USN | | | | | | | | | | |

Internal Assessment Test III – SEP 2023

Sub:	MICROCONLLER AND EMBEDDED SYSTEM					Sub Code:	21CS43	Branch:	CSE
Date:	09/09/23	Duration:	90 mins	Max Marks:	50	Sem / Sec:	IV Sem A/B/C		OBE

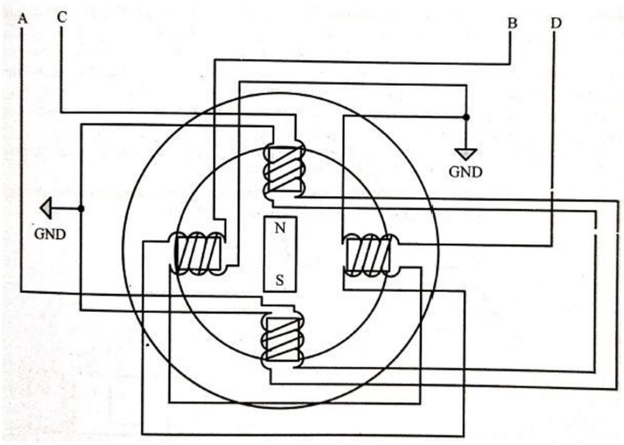
Answer any FIVE FULL Questions

		MARKS	CO	RBT
1	**i) Difference between Sensor and Actuators** **Ans:**	[4+3+3]	CO2 CO3	L2

Sensors	Actuators
1. Sensor is an input device	1. Actuator is an output device
2. Convert a physical parameter to an electrical output	2. Convert an electrical signal to a physical Output
3. A device that detects events or changes in the environment and send the information to another electronic device	3. A component of a machine that is responsible for moving and controlling mechanisms
4. Sensor help to monitor the changes in the Environment	4. Actuator helps to control the environment or physical changes

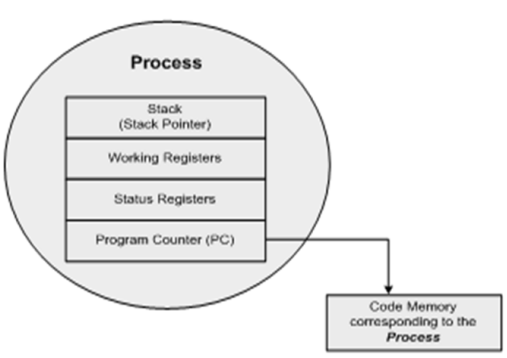
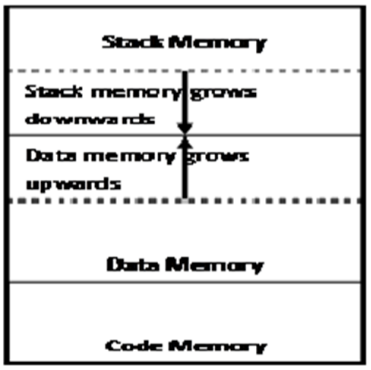
ii) Explain the working principle of Stepper Motor

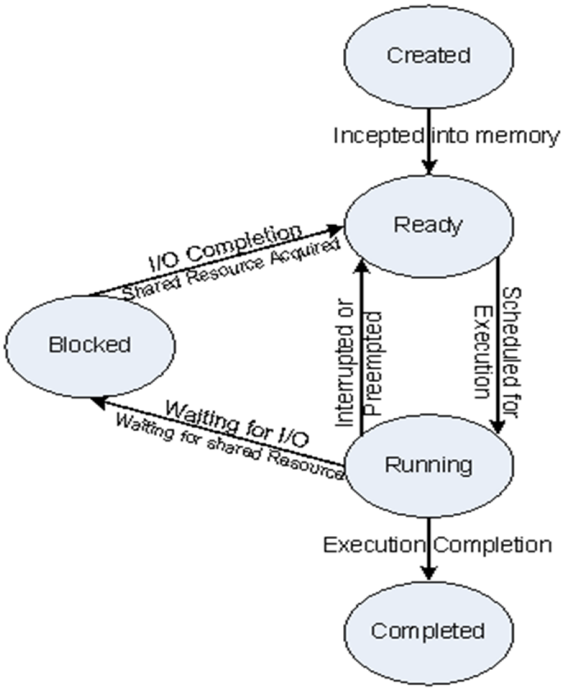
Ans: A *stepper motor* is an electro-mechanical device which generates discrete displacement (motion) in response, to de-electrical signals.



