**LAB5**

**Interrupts using msp430 microcontroller and interrupt based Knight rider**

****

**Spring 2025**

Submitted by: **Mohsin Sajjad**

Registration No: **22pwsce2149**

Class Section: **A**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”



Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Engr. Faheem Jan**

Month Day, Year (23 03, 2025)

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

**Interrupt:**

• Anexternal event that interrupt the microcontroller that a device need its service

• Whenever a device needs its service, the device notify the microcontroller by sending it an interrupt signal

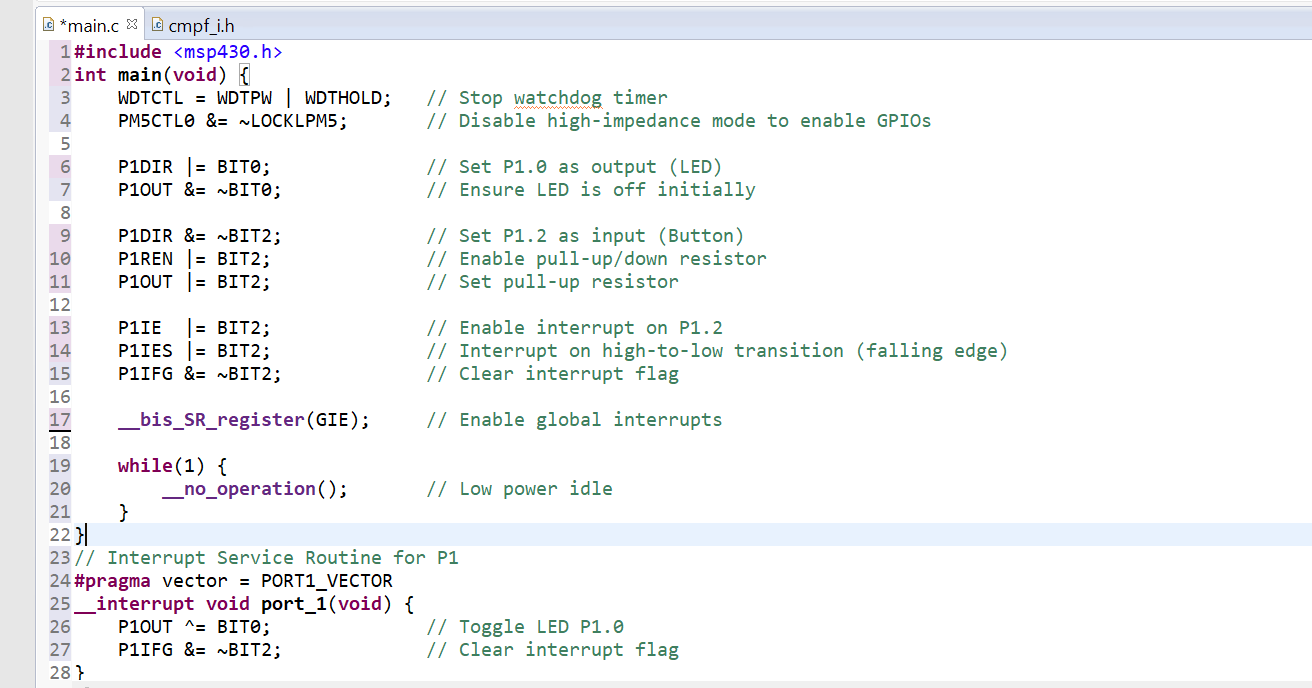
• Uponreceiving The microcontroller interrupt whatever it is doing and service the device

• The program associated with interrupt is called ISR(Interrupt Service Routine)

**TASKS:**

**TASK 01:**

Write Msp430 program which toggle LED P1.0 when an interrupt occur at P1.2



**Output:**

****

**Conclusion:**

This MSP430 program sets up P1.2 as a button input and P1.0 as an LED output. When the button is pressed, an interrupt is triggered, toggling the LED state.

