K to derive stable current reference for DAC.The current output from pin 4 of DAC is passed through a 5K pot. and converts as a voltage	Exp.2+ Fig.5= 7 Fig.5+ Exp.2= 7	8+7 =15	
X (b)	The H-bridge circuit is arranged using four 2N3055 transistors Q1, Q2, Q3 and Q4.At any time Q1, Q3 or Q2, Q4 are made ON. Hence current flows from the source to ground through			

	the motor. The direction of motor is dependent on the polarity of the current. Hence by changing the ON transistor pairs, we can control the direction of the motor. The base of the transistors Q1,Q2,Q3 and Q4 are connected to port pins P1.0,P1.1,P1.2 , P1.3 respectively. On controlling the output coming out of these port Pins we can achieve control over motor rotation direction. Two push to ON switches are connected to P3.0 and P3.1 to interrupt the microcontroller to change the direction. 	Exp.3+ Fig.5= 8	7+8 =15	15X4 =60