Based on the coil winding arrangements, a two-phase stepper motor is classified into two. They are:

» **Unipolar:** A unipolar stepper motor contains two windings per phase.

	» The direction of rotation (clockwise or anticlockwise) of a stepper motor is controlled by changing the direction of current flow. » The coils are represented as A, B, C and D. » Coils A and C carry current in opposite directions for phase 1 » Similarly, B and D carry current in opposite directions for phase 2. **Bipolar:** A bipolar stepper motor contains single winding per phase. For reversing the motor rotation the current flow through the windings reversed dynamically.			
2	**Define Process. Explain in detail the structure, memory organization and state transmission of the process.** **Ans:** A "**Process**" is a program, or part of it, in execution. Process is also known as an instance of a program in execution. A process requires various system resources like CPU for executing the process, memory for storing the code corresponding to the process and associated variables, I/O devices for information exchange etc. **Structure of a Processes:** The concept of "Process" leads to concurrent execution of tasks and thereby, efficient utilization of the CPU and other system resources. Concurrent execution is achieved through the sharing of CPU among the processes.  Structure of a Process  Memory Oraganization of a Process **Memory Organization**: • A process which inherits all the properties of the CPU. When the process gets its turn, its registers and Program Counter register becomes mapped to the physical registers of the CPU. • The memory occupied by the process is segregated into three regions namely; Stack memory, **Data memory and Code memory** The "***Stack***" memory holds all temporary data such as variables local to the process. The "***Data***" **memory** holds all global data for the process. The "***Code***" **memory** contains the program code (instructions) corresponding to the process. **Process state transmission:**	[2+8]	CO2 CO3	L2



Created State: The state at which a process is being created is referred as "Created State". The Operating System recognizes a process in the "Created "State" but no resources are allocated to the process.

Ready State: The state, where a process is incepted into the memory and awaiting the processor time for execution, is known as "Ready State". At this stage, the process is placed in the "Ready list" queue maintained by the OS.

Running State: The state where in the source code instructions corresponding to the process is being executed is called "Running State". Running state is the state at which the process execution happens.

Blocked State/ Wait State: Refers to a state where a running process is temporarily suspended from execution and does not have immediate access to resources. The blocked state might have invoked by various conditions like- the process enters a wait state for an event to occur (E.g. Waiting for user inputs such as keyboard input) or waiting for getting access to a shared resource like semaphore, mutex etc.

Completed State: A state where the process completes its execution.

3	**i) Difference between CPLD and FPGA** **Ans:**	[2+2+6]	CO2 CO3	L2

CPLDs	FPGAs
1. PLD is used for construction of CPLD	1. Logic blocks are used for construction of FPGA
2. CPLD is non-volatile & less costly	2. FPGA is volatile & costly
3. Delays are much more predictable in CPLDs	3. Prediction of delay is difficult in FPGA
4. Operating speed is low & is suitable for control circuit	4. Operating speed is high & is suitable for timing circuit
5. CPLD has less flexibility and design Capacity	5. FPGA has more flexibility as well as design capacity
6. CPLD could work immediately after power up	6. FPGA could not work until the configuration is done
7. CPLDs are considered as 'coarse-grain' devices	7. FPGAs are considered as 'fine-grain' devices

ii) Difference between ASIC and FPGA

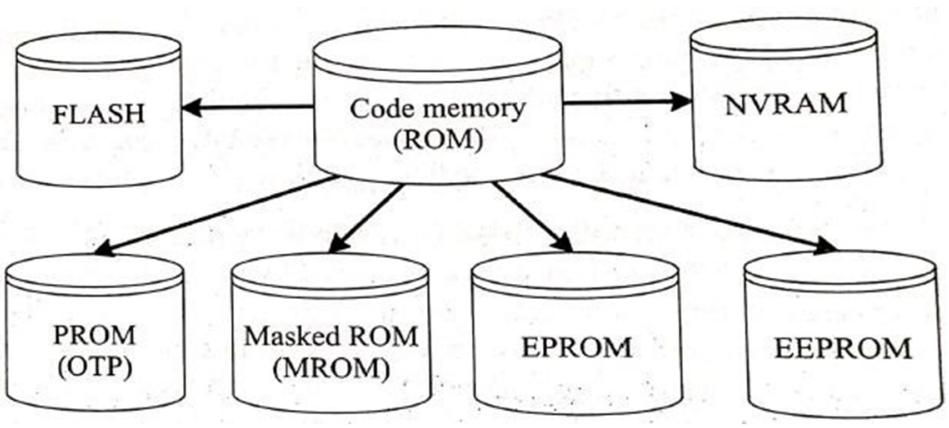
Ans:

