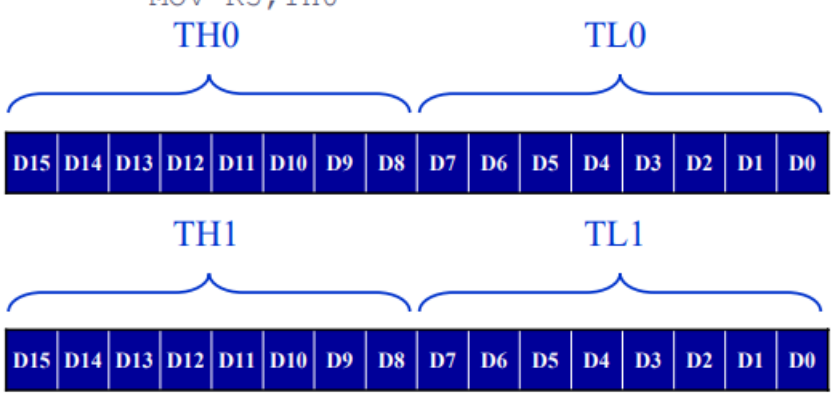
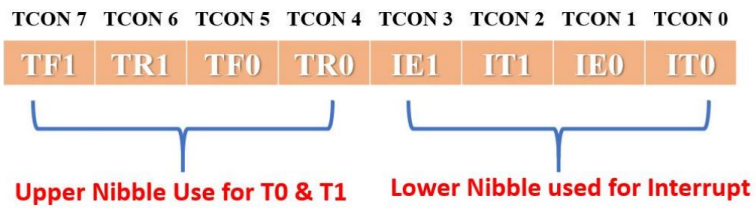
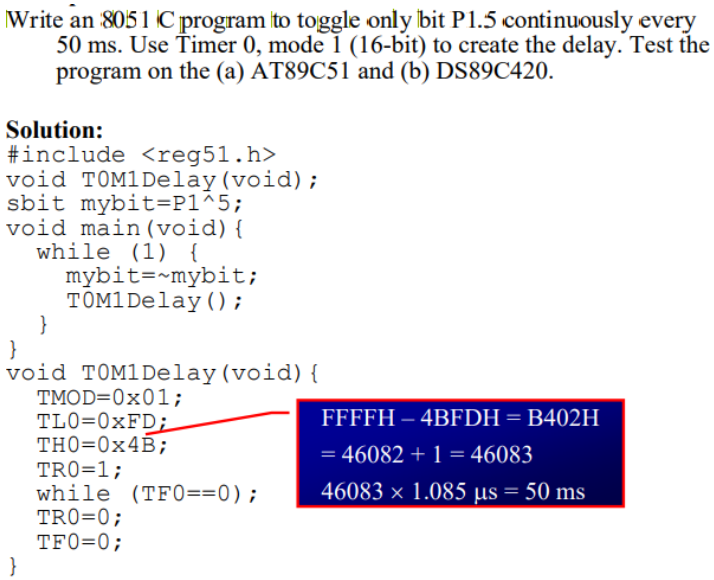
1	Name the timer/ counter available in 8051. Explain the working of the timer/ counter in brief. The 8051 has two timers/counters, they can be used either as Timers to generate a time delay or as Event counters to count events happening outside the microcontroller Both Timer 0 and Timer 1 are 16 bits wide Since 8051 has an 8-bit architecture, each 16-bits timer is accessed as two separate registers of low byte and high byte The low byte register is called TL0/TL1 The high byte register is called TH0/TH1  Both timers 0 and 1 use the same register, called TMOD (timer mode), to set the various timer operation mode.
2	Which are the different Special Function Registers associated with Timer/ Counter operation. TMOD(Timer mode register) and TCON(Timer control register) TMOD is a 8-bit register  Gate : if 0,the start and stop operations can be controlled by software. If 1, the start and stop operations can be controlled by hardware. C/T:Timer or counter selected Cleared for timer operation (input from internal system clock) If C/T = 0, it is used as a timer for time delay generation. Set for counter operation (input from Tx input pin). M1 | M0 | Mode of operation 0 | 0 | Mode 0 (13-bit timer mode) 0 | 1 | Mode 1 (16-bit timer mode) 1 | 0 | Mode 2 (8-bit Auto Reload mode) 1 | 1 | Mode 3 (Split timer mode) TCON(Timer control register) • It is an 8-bit register.

