1. **Interrupts**

1.What is Timer ?

2. What is Counter ?

3. What happens when an interrupt occurs?

4. What are the various types of interrupts?

5. What is stored in interrupt vector table?

6. What is interrupt mask bit?

7. What are timers and counters? Are they different on hardware or are they same?

8. What is interrupt? Explain. If interrupt occurs, how system will perform?

9. What happens when interrupt flag is raised in microcontroller?

10. How to generate an interrupt for every 5000 ticks ?

11. How do you write the ISR when interrupts from multiple sources occur?

1. **What is Timer?**
   * A timer is a hardware or software component that keeps track of time intervals. In embedded systems, timers are used for tasks such as generating periodic interrupts, measuring time intervals, or generating delays.
2. **What is Counter?**
   * A counter is a device or software function that counts events, often triggered by external or internal signals. Counters can be used for counting pulses, events, or for generating timed intervals similar to timers.
3. **What happens when an interrupt occurs?**
   * When an interrupt occurs, the normal execution flow of the program is temporarily halted, and control is transferred to an interrupt service routine (ISR). The ISR handles the event that caused the interrupt, and once completed, control is returned to the main program where it left off.
4. **What are the various types of interrupts?**
   * There are several types of interrupts:
     + **Hardware Interrupts:** Triggered by hardware events such as timers, I/O ports, etc.
     + **Software Interrupts:** Triggered by software instructions.
     + **Maskable Interrupts:** Can be enabled or disabled by setting or clearing a bit.
     + **Non-maskable Interrupts (NMI):** Cannot be disabled and are typically used for critical events.
     + **Vectored Interrupts:** The address of the ISR is predefined.
     + **Non-vectored Interrupts:** The address of the ISR is not predefined and must be provided by the interrupting device.
5. **What is stored in interrupt vector table?**
   * The interrupt vector table stores the addresses of the interrupt service routines (ISRs) for various interrupt sources. When an interrupt occurs, the system uses this table to find and execute the corresponding ISR.
6. **What is interrupt mask bit?**
   * An interrupt mask bit is a bit in a control register that can be set or cleared to enable or disable specific interrupts. Masking an interrupt means temporarily ignoring it, while unmasking allows it to be recognized and handled by the system.
7. **What are timers and counters? Are they different on hardware or are they same?**
   * Timers and counters are similar in functionality but serve different purposes:
     + **Timers:** Measure time intervals and can generate periodic events.
     + **Counters:** Count external events or pulses.
   * In hardware, timers and counters may use similar circuitry but are configured differently based on their intended use.
8. **What is interrupt? Explain. If interrupt occurs, how system will perform?**
   * An interrupt is a signal that prompts the system to stop its current operations and execute a special piece of code called an interrupt service routine (ISR). When an interrupt occurs, the system saves its current state, executes the ISR to handle the interrupt, and then restores its state to continue executing the original program.
9. **What happens when interrupt flag is raised in microcontroller?**
   * When an interrupt flag is raised, it indicates that an interrupt event has occurred. The microcontroller checks if the interrupt is enabled and if so, it halts the current execution, saves the context, and jumps to the ISR corresponding to the raised interrupt flag.
10. **How to generate an interrupt for every 5000 ticks?**
    * To generate an interrupt for every 5000 ticks, configure a timer in the microcontroller to generate an interrupt after counting 5000 clock cycles. This typically involves setting the timer's count value and enabling the timer interrupt.
11. **How do you write the ISR when interrupts from multiple sources occur?**
    * When writing an ISR for multiple interrupt sources, the ISR should identify the source of the interrupt by checking the interrupt flags and then handle each source accordingly. Here's an example in pseudocode:

c

Copy code

void ISR() {

if (interrupt\_flag1) {

// Handle interrupt from source 1

clear\_interrupt\_flag1();

}

if (interrupt\_flag2) {

// Handle interrupt from source 2

clear\_interrupt\_flag2();

}

// Add checks and handlers for other sources as needed

}

This approach ensures that all active interrupts are handled properly, and the system can respond to multiple interrupt sources efficiently.