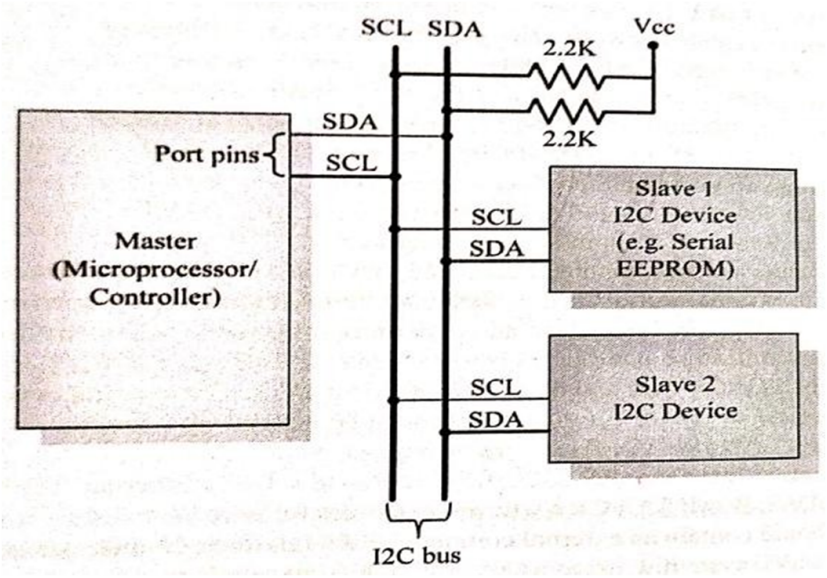
FPGAs	ASICs
1. FPGA is a reprogrammable integrated circuit	1. ASIC is a unique type of integrated circuit meant for a specific application
2. FPGA is not efficient in terms of use of Materials	2. ASIC wastes very little material, recurring cost is low
3. FPGA is better than ASIC when building low volume production circuits	3. Cost of ASIC is low only when it is produced in large qualtity
4. FPGA is alterable	4. Once created, ASIC can no longer be Altered
5. FPGAs are useful for research and de3velopment activities. Prototype fabrication using FPGA is affordable and fast	5. ASICs are not suitable for research and development purpose, as they are reconfigurable

iii) Briefly explain about Embedded Memory

Ans: » Memory is an important part of a processor/ controller based embedded systems.

» Some of the processors/ controllers contain built in memory and this memory is referred as *on-chip memory*.

	» Others do not contain any memory inside the chip and requires external memory to be connected with the controller/processor to store the control algorithm. It is called *off-chip memory*. » Also some working memory is required for holding data temporarily during certain operations.  **Masked Memory (MROM):** A one-time programmable device, Uses hardwired technology for storing data, Factory programmed by masking and metallization process at the time of production. **Programmable Read Only Memory (PROM)/ One Time Programmable Memory (OTP):** Not pre-programmed by the manufacturer, End user is responsible for programming . **Erasable Programmable Read Only Memory (EPROM):** EPROM gives the flexibility to reprogram the same chip. EPROM stores the bit information by charging the floating gate of an FET. Bit information is stored by using an EPROM programmer, which applies high voltage to charge the floating gate. **Electrically Erasable Programmable Read Only Memory (EEPROM):** The information contained in the EEPROM memory can be altered by using electrical signals at the register/Byte level. **FLASH:** FLASH is the latest ROM technology and is the most popular ROM technology used in today's embedded designs. FLASH memory is a variation of EEPROM technology. It combines the reprogrammability of EEPROM and the high capability of standard ROMs. **NVRAM: Non-volatile RAM is a random access memory with battery** backup.			
4	**i) Explain the working principle of I2C Bus** **Ans:**	[4+3+3]	CO3 CO2	L3



The sequence of operations for communicating with an I2C slave device is listed below:

1.The master device pulls the clock line (SCL) of the bus to 'HIGH '

2.The master device pulls the data line (SDA) 'LOW', when the SCL line is at logic 'HIGH' (This is the 'Start' condition for data transfer)

