

1.1 Objective

The main objective of this experiment is to blink the on-board, red LED (connected to P1.0) using GPIO. This experiment will help you to learn and understand the procedure for programming the MSP-EXP430G2 LaunchPad digital I/O pins.

1.2 Introduction

1.2.1 Digital I/O

MSP430G2553 devices have up to eight digital I/O ports (P1 to P8). Each port has up to eight I/O pins that are configured with user software. Every I/O pin is individually configurable as input or output, and can be individually read or written to.

The features of the digital I/O are:

- Independently programmable individual I/Os
- Any combination of input or output
- Individually configurable P1 and P2 interrupts
- Independent input and output data registers
- Individually configurable pull-up or pull-down resistors
- Individually configurable pin-oscillator function in some MSP430 devices

The MSP430G2553 has two general-purpose digital I/O pins connected to green LED (P1.6) and red LED (P1.0) on the MSP-EXP430G2 LaunchPad for visual feedback. In this experiment, the code programmed into the MSP430G2553 processor toggles the output on Port P1.0 at fixed time intervals computed within the code. A HIGH on P1.0 turns the LED ON, while a LOW on P1.0 turns the LED OFF. Thus, the toggling output blinks the red LED connected to it. The functional block diagram shown in [Figure 1-1](#) illustrates the working principle of the experiment.

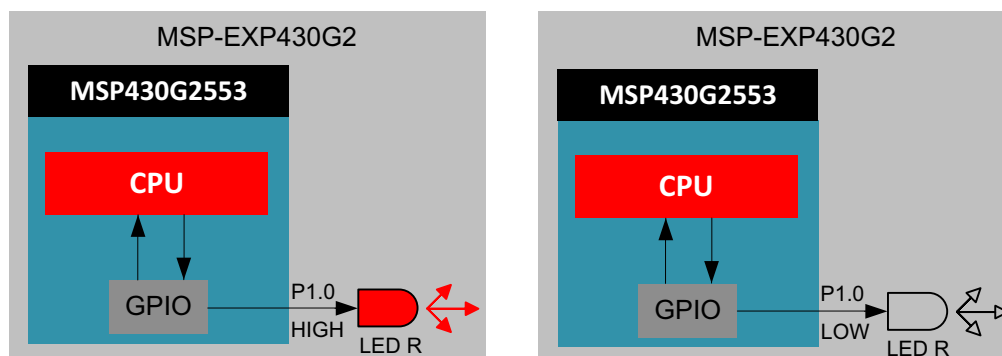


Figure 1-1 Functional Block Diagram

[Figure 1-2](#) shows the Launchpad schematics for the connection of Digital I/O pins to the on-board LEDs in the LaunchPad.

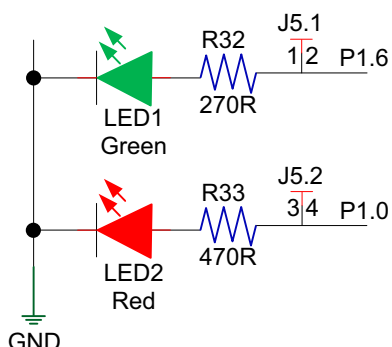


Figure 1-2 LaunchPad Schematics

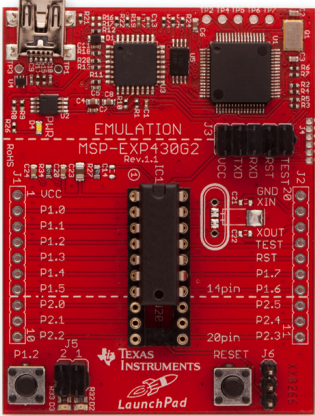
1.3 Component Requirements

1.3.1 Software Requirement

[Code Composer Studio](#)

1.3.2 Hardware Requirement

Table 1-1: Components Required for the Experiment

S.No	Components	Specifications	Images
1.	MSP430G2 Launch-Pad	MSP-EXP430G2XL	
2.	USB cable		

1.4 Software

The software for the experiment is written in C and developed using the CCS Integrated Development Environment (IDE). Refer to “[Project Creation and Build](#)” in the Getting Started section of this manual for project build and debug using CCS. The software is programmed into the target device MSP430G2553 on the MSP-EXP430G2 LaunchPad using the USB interface.

1.4.1 Flowchart

The flowchart for the code is shown in [Figure 1-3](#). The program code first disables the watchdog timer to prevent a processor restart on expiry of the WDT count. The port pin P1.0 connected to

the red LED is configured as output. A HIGH on the pin turns the red LED on, while a LOW turns the LED off. The P1.0 output is toggled using bit XOR function at regular intervals determined by the software delay set in the program using variable 'i'. Thus, the red LED on the MSP-EXP430G2 LaunchPad blinks at regular intervals.