	
3	With a neat diagram explain each bit of TMOD register. TMOD is a 8-bit register ->The lower 4 bits are for Timer 0 ->The upper 4 bits are for Timer 1 ->In each case, ƒ The lower 2 bits are used to set the timer mode ƒ The upper 2 bits to specify the operation  Gate : if 0,the start and stop operations can be controlled by software. If 1, the start and stop operations can be controlled by hardware. C/T:Timer or counter selected Cleared for timer operation (input from internal system clock) If C/T = 0, it is used as a timer for time delay generation. Set for counter operation (input from Tx input pin).C/T=1 M1,M2: **M1 / M0 — Mode — Operating Mode** **0 0 — 0 — 13-bit timer mode** 8-bit timer/counter THx with TLx as 5-bit prescaler **0 1 — 1 — 16-bit timer mode** 16-bit timer/counter THx and TLx are cascaded; there is no prescaler **1 0 — 2 — 8-bit auto reload** 8-bit auto reload timer/counter; THx holds a value which is to be reloaded TLx each time it overfolws **1 1 — 3 — Split timer mode**
4	Indicate which mode and which timer are selected for each of the following. a) TMOD= 0x10 timer 1 in mode 1 b) TMOD= 0x01 timer 0 in mode 1 c) TMOD= 0x11 timer 0 in mode1 ,timer 1 in mode 1 d) TMOD= 0x20 timer 1 in mode 2 e) TMOD= 0x02 timer 0 in mode 2
5	Find the timer's clock frequency and time period for the following XTAL frequencies: a) 11.0592 MHz Timer,s clock frequency : 11.0592/12=0.9216 mhz Time period = 1/0.9216=1.085 us b) 16 MHz Timer,s clock frequency : 16/12=1.33 mhz Time period = 1/1.33=0.75 us

6	Load TMOD with appropriate value to configure: a) Timer 0 in Mode 0 TMOD=0X00 b) Timer 1 in Mode 0 TMOD=0X00 c) Timer 0 in Mode 0, Timer 1 in Mode 0 TMOD=0X00 d) Timer 0 in Mode 0, Timer 1 in Mode 1 TMOD=0X10 e) Timer 0 in Mode 0, Timer 1 in Mode 2 TMOD=0X20 f) Timer 0 in Mode 1, Timer 1 in Mode 2 TMOD=0X21 g) Timer 0 in Mode 1, Timer 1 in Mode 2 TMOD=0X21 h) Timer 0 in Mode 1, Timer 1 in Mode 2 TMOD=0X21 i) Timer 0 in Mode 2, Timer 1 in Mode 2 TMOD=0X22 j) Timer 0 in Mode 2, Timer 1 in Mode 2 TMOD=0X22 k) Timer 0 in Mode 2, Timer 1 in Mode 2 TMOD=0X22

7	Write an 8051 C program to toggle only bit P1.5 continuously every 50 ms. Use Timer 0, mode 1 (16-bit) to create the delay.
	 
	Write an 8051 C program to toggle only bit P1.5 continuously every 50 ms. Use Timer 0, mode 1 (16-bit) to create the delay. Test the program on the (a) AT89C51 and (b) DS89C420.
	