3.The master device sends the address (7-bit or 10-bit wide) of the 'slave' device to which it wants to communicate, over the SDA line. Clock pulses are generated at the SCL line for synchronizing the bit reception by the slave device. The MSB of the data is always transmitted first. The data in the bus is valid during the 'HIGH' period of the clock signal

4.The master device sends the Read or Write bit (Bit value = 1 Read operation; Bit value = 0 Write operation) according to the requirement

5.The master device waits for the acknowledgement bit from the slave device

6.The slave device with the address requested by the master device responds by sending an acknowledge bit (Bit value = 1) over the SDA line

7.Upon receiving the acknowledge bit, the master device sends the 8-bit data to the slave device over SDA line, if the requested operation is 'Write to device'. If the requested operation is 'Read from device', the slave device sends data to the master over the SDA line

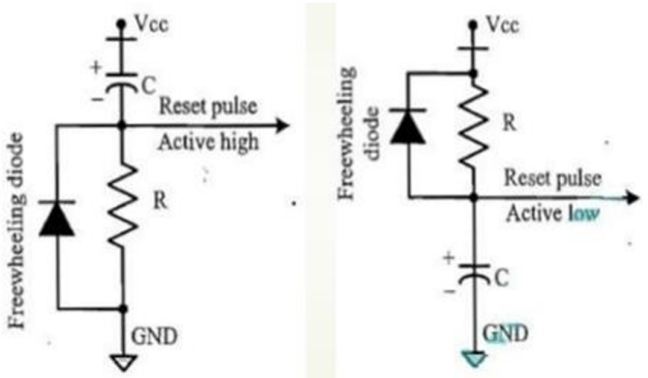
8.The master device waits for the acknowledgement bit from the device upon byte transfer complete for a write operation and sends an acknowledge bit to the Slave device for a read operation

9.The master device terminates the transfer by pulling the SDA line 'HIGH' when the clock line SCL is at logic 'HIGH' (Indicating the 'STOP' condition).

ii) Write a short note on a) Reset Circuit b) Watch Dog timer

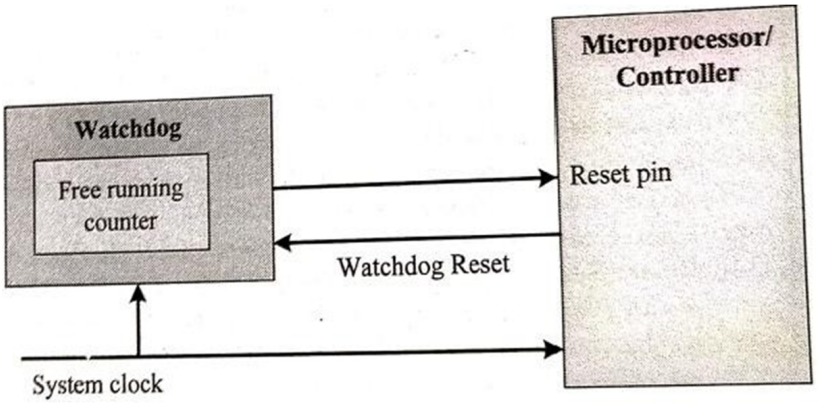
Ans: Reset Circuit: The *reset circuit* is essential to ensure that the device is not operating at a voltage level where the device is not guaranteed to operate, during system power ON.

- The reset signal brings the internal registers and the different hardware systems of the processor/ controller to a known state and starts the firmware execution from the reset vector (Normally from vector address 0x0000 for conventional processors/ controllers.
- The reset signal can be either active high or active low. Since the processor operation is synchronized to a clock signal, the reset pulse should be wide enough to give time for the clock oscillator to stabilize before the internal reset state starts.
- The reset signal to the processor can be applied at power ON through an external passive reset circuit comprising a Capacitor and Resistor or through a standard Reset IC like MAX810 from Maxim Dallas. Select the reset IC based on the type of reset signal and logic level (CMOS/ TTL) supported by the processor/ controller in use.
- Some microprocessors /controllers contain built-in internal reset circuitry and they don't require external reset circuitry.



