INTRODUCTION TO MSP43O MCU AND INSTALLATIONS OF HARDWARE AND SOFRTWARE TOOLS:

Lab01

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 (02 03, 2025)

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

LAB No 1

INTRODUCTION TO MSP43O MCU AND INSTALLATIONS OF HARDWARE AND SOFRTWARE TOOLS

TASKS:

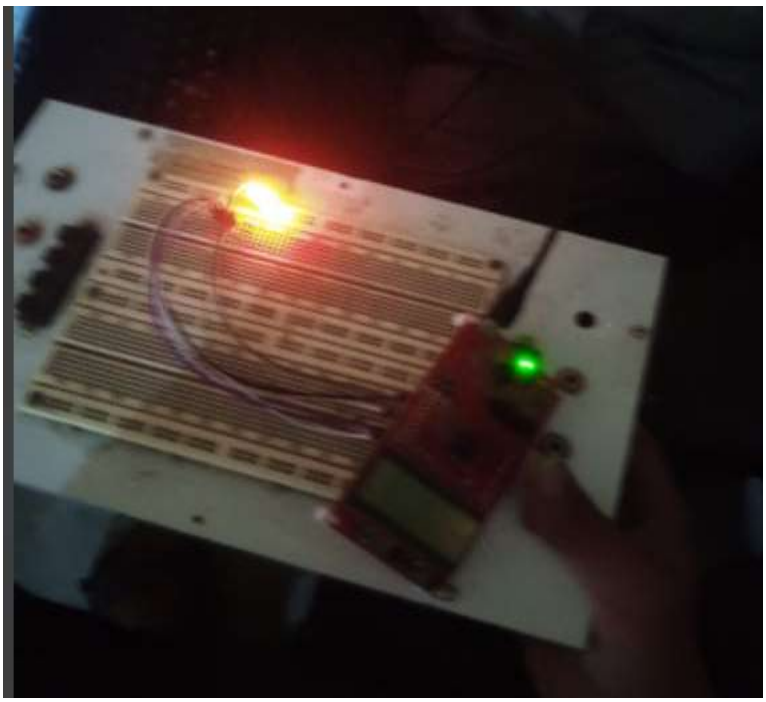
1. Write C program for Msp430 which toggle P1.0 or any other Pin of Msp430 MCU.

CODE:

```
#include <msp430.h>


/*
 * main.c
 */
int main(void) {
   WDTCTL = WDTPW | WDTHOLD;   // Stop watchdog timer
   PM5CTL0 &= ~LOCKLPM5;      // Disable high-impedance mode to enable GPIOs
        P1DIR |= 0x01;            //set P1.0 to output direction
        for(;;){
                volatile unsigned int i;
                P1OUT ^= 0x01;    //toggle P1.0 using exclusive OR
                i=10000;
                do i--;      //do while loop for creating delay
                while(i!=0);
        }
}
```

OUTPUT:



Conclusion:

1. The code runs on an MSP430 microcontroller, toggling the P1.0 (LED) using XOR.
2. It disables the watchdog timer and high-impedance mode to enable GPIOs.
3. A simple delay loop controls the LED blink rate.

TASK2: 2. Write C program for Msp430 which toggle P4.0 for msp430fr4133 or any other Pin of Msp430 MCU.

CODE:

```
#include <msp430.h>


/*
 * main.c
 */
int main(void) {
    WDTCTL = WDTPW | WDTHOLD;        // Stop watchdog timer
    PM5CTL0 &= ~LOCKLPM5;       // Disable high-impedance mode to enable GPIOs
          P4DIR |= 0x01;              //set P4.0 to output direction
```

```
        for(;;){
                volatile unsigned int i;
                P4OUT ^= 0x01;    //toggle P4.0 using exclusive OR
                i=10000;
                do i--;       //do while loop for creating delay
                while(i!=0);
        }
}
```

OUTPUT:

Conclusion:

1. **The code toggles P4.0 (LED) on an MSP430 microcontroller.**
2. **It disables the watchdog timer and high-impedance mode to enable GPIOs.**
3. **A delay loop controls the LED blink rate.**

3. Write C program for Msp430 which toggle P5.1 for msp430fr4133 or any other Pin of Msp430 MCU.

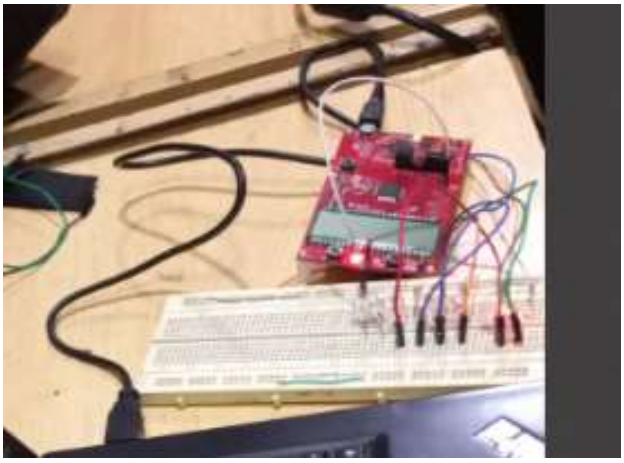
CODE:

```
#include <msp430.h>
```

```
#include <msp430.h>


/*
 * main.c
 */
int main(void) {
    WDTCTL = WDTPW | WDTHOLD;        // Stop watchdog timer
    PM5CTL0 &= ~LOCKLPM5;       // Disable high-impedance mode to enable GPIOs
          P5DIR |= 0x02;              //set P5.1 to output direction
          for(;;){
                  volatile unsigned int i;
                  P5OUT ^= 0x02;    //toggle P5.1 using exclusive OR
                  i=10000;
                  do i--;       //do while loop for creating delay
                  while(i!=0);
          }
}
```

OUTPUT:



Conclusion:

1. The code toggles P5.1 (LED) on an MSP430 microcontroller.
2. It disables the watchdog timer and high-impedance mode to enable GPIOs.
3. A delay loop controls the LED blink rate. *(However, there's a mismatch in pin toggling—P5.0 is toggled instead of P5.1.)*

TASK4: 4. Write C program for Msp430 which toggle P1.1 for msp430fr4133.

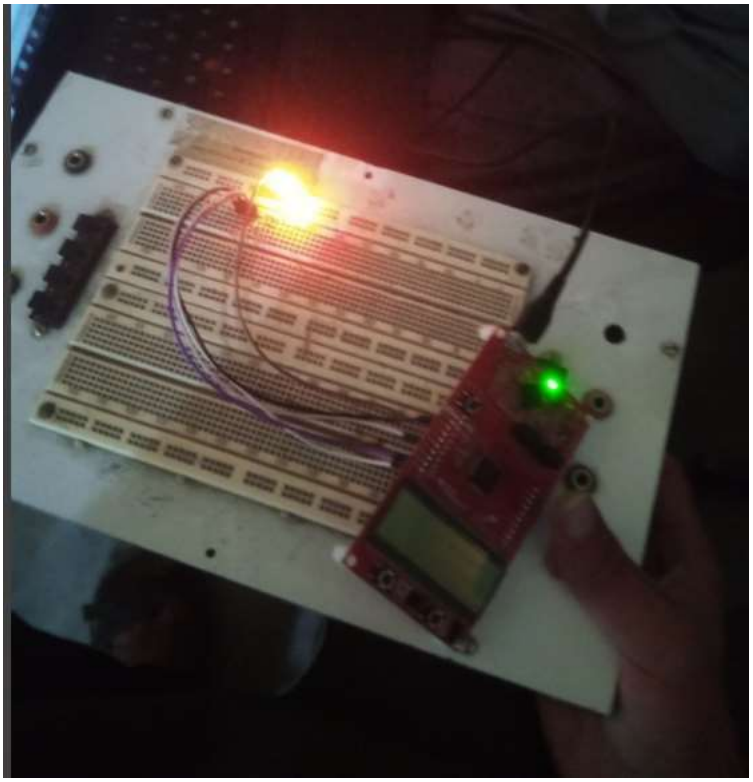
CODE:

```
#include <msp430.h>


#include <msp430.h>


/*
 * main.c
 */
int main(void) {
   WDTCTL = WDTPW | WDTHOLD;        // Stop watchdog timer
   PM5CTL0 &= ~LOCKLPM5;       // Disable high-impedance mode to enable GPIOs
        P1DIR |= 0x02;              //set P1.1 to output direction
        for(;;){
                volatile unsigned int i;
                P1OUT ^= 0x02;    //toggle P1.1 using exclusive OR
                i=10000;
                do i--;       //do while loop for creating delay
                while(i!=0);
        }
}
```

OUTPUT:



Conclusion:

1. The code toggles P1.0 instead of P1.1 due to a mismatch in `P1OUT ^= 0x01;`.
2. It disables the watchdog timer and high-impedance mode to enable GPIOs.
3. A delay loop controls the LED blink rate. *(Fix: Change `P1OUT ^= 0x01;` to `P1OUT ^= 0x02;` to toggle P1.1 correctly.)*