1.Compare polling versus interrupts in 8051.Explain the different types of interrupts.

The below table explains the comparisons of different features of polling and interrupts.

Specification	Interrupt	Polling
Method	Device notify MCU by sending it an interrupt signals while the MCU is doing another work	MCU continuously monitors devices to determine whether they need services
Response Time	Faster	Slower
Need of MCU Time	Less	Slower
Priority Setting	Yes	No

There are 6 interrupts in the 8051

- **Reset** – when the reset pin is activated, the 8051 will reset all registers and ports and jumps to address location 0000H starting up execution.
- **2 external interrupts** – Hardware external interrupts (INT0 and INT1) at pins 12 & 13 are used to receive interrupt signals from external devices.
- **2 timer interrupts** – They are Timer 0 and Timer 1 which will give out interrupt signal when the timers count to zero.
- **1 serial port interrupt** - It is used for data reception and transmission during serial communication.

2. Explain SCON register with its bit pattern. Explain the steps involved in executing an Interrupt.

The SCON register is an 8-bit register used to program the start bit, stop bit, and data bits of data framing, among other things. The following describes various bits of the SCON register.

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0	SCON.7	Serial port mode specifier
SM1	SCON.6	Serial port mode specifier
SM2	SCON.5	Used for multiprocessor communication. (Make it 0.)
REN	SCON.4	Set/cleared by software to enable/disable reception.
TB8	SCON.3	Not widely used.
RB8	SCON.2	Not widely used.
TI	SCON.1	Transmit interrupt flag. Set by hardware at the beginning of the stop bit in mode 1. Must be cleared by software.
RI	SCON.0	Receive interrupt flag. Set by hardware halfway through the stop bit time in mode 1. Must be cleared by software.

SMO and SMI are D7 and D6 of the SCON register, respectively. These two bits determine the framing of data by specifying the number of bits per character, and the start and stop bits. They take the following combinations.

In the SCON register, when serial mode 1 is chosen, the data framing is 8 bits, 1 stop bit, and 1 start bit, which makes it compatible with the COM port of IBM/compatible PCs. The serial mode 1 allows the baud rate to be variable and is set by Timer 1 of the 8051. In serial mode 1, for each character a total of 10 bits are transferred, where the first bit is the start bit, followed by 8 bits of data, and finally 1 stop bit.

SM2

SM2 is the D5 bit of the SCON register. This bit enables the multiprocessing capability of the 8051 and is beyond the discussion of this chapter. For our applications, we will make SM2 = 0 since we are not using the 8051 in a multiprocessor environment.

REN

The REN (receive enable), bit is D4 of the SCON register. The REN bit is also referred to as SCON.4 since SCON is a bit-addressable register. When the REN bit is high, it allows the 8051 to receive data on the RxD pin of the 8051. If we want the 8051 to both transfer and receive data, REN must be set to 1. By making REN = 0, the receiver is disabled. Making REN — 1 or REN = 0 can be achieved by the instructions “SETB SCON. 4″ and “CLR SCON. 4″, respectively. Notice that these instructions use the bit-addressable features of register SCON. This bit can be used to block any serial data reception and is an extremely important bit in the SCON register.

TBS

TBS (transfer bit 8) is bit D3 of SCON. It is used for serial modes 2 and 3. We make TBS = 0 since it is not used in our applications.

RB8

RB8 (receive bit 8) is bit D2 of the SCON register. In serial mode 1, this bit gets a copy of the stop bit when an 8-bit data is received. This bit (as is the case for TBS) is rarely used anymore. In all our applications we will make RB8 = 0. Like TB8, the RB8 bit is also used in serial modes 2 and 3.

Tl

TI (transmit interrupt) is bit Dl of the SCON register. This is an extremely important flag bit in the SCON register. When the 8051 finishes the transfer of the 8-bit character, it raises the TI flag to indicate that it is ready to transfer another byte. The TI bit is raised at the beginning of the stop bit. We will discuss its role further when programming examples of data transmission are given.

Rl

RI (receive interrupt) is the DO bit of the SCON register. This is another extremely important flag bit in the SCON register. When the 8051 receives data serially via RxD, it gets rid of the start and stop bits and places the byte in the SBUF register. Then it raises the RI flag bit to indicate that a byte has been received and should be picked up before it is lost. RI is raised halfway through the stop bit, and we will soon see how this bit is used in programs for receiving data serially.