	Solution:
	`#include <reg51.h>`
	`void T0M1Delay(void);`
	`sbit mybit=P1^5;`
	`void main(void){`
	`  while (1) {`
	`  mybit=~mybit;`
	`  T0M1Delay();`
	`  }`
	`}`
	`void T0M1Delay(void){`
	`  TMOD=0x01;`
	`  TL0=0xFD;`
	`  TH0=0x4B;`
	`  TR0=1;`
	`  while (TF0==0);`
	`  TR0=0;`
	`  TF0=0;`
	`}`
	FFFFH – 4BFDH = B402H
	= 46082 + 1 = 46083
	46083 × 1.085 µs = 50 ms

8	Write an 8051 C program to toggle only pin P1.5 continuously every 250 ms. Use Timer 0, mode 2 (8-bit auto-reload) to create the delay.
	
	`#include<reg51.h>`
	`void delay(void);`
	`sbit mybit=P1^5;`
	`void main(void){`
	` unsigned int x;`
	` while(1){`
	`  mybit=~mybit;`
	`for(x=0;x<1000;x++)`
	`  delay();`
	`}`
	`}`
	`void delay(void){`
	`TMOD=0X02;`
	`TH0=1A;`
	`TR0=1;`
	`While(TF0=0);`
	`TR0=0;`
	`TF0=0;`
	`}`

9	Assume that a 1-Hz external clock is being fed into pin T1 (P3.5).Write a C program for counter 1 in mode 2 (8-bit auto reload) to count up and display the state of the TL1 count on P1. Start the count at 0H.
	#include<reg51.h>
	sbit T1=P1^5;
	void main(void) {
	T1=1;
	TMOD=0X60;
	TH1=0;
	While(1) {
	Do {
	TR1=1;
	P1=TL1;
	} while(TF1==0);
	}
	TR1=0;
	TF1=0;
	}

10	Assume that a 1-Hz external clock is being fed into pin T0 (P3.4). Write a C program for counter 1 in mode 1 (16-bit) to count the pulses and display the state of the TH0 and TL0 registers on P2 and P1,respectively.
	
	#include<reg51.h>
	sbit T1=P3^4;
	void main(void) {
	T0=1;
	TMOD=0X05;
	TH0=0;
	TL0=0;
	While(1) {
	Do {
	TR0=1;
	P2=TH0;
	P1=TL0;
	}while(TF0==0);
	}
	TR0=0;
	TF0=0;
	}

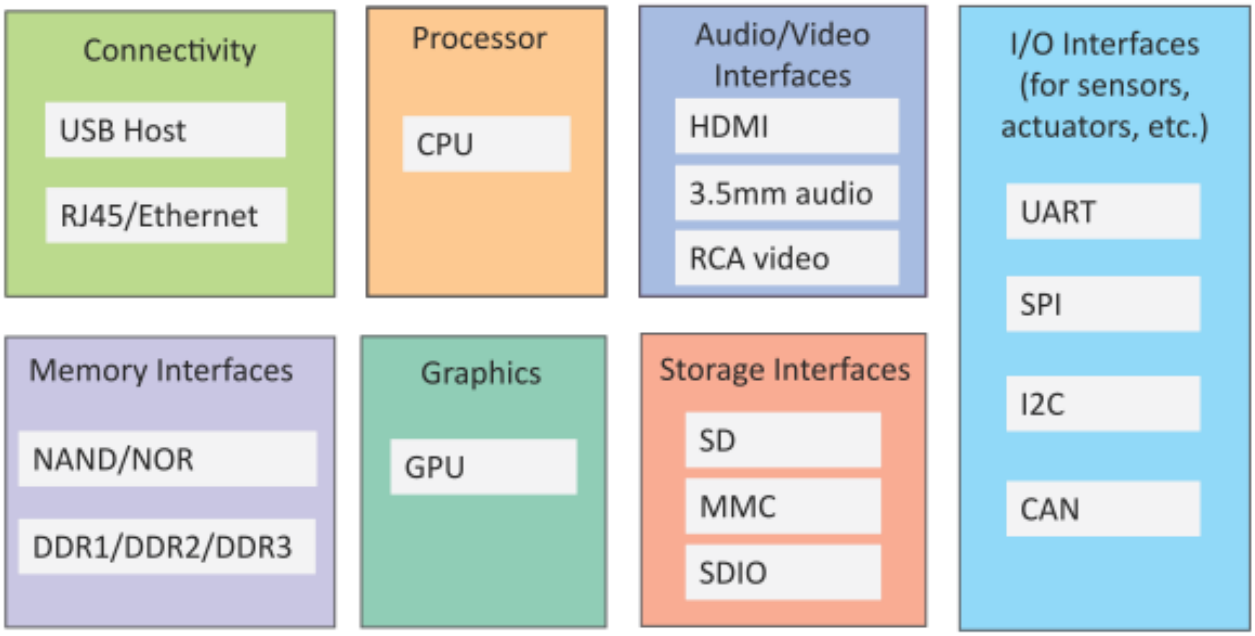
11	Explain each bit of SCON register.
12	List out the steps in programming the 8051 to transfer character bytes serially. 1. TMOD register is loaded with the value 20H, indicating the use of timer 1 in mode 2 (8-bit auto-reload) to set baud rate. 2. The TH1 is loaded with one of the values to set baud rate for serial data transfer. 3. The SCON register is loaded with the value 50H, indicating serial mode 1, where an 8- bit data is framed with start and stop bits. 4. TR1 is set to 1 to start timer 1. 5. TI is cleared by CLR TI instruction. 6. The character byte to be transferred serially is written into SBUF register. 7. The TI flag bit is monitored with the use of instruction JNB TI,xx to see if the character has been transferred completely. 8. To transfer the next byte, go to step 5