1. **Digital Circuits**
2. **MC v/s MP**

1. Differences b/w Microprocessor vs microcontroller

2. How do you say whether a microcontroller is 8bit or 16bit?

3. . Explain 8086 microprocessors.

● How many bit processor it is?

● Why is it 16 bit?

● Can we do 32- bit operations?

1. What controller have you worked on? Explain.

 **Differences between Microprocessor and Microcontroller:**

* **Microprocessor:**
  + General-purpose processing unit.
  + Requires external components like memory and peripherals to function.
  + Used in computers and high-performance systems.
* **Microcontroller:**
  + Integrated system with a CPU, memory, and peripherals on a single chip.
  + Designed for specific control applications.
  + Used in embedded systems like home appliances, automotive controls, etc.

 **How to say whether a microcontroller is 8-bit or 16-bit?**

* The bit size of a microcontroller is determined by the width of its data bus, registers, and ALU (Arithmetic Logic Unit). If these components can handle 8 bits at a time, it's an 8-bit microcontroller. If they handle 16 bits, it's a 16-bit microcontroller.

 **Explain 8086 Microprocessors:**

* **Bit Processor:** 16-bit processor.
* **Why 16-bit:** It has a 16-bit data bus and 16-bit registers, meaning it can process 16 bits of data at a time.
* **32-bit Operations:** Yes, but indirectly. The 8086 can perform 32-bit operations by combining two 16-bit operations, but it's less efficient than a true 32-bit processor.

### PIC16F4580 Microcontroller

**Description:** The PIC16F4580 is an 8-bit microcontroller from Microchip Technology. It is part of the PIC16F family and is commonly used in embedded systems for various applications due to its versatility and ease of use.

**Key Features:**

* **CPU:** 8-bit
* **Memory:**
  + **Program Memory:** 16 KB Flash
  + **Data Memory:** 368 Bytes RAM
  + **EEPROM:** 256 Bytes
* **Clock Speed:** Up to 20 MHz
* **I/O Pins:** 36
* **Peripherals:**
  + **Timers:** Three timers (Timer0, Timer1, Timer2)
  + **ADC:** 10-bit Analog-to-Digital Converter with 13 channels
  + **Comparators:** Two analog comparators
  + **PWM:** Pulse Width Modulation capabilities
  + **USART:** Universal Synchronous Asynchronous Receiver Transmitter for serial communication
  + **MSSP:** Master Synchronous Serial Port supporting both SPI and I2C modes
  + **CCP:** Capture/Compare/PWM module

**Usage:** I used the PIC16F4580 in a project to develop a simple home automation system. Here’s a brief overview of how it was implemented:

1. **Sensors and Inputs:**
   * Connected various sensors like temperature, humidity, and light sensors to the ADC channels to monitor environmental conditions.
2. **Control Outputs:**
   * Used digital I/O pins to control relays for turning on and off home appliances like lights and fans.
3. **Communication:**
   * Employed the USART for serial communication with a PC for monitoring and controlling the system remotely.
4. **Timers and PWM:**
   * Utilized timers to create time-based events, such as scheduling when appliances should turn on or off.
   * Used PWM to control the speed of a fan based on the temperature sensor readings.
5. **Program Development:**
   * Developed the firmware using MPLAB X IDE and programmed the PIC16F4580 using a PICkit programmer.
   * Wrote the code in C using the MPLAB XC8 compiler, implementing functions for sensor data reading, control logic, and communication.
6. **Projects**

1.Car black box project [June 2023]

1. **ADC**
2. **PWM**

1.Explain PWM. Draw waveforms for 100% and 25% duty cycle

1. **Communication protocols**

1. Explain about CAN protocol in details [Oct 2023]

1. **Embedded Systems**

1.What is Safety tools in automotive [June 2023]

2. Safety of the automation industry. what would you do if a system fails and others. [June 2023]

3. What is Embedded C [June 2023]

4. What is bootloader? [June 2023]

1. **Basic Electronics [ Filters, Rectifiers, etc]**

1.What is frequency reuse [June 2023]

1. **Control Systems**