3. Write an 8051 C program to display “CMRIT” in a 16x2 LCD.

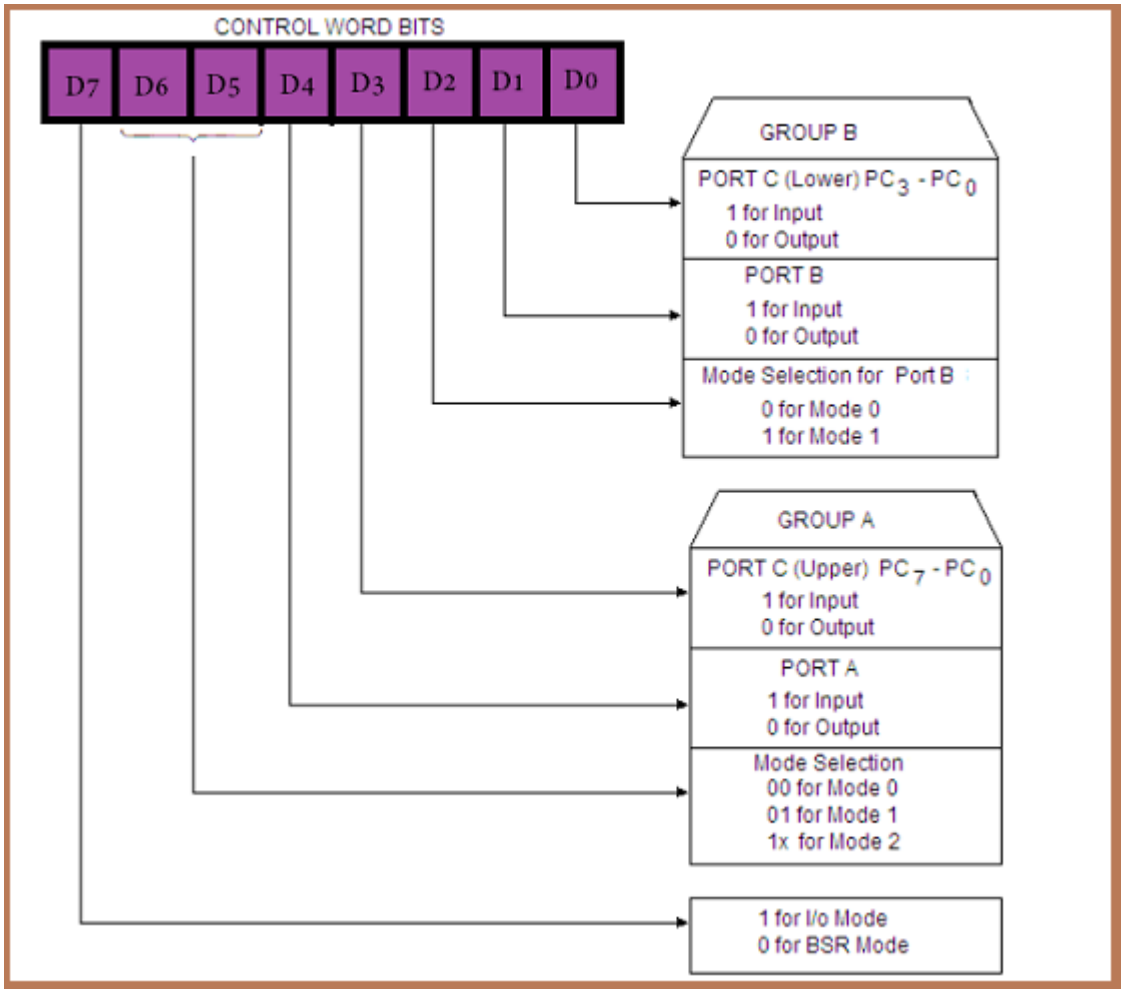
```
#include<reg51.h>
#define display_port P2      //Data pins connected to port 2 on microcontroller
sbit rs = P3^2;  //RS pin connected to pin 2 of port 3
sbit rw = P3^3;  // RW pin connected to pin 3 of port 3
sbit e =  P3^4;  //E pin connected to pin 4 of port 3

void msdelay(unsigned int time)  // Function for creating delay in milliseconds.
{
   unsigned i,j ;
   for(i=0;i<time;i++)
   for(j=0;j<1275;j++);
}
void lcd_cmd(unsigned char command)  //Function to send command instruction to LCD
{
   display_port = command;
   rs= 0;
   rw=0;
   e=1;
   msdelay(1);
```

```
    e=0;
}

void lcd_data(unsigned char disp_data)  //Function to send display data to LCD
{
    display_port = disp_data;
    rs= 1;
    rw=0;
    e=1;
    msdelay(1);
    e=0;
}

 void lcd_init()    //Function to prepare the LCD  and get it ready
{
    lcd_cmd(0x38);  // for using 2 lines and 5X7 matrix of LCD
    msdelay(10);
    lcd_cmd(0x0F);  // turn display ON, cursor blinking
    msdelay(10);
    lcd_cmd(0x01);  //clear screen
    msdelay(10);
    lcd_cmd(0x81);  // bring cursor to position 1 of line 1
    msdelay(10);
}
void main()
{
    unsigned char a[15]="CMRIT";    .
    int l=0;
    lcd_init();
    while(a[l] != '\0') // searching the null terminator in the sentence
```

```
    {
        lcd_data(a[l]);
        l++;
        msdelay(50);
    }
}
```