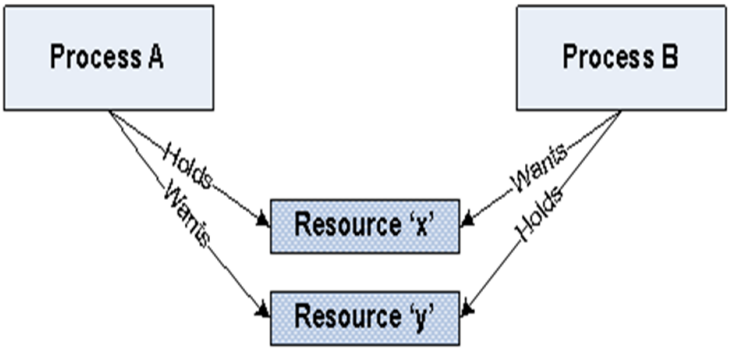
Watch Dog Timer:

- A watchdog to monitor the firmware execution and reset the system processor/ microcontroller when the program execution hangs up. A *watchdog timer*, or simply a *watchdog*, is a hardware timer for monitoring the firmware execution.
- Depending on the internal implementation, the watchdog timer increments or decrements a free running counter with each clock pulse and generates a reset signal to reset the processor if the count reaches zero for a down counting watchdog, or the highest count value for an up counting watchdog.
- If the watchdog counter is in the enabled state, the firmware can write a zero (for up counting watchdog implementation) to it before starting the execution of a piece of code and the watchdog will start counting.
- If the firmware execution doesn't complete due to malfunctioning, within the time required by the watchdog to reach the maximum count, the counter will generate a reset pulse and this will reset the processor.

	● If the firmware execution completes before the expiration of the watchdog, you can reset the count by writing a 0 (for an up counting watchdog timer) to the watchdog timer register.  			
5	**i) Explain Multi-processing, Multi-tasking and Multi-programming** **Ans:** ● **Multiprogramming** – Multiprogramming is known as keeping multiple programs in the main memory at the same time ready for execution. ● **Multiprocessing** – A computer using more than one CPU at a time. ● **Multitasking** – Multitasking is nothing but multiprogramming with a Round-robin scheduling algorithm. **ii) Write a note on Message Passing** **Ans: Message passing** is a/ an synchronous/ asynchronous information exchange mechanism for Inter Process/ Thread Communication. ● The major difference between shared memory and message passing technique is ❖ Through shared memory lots of data can be shared whereas only limited amount of info/ data is passed through message passing. ❖ Message passing is relatively fast and free from the synchronization overheads compared to shared memory. ● Based on the message passing operation between the processes, message passing is classified into – 1. Message Queue 2. Mailbox 3. Signalling **1. Message Queue:** Process which wants to talk to another process posts the message to a First-In-First-Out (FIFO) queue called "*Message queue*", which stores the messages temporarily in a system defined memory object, to pass it to the desired process. **2. Mailbox:** *Mailbox* is a special implementation of message queue. Usually used for one way communication, only a single message is exchanged through mailbox whereas „message queue" can be used for exchanging multiple messages. **3. Signalling:** Signals are used for an asynchronous notification mechanism. The signal mainly used for the execution synchronization of tasks process/ tasks. Signals do not carry any data and are not queued. The implementation of signals is OS kernel dependent.	[2+2+2] [4]	CO4	L3
6	**Illustrate the concept of "deadlock" with a neat diagram. Mention the different conditions favour a deadlock situation.**	[5+5]	CO5	L3

Ans: Deadlock is the condition in which a process is waiting for a resource held by another process which is waiting for a resource held by the first process; hence, none of the processes are able to make any progress in their execution.

- Process A holds a resource "*x*" and it wants a resource "*y*" held by Process B.
- Process B is currently holding resource "*y*" and it wants the resource "*x*" which is currently held by Process A.
- Both hold the respective resources and they compete each other to get the resource held by the respective processes.



Different conditions favour a deadlock situation.

Mutual Exclusion: The criteria that only one process can hold a resource at a time. Meaning processes should access shared resources with mutual exclusion.

Hold & Wait: The condition in which a process holds a shared resource by acquiring the lock controlling the shared access and waiting for additional resources held by other processes.

No Resource Preemption: The criteria that Operating System cannot take back a resource from a process which is currently holding it and the resource can only be released voluntarily by the process holding it.

Circular Wait: A process is waiting for a resource which is currently held by another process which in turn is waiting for a resource held by the first process. In general there exists a set of waiting process P0, P1 …. Pn with P0 is waiting for a resource held by P1 and P1 is waiting for a resource held by P0, ……,Pn is waiting for a resource held by P0 and P0 is waiting for a resource held by Pn and so on. This forms a circular wait queue.