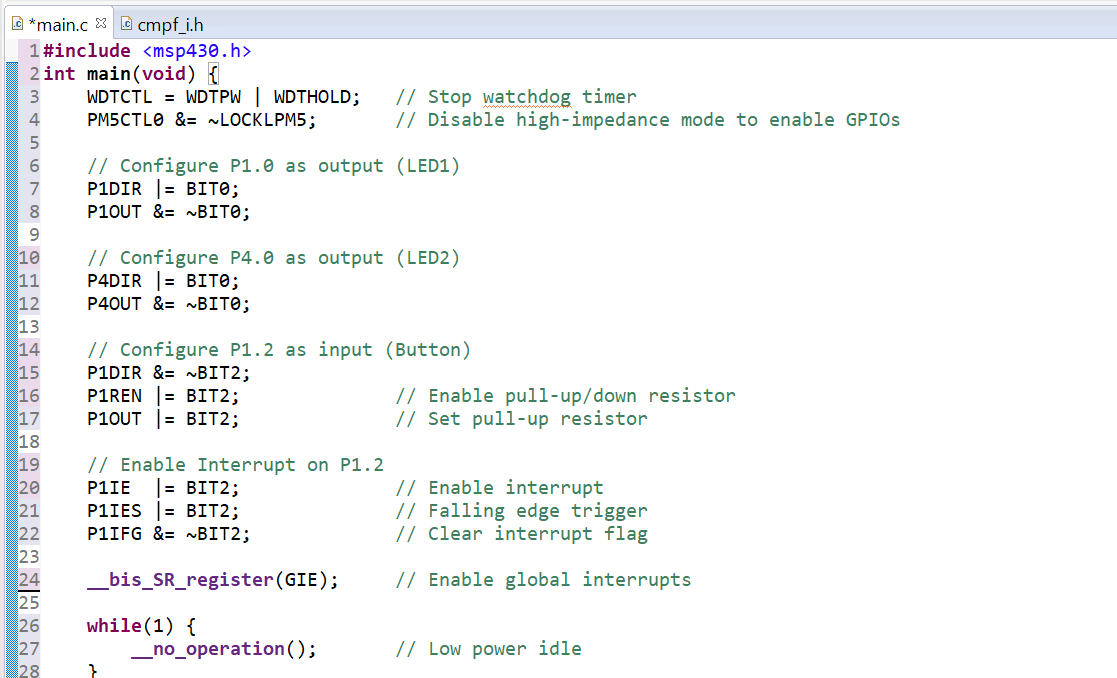
1. Stops the watchdog timer and unlocks GPIOs.
2. Configures the LED (P1.0) as output and ensures it's initially off.
3. Sets up P1.2 as an input with a pull-up resistor.
4. Enables interrupts on P1.2, triggering on a button press (falling edge).
5. Interrupt Service Routine (ISR) toggles the LED whenever the button is pressed.

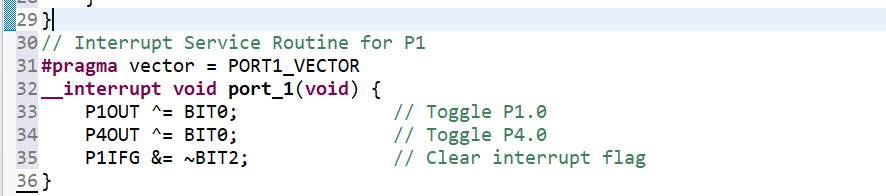
This is a basic interrupt-based button press LED toggle program for low-power embedded systems.

**Task 02:**

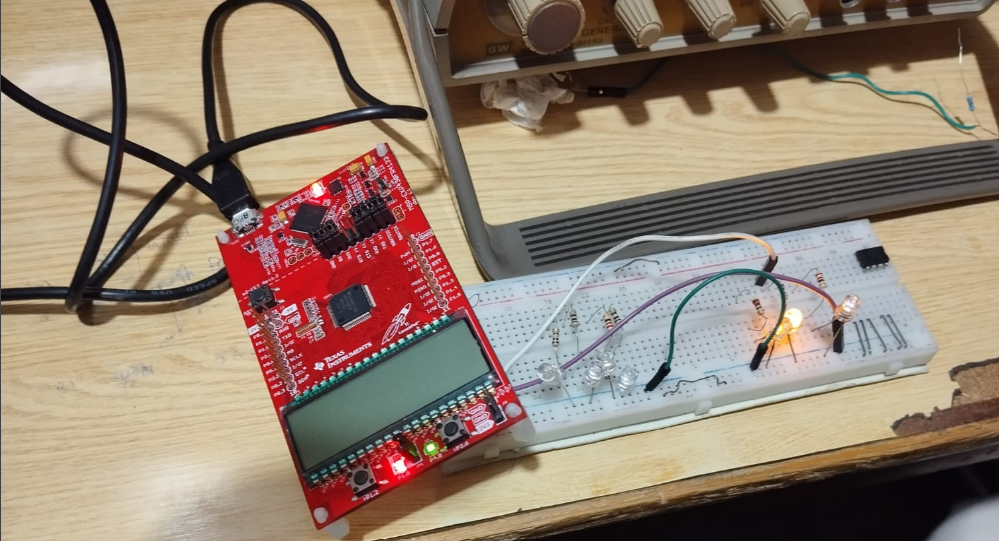
**Toggle LED p1.0 and P4.0 when an interrupt occurs at P1.2**

**CODE:**

****

****

**Output:**

****

**Conclusion:**

This MSP430 program controls two LEDs (P1.0 and P4.0) using a button (P1.2) with an interrupt-based approach.

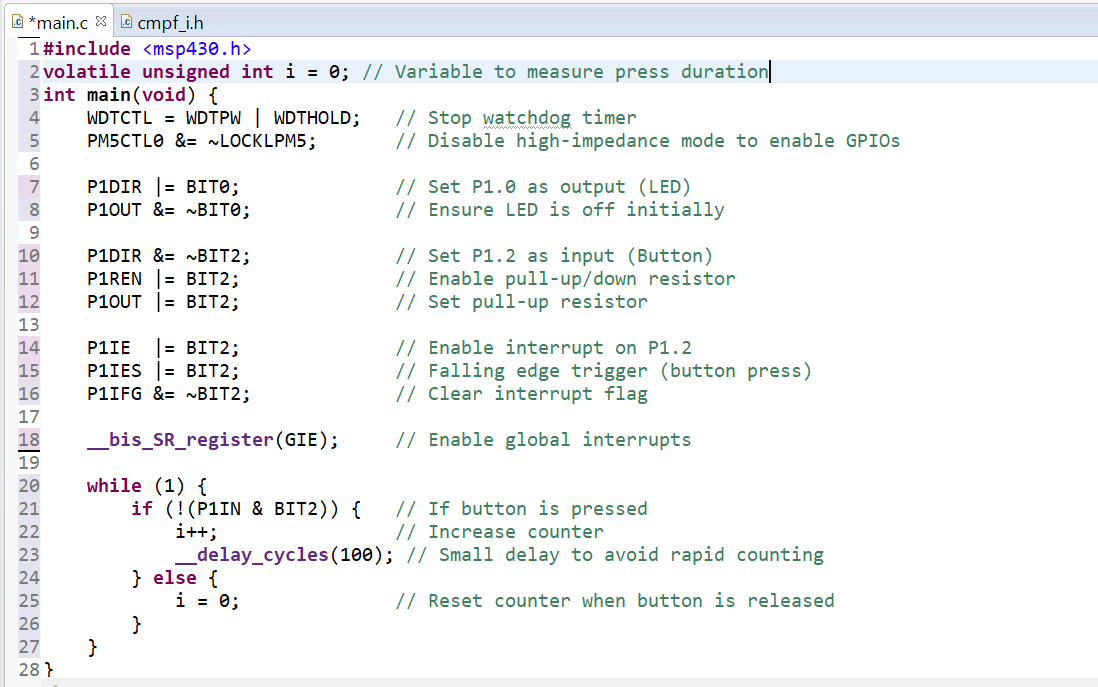
1. Stops the watchdog timer and unlocks GPIOs.
2. Configures P1.0 (LED1) and P4.0 (LED2) as outputs, initially turning them off.
3. Sets up P1.2 as an input with a pull-up resistor for the button.
4. Enables an interrupt on P1.2, triggering on a button press (falling edge).
5. The ISR toggles both LEDs whenever the button is pressed.

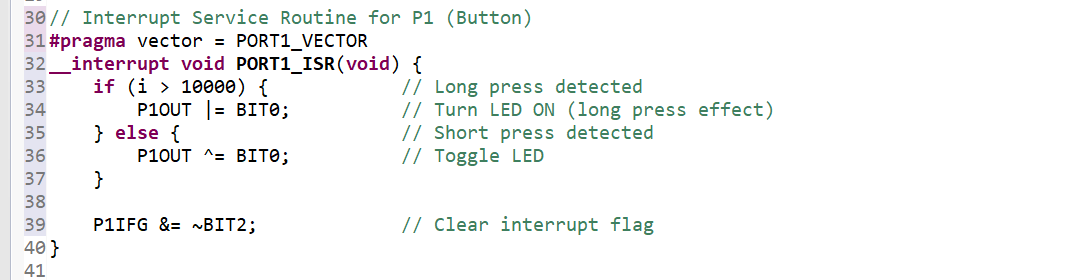
This allows low-power operation and ensures LEDs change state only when the button is pressed.

**Task 03:**

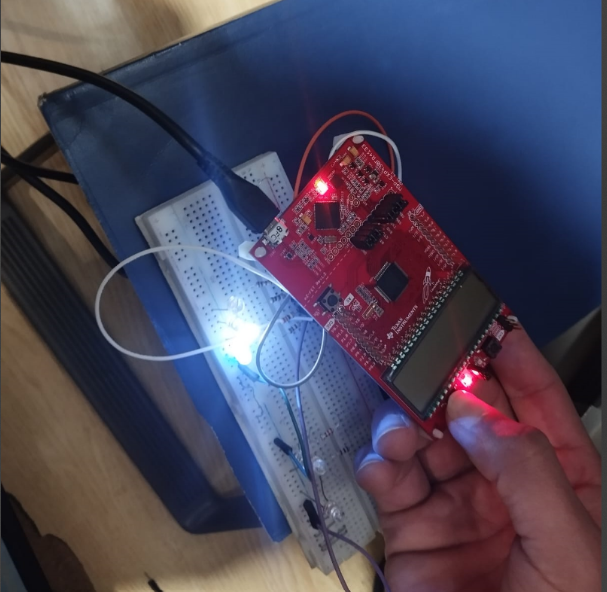
**Detect short or long press of button if button is pressed for longer time the LED should remain ON if the button is pressed for a shorter time it should toggle.**

**CODE:**

****

****

**Output:**

****

**Conclusion:**

This MSP430 program controls an LED (P1.0) based on button press duration (P1.2) using an interrupt-based approach and a counter.

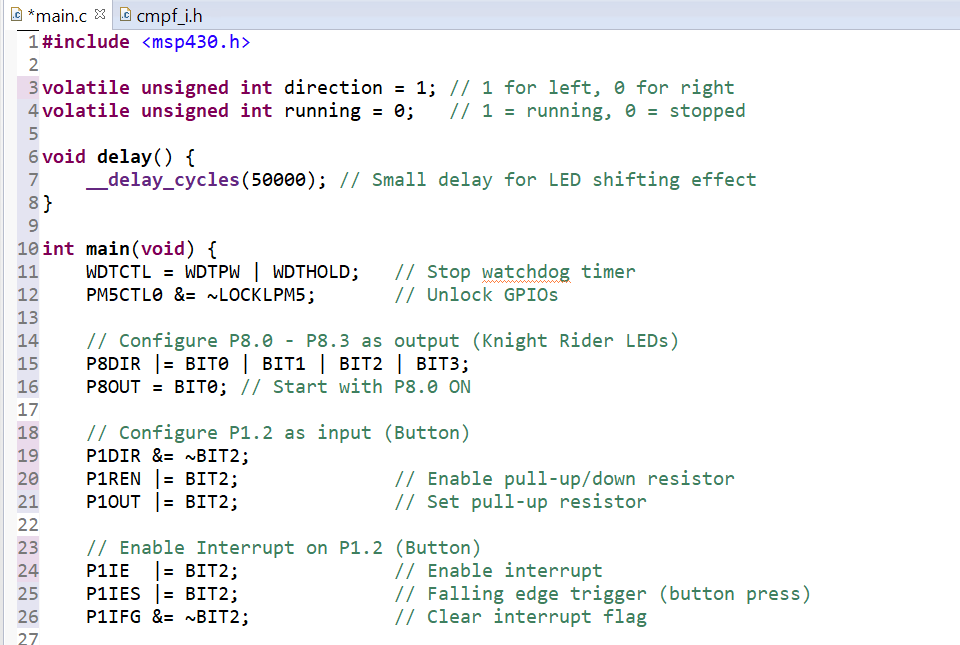
1. Stops watchdog timer and unlocks GPIOs.
2. Configures P1.0 as an output (LED) and P1.2 as an input (Button) with a pull-up resistor.
3. Enables an interrupt on P1.2 (falling edge trigger).
4. In the main loop, a counter (i) tracks button press duration.
5. In the ISR:
   * If pressed for a long time (i > 10000), the LED stays ON.
   * If pressed briefly, the LED toggles.

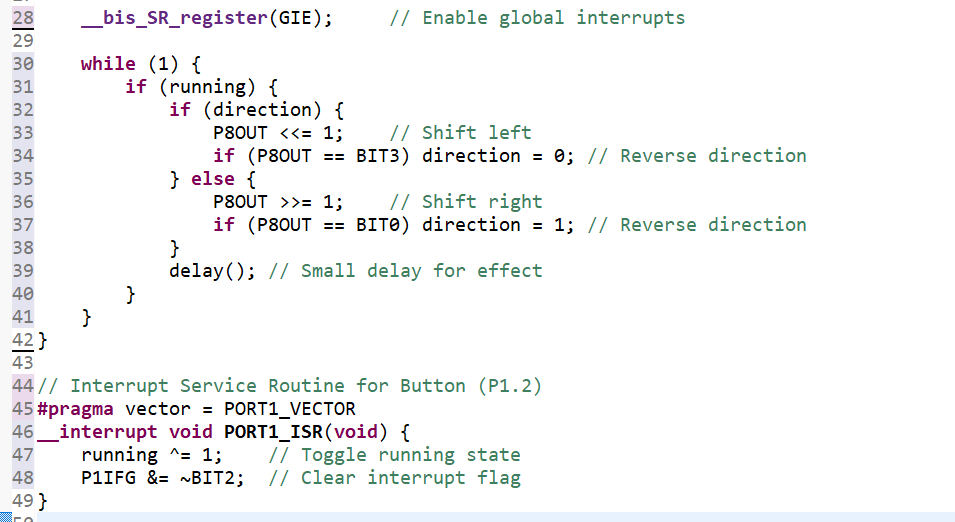
This allows short and long press detection, making the LED behave differently based on press duration.

**Task 04:**

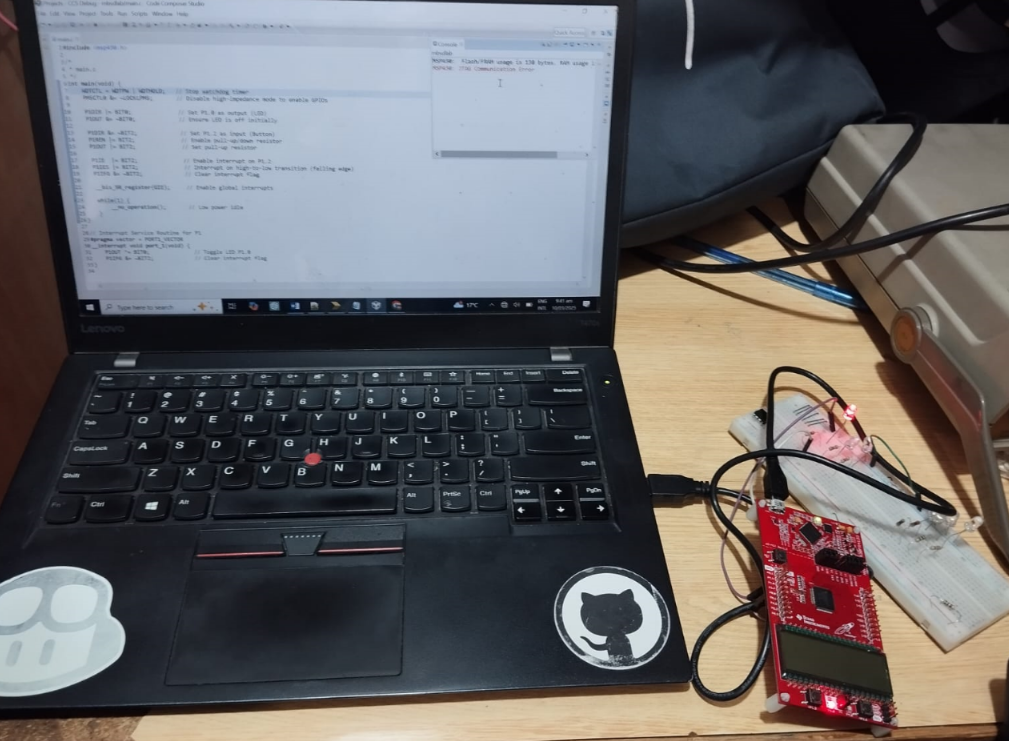
**Write code for interrupt based knight rider.**

**CODE:**

****

****

**Output:**

****

**Conclusion:**

This Knight Rider LED effect code for the MSP430 uses interrupts to toggle LED movement on P8.0 - P8.3 when a button (P1.2) is pressed. The LEDs shift left and right smoothly, reversing direction at the edges. The interrupt ensures the effect can be started or stopped dynamically, making it an efficient and interactive implementation.