4. Explain the control word format of 8255 IC. What is the control word for all the ports as output ports?

8255 is a general purpose programmable I/O device designed to interface the CPU with its outside world such as ADC, DAC, keyboard etc. It consists of three 8-bit bidirectional I/O ports i.e. PORT A, PORT B and PORT C. It is possible to assign different ports as input or output functions. The general format for control word of 8255 IC is given below.



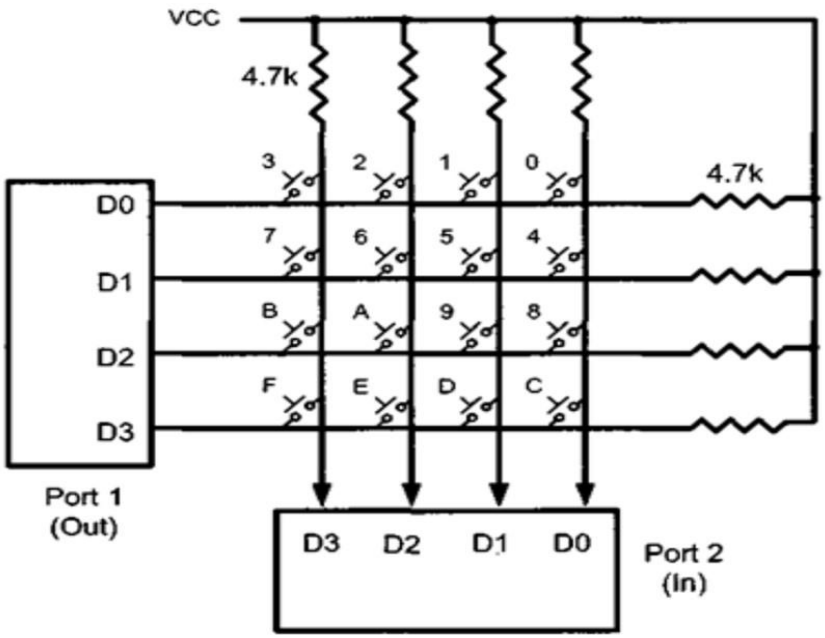
The control word to configure all the ports as output is given below.

Control Word Bits								Control Word	PORT A	PORT C_{Upper}	PORT B	PORT C_{Lower}
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0					
1	0	0	0	0	0	0	0	80 H	Output	Output	Output	Output

5. Explain the architecture and working of Keyboard interfacing with 8051

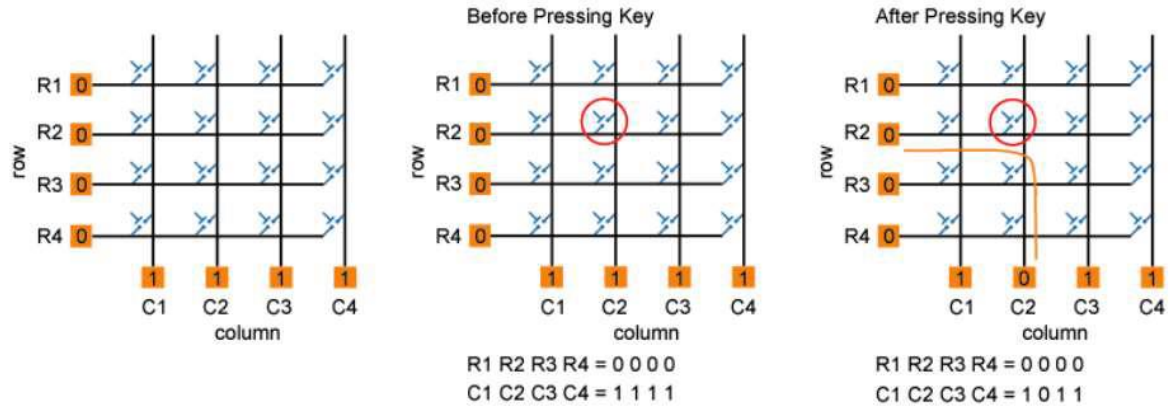
Keyboards are organized in a matrix of rows and columns. The 8051 can accesses both rows and columns through ports. Therefore, with two 8-bit ports, an 8 x 8 matrix of keys can be connected to a microcontroller. When a key is pressed, a row and a column make a contact. Otherwise, there is no connection between rows and columns. A 4x4 matrix connected to two ports. The rows are connected to an output port and the columns are connected to an input port.