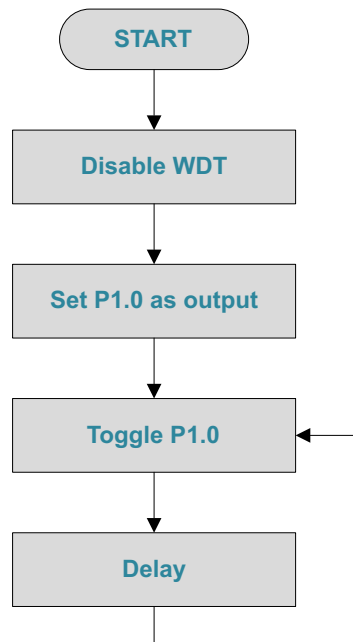


Figure 1-3 Flowchart to Blink the LED Using GPIO

1.4.2 C Program Code to Blink the LED with GPIO

```

#include<msp430.h>

int main(void) {
    WDTCTL = WDTPW | WDTHOLD;           // Stop watchdog timer
    P1DIR |= 0x01;                       // Set P1.0 to output direction

    while(1) {
        volatile unsigned long i;       // Volatile to prevent
                                         // optimization

        P1OUT ^= 0x01;                  // Toggle P1.0 using XOR
        i = 50000;                       // SW Delay
        do i--;
        while(i != 0);
    }
}
  
```

1.4.3 Registers Used

The registers used in this program are:

- **WDTCTL: 16-bit Watchdog Timer Control Register** - The Watchdog Timer (WDT) is typically used to trigger a system reset after a certain amount of time. In most examples, the timer is stopped during the first line of code to prevent a continuous restart loop and for ease of debugging. In our program, the instruction `WDTCTL = WDTPW | WDTHOLD` sets the password (WDTPW) and WDTHOLD bit to stop the timer.

WDTCTL, Watchdog Timer + Register

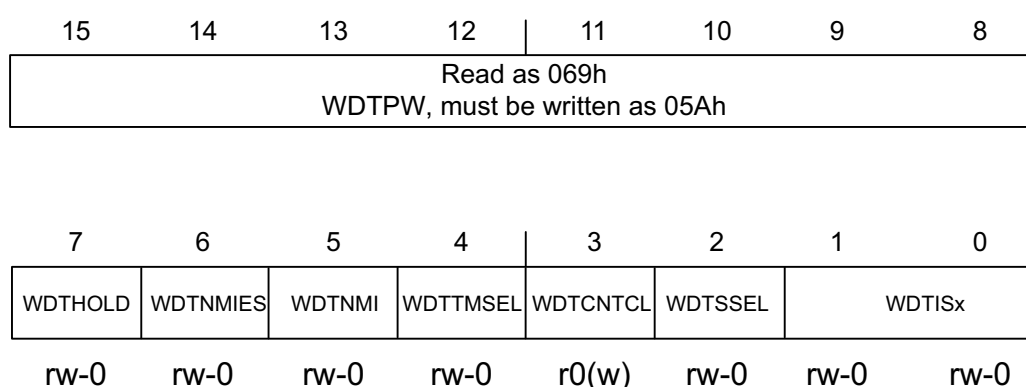


Figure 1-4 Watchdog Timer Control Register

- **P1DIR: 8-bit Direction Register** - Each bit in the direction register selects the direction of the corresponding pin as an input or an output. Here, bit 0 is set to "1" directing the Port Pin 1.0 as an output.
- **P1OUT: 8-bit Output Register** - Each bit in the output register is the value to be output on the corresponding I/O pin when the pin is configured as an output. A bit 1 sets the pin HIGH, while a 0 resets the pin to LOW.

1.5 Procedure

1. Connect the MSP-EXP430G2 LaunchPad to the PC using the USB cable supplied.
2. Build, program and debug the code into the LaunchPad using CCS to view the status of the red LED.
3. In the CCS debug perspective, select **View** ➔ **Registers**.

1.6 Observation

While the code is running, pause it. In the **Registers** tab of CCS, expand and view the registers P1OUT, P1DIR and WDTCTL. Compare the values with your code. The value of the registers when the LED is ON appears in the CCS watch window as shown in [Figure 1-5](#).

(x)= Variables Expressions Registers		
Name	Value	Description
P1	1	P1
P0	1	P0
P1DIR	0x01	Port 1 Direction [Memory Mapped]

Figure 1-5 Register values in CCS Watch Window for LED On

The value of the registers when the LED is OFF appears in the CCS watch window as shown in [Figure 1-6](#).