13	List out the steps in programming the 8051 to receive character bytes serially. **In programming the 8051 to receive character bytes serially** 1. TMOD register is loaded with the value 20H, indicating the use of timer 1 in mode 2 (8-bit auto-reload) to set baud rate 2. TH1 is loaded to set baud rate 3. The SCON register is loaded with the value 50H, indicating serial mode 1, where an 8-bit data is framed with start and stop bits 4. TR1 is set to 1 to start timer 1 5. RI is cleared by `CLR RI` instruction 6. The RI flag bit is monitored with the use of instruction `JNB RI,xx` to see if an entire character has been received yet 7. When RI is raised, SBUF has the byte, its contents are moved into a safe place 8. To receive the next character, go to step 5

14	Write an 8051 C program to transfer the message “YES” serially at 9600 baud, 8-bit data, 1 stop bit. Do this continuously.
	
	`#include <reg51.h>`
	`void SerTx (unsigned char);`
	`void main(void) {`
	`THOD=0x20;  //use Timer 1, mode 2`
	`TH1=0xFD;  //9600 baud rate`
	`SCON=0x50;`
	`TR1=1;  //start timer`
	`while (1){`
	`SerTx( 'Y');`
	`SerTx('E');`
	`SerTx('S');`
	`} }`
	`void SerTx(unsigned char x) {`
	`SBUF=x;  //place value in buffer`
	`while (TI==0);  //wait until transmitted`
	`TI=0;`
	`}`

15	Program the 8051 in C to receive bytes of data serially and put them in P1. Set the baud rate at 4800, 8-bit data, and 1 stop bit.
	
	#include <reg51.h>
	void main(void) {
	unsigned char mybyte;
	TMOD=0x20; //use Timer 1, mode 2
	THI=0xFA; //4800 baud rate
	SCON=0x50;
	TR1=1; //start timer
	while (1) { //repeat forever
	while (RI==0); //wait to receive
	mybyte=SBUF; //save value
	Pl=mybyte; //write value to port
	RI=0;
	}
	}

16	Interface DAC 0800 with 8051 Microcontroller and develop an Embedded 'C' program to generate the rectangular waveform with 65% duty cycle on P0. Assume XTAL= 11.0592 MHz and T=100ms.
17	Define IoT & explain its characteristics. • Internet Of Things is Fully Networked and Connected Devices sending analytics data back to cloud or data center. • The definition of Internet of things is that it is the network in which every object or thing is provided unique identifier and data is transferred through a network without any verbal communication. ***Characteristics*** • **Dynamic Global network & Self-Adapting:** Adapt the changes w.r.t changing contexts • **Self-Configuring:** E.g., Fetching latest software updates without manual intervention. • **Interoperable Communication Protocols:** Communicate through various protocols • **Unique Identity:** Such as Unique IP Address or a URI • **Integrated into Information Network:** This allows to communicate and exchange data with other devices to perform certain analysis.

18	Explain in detail a generic block diagram of an IoT Device.  1. **Connectivity:** Devices like USB hosts and Ethernet are used for connectivity between the devices and the server. 2. **Processor** A processor like a CPU and other units are used to process the data. These data are further used to improve the decision quality of an IoT system. 3. **Audio/Video Interfaces** AN interface like HDMI and RCA devices is used to record audio and videos in a system. 4. **Input/Output Interface** To give input and output signals to sensors, and actuators we use things like UART, SPI, CAN, etc. 5. **Storage Interfaces** Things like SD, MMC and SDIO are used to store the data generated from an IoT device. 6. Other things like DDR and GPU are used to control the activity of an IoT system.
19	Describe an example of IoT Service that uses Publish-Subscribe communication model.

20	Describe an example of IoT Service that uses Request Response communication model ➢ The client, when required, requests the information from the server. This request is usually in the encoded format. ➢ This model is stateless since the data between the requests is not retained and each request is independently handled. ➢ The server categories the request and fetches the data from the database and its resource representation. This data is converted to response and is transferred in an encoded format to the client. The client, in turn, receives the response. ➢ On the other hand - In Request-Response communication model client sends a request to the server and the server responds to the request. When the server receives the request, it decides how to respond, fetches the data retrieves resources, and prepares the response and sends it to the client.
21	Briefly explain all the six IoT levels. A level-1 IoT system has a single node/device that performs sensing and/or actuation, stores data, performs analysis and hosts the application. Level-1 IoT systems are suitable for modelling low-cost and low-complexity solutions where the data involved is not big and the analysis requirements are not computationally intensive. **Example:** Home Automation System A level-2 IoT system has a single node that performs sensing and/or actuation and local analysis. Data is stored in the cloud and the application is usually cloud-based. Level-2 IoT systems are suitable for solutions where the data involved is big; however, the primary analysis requirement is not computationally intensive and can be done locally. **Example:** Smart Irrigation A level-3 IoT system has a single node. Data is stored and analysed in the cloud and the application is cloud-based. Level-3 IoT systems are suitable for solutions where the data involved is big and the analysis requirements are computationally intensive. **Example:** Tracking Package Handling A level-4 IoT system has multiple nodes that perform local analysis. Data is stored in the cloud and the application is cloud-based. Level-4 contains local and cloud-based observer nodes which can subscribe to and receive information collected in the cloud from IoT devices. Level-4 IoT systems are suitable for solutions where multiple nodes are required, the data involved is big and the analysis requirements are computationally intensive. **Example:** Noise Monitoring A level-5 IoT system has multiple end nodes and one coordinator node. The end nodes perform sensing and/or actuation. The coordinator node collects data from the end nodes and sends it to the cloud. Data is stored and analysed in the cloud and the application is cloud-based.

	Level-5 IoT systems are suitable for solutions based on wireless sensor networks, in which the data involved is big and the analysis requirements are computationally intensive. **Example:** Forest Fire Detection A level-6 IoT system has multiple independent end nodes that perform sensing and/or actuation and send data to the cloud. Data is stored in the cloud and the application is cloud-based. The analytics component analyzes the data and stores the results in the cloud database. The results are visualized with the cloud-based application. The centralized controller is aware of the status of all the end nodes and sends control commands to the nodes. **Example:** Weather Monitoring System
22	Illustrate the Home Automation IoT application w.r.t. Level-1 Deployment model
23	Illustrate the weather monitoring IoT application w.r.t. suitable Deployment level