Grounding Rows and reading Columns:-



Key Scanning:

It is the function of the microcontroller to scan the keyboard continuously to detect and identify the key pressed. To detect a pressed key, the microcontroller grounds all rows by providing 0 to the output latch, then it reads the columns. If the data read from columns is D3 – D0 =1111, no key has been pressed and the process continues till key press is detected. If one of the column bits has a zero, this means that a key press has occurred. For example, if D3 – D0 = 1101, this means that a key in the D1 column has been pressed. After detecting a key press, microcontroller will go through the process of identifying the key. Starting with the top row, the microcontroller grounds it by providing a low to row D0 only. It reads the columns, if the data read is all 1s, no key in that row is activated and the process is moved to the next row. It grounds the next row, reads the columns, and checks for any zero. This process continues until the row is identified. After identification of the row in which the key has been pressed. Find out which column the pressed key belongs to.



6. Write 8051 C program to transmit a message “VTU” serially at the baud rate of 9600. Use 8 bit data with one stop and one start.

```
#include <reg51.h>

void SerTx(unsigned char);

void main(void){

TMOD=0x20; //use Timer 1, mode 2

TH1=0xFD; //9600 baud rate

SCON=0x50;

TR1=1; //start timer

while (1) {

SerTx(“V”);

SerTx(“T”);

SerTx(“U”);

}

}

void SerTx(unsigned char x){

SBUF=x; //place value in buffer

while (TI==0); //wait until transmitted

TI=0;

}
```

7. Write and explain a C program and assembly to monitor the status of a switch SW connected to pin P2.7 and perform the following:

i) If SW=0, the stepper motor rotates clockwise.

ii) If SW=1, the stepper motor rotates in anticlockwise

Use wave drive 4-step sequence

```
#include<reg51.h>
sbit SW = P2^7;
sbit Enable = P1^0;
sbit M1 = P1^1;
sbit M2=P1^2;
void main()
{
SW =1;
Enable =0;
M1=0;
M2=0;
While(1)
{
        Enable = 1;
        If (SW ==1)
        {
                M1 = 1;
                M2 = 0;
        }
        else
        {   M1 = 0;
            M2 = 1;
        }
}
}
```

8a. Write a program that continuously gets 8-bit data from ‘P0’ and sends it to ‘P1’ where simultaneously creating a square wave of 200μs period on pin P2.0. Use timer-0 to create square wave. Assume XTAL=11.0592.

```
ORG 0000H
```

```
LJMP MAIN
ORG 000BH
CPL P2.1
RETI
ORG 0030H
MAIN:  MOV TMOD, #02H
MOV P0,#0FFH
MOV TH0, #-92
MOV IE,#82H
SETB TR0
BACK: MOV A, P0
MOV P1,A
SJMP BACK
END
```

8b. Explain the importance of TI and RI flags.

When 8051 is transmitting serial data then TI flag, the flag must be monitored to check whether data transfer operation is executed successfully or not. This bit is set by the hardware at the beginning of the stop bit in mode 1. This flag bit must be cleared by software. In embedded C the instructions that can be used to monitor the status of TI flag is-

```
SBUF = 'U';      // load SBUF with the data that is required to be transmitted
While (TI ==0);       //stay on this line till TI=0 (wait until transmitted)
TI=0;              // clear TI flag
```

Similarly, during reception RI flag, the flag must be monitored to check whether any data is received or not. This flag bit is set in half way through the stop bit in mode 1 and must be cleared by the software. The instructions that can be used to monitor the status of RI flag in embedded C are-

```
While (RI==0); // stay on this line till RI=0 (wait to receive)
temp = SBUF; //read SBUF and store it in some temp variable
P2 = temp;  // put the received data on port 2
```

```
RI=0;   // clear the RI flag
```

In simple terms TI = 1 indicates that data is transferred successfully and RI = 1 indicates that data byte is received successfully.