(x)= Variables Expressions Registers		
Name	Value	Description
P1	1	P1
P0	0	P0
P1DIR	0x01	Port 1 Direction [Memory Mapped]

Figure 1-6 Register values in CCS Watch Window for LED Off

1.7 Summary

In this experiment, we have learnt to configure and program the digital I/O pins of MSP430G2553 and have successfully programmed the port P1.0 to blink the on-board green LED of MSP430G2 LaunchPad.

1.8 Exercise

1. Alter the delay with which the LED blinks.
2. Alter the code to make the green LED blink.
3. Alter the code to make the green and red LEDs blink:
 - i. Together
 - ii. Alternately

Experiment 2
LED Control Using a Switch

Topics	Page
2.1 Objective	26
2.2 Introduction.....	26
2.3 Component Requirements.....	28
2.4 Software	28
2.5 Procedure.....	30
2.6 Observation.....	30
2.7 Summary	31
2.8 Exercise	31

2.1 Objective

The main objective of this experiment is to control the on-board, green LED of MSP430G2 Launchpad by an input from the on-board switch of MSP430G2 Launchpad. This experiment will help you to learn and understand the procedure for programming the MSP430G2553 GPIO pins as input and output.

2.2 Introduction

2.2.1 Digital I/O

MSP430 devices have up to eight digital I/O ports (P1 to P8). Each port has up to eight I/O pins that are configured with user software. Every I/O pin is individually configurable as input or output, and can be individually read or written to.

The features of the digital I/O are:

- Independently programmable individual I/Os
- Any combination of input or output
- Individually configurable P1 and P2 interrupts
- Independent input and output data registers
- Individually configurable pull-up or pull-down resistors
- Individually configurable pin-oscillator function in some MSP430 devices

The MSP-EXP430G2 LaunchPad has two general-purpose digital I/O pins connected to green and red LEDs for visual feedback. It also has two push buttons for user feedback and device reset. In this experiment, the input from the left push button switch (S2) connected to Port P1.3 is read by the processor for LED control. If the switch is pressed, the green LED is turned OFF by output at port P1.6 reset to '0'. Else, the output at port P1.6 is set HIGH and the green LED is turned ON. The LED is therefore controlled by the switch S2.

The functional block diagram shown in [Figure 2-1](#) illustrates the working principle of the experiment.

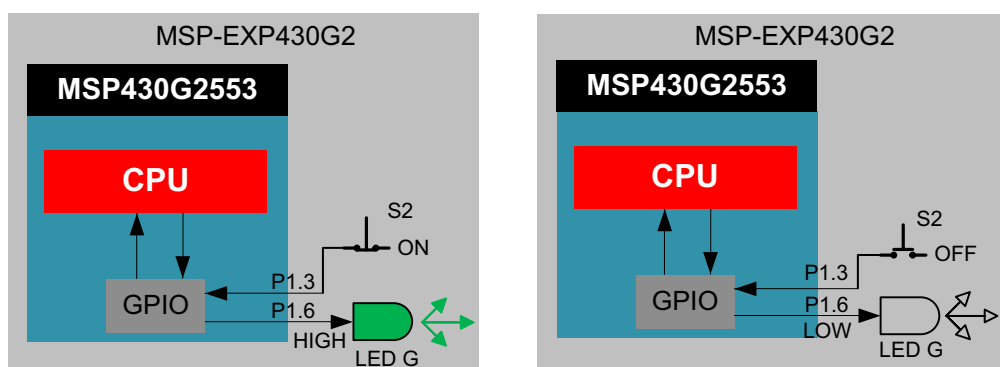


Figure 2-1 Functional Block Diagram

[Figure 2-2](#) shows the Launchpad schematics for the connection of Digital I/O pins to the on-board LEDs in the LaunchPad.

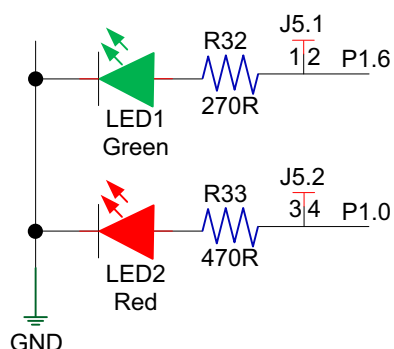


Figure 2-2 Launchpad Schematics

The input pins for the MSP430G2553 need to be either pulled-up or pulled-down with a resistor to avoid floating inputs. The selection of a pull-up or pull down resistor for the inputs is performed in software. A pull-up resistor enables a default hardware connection for the pin to V_{CC} as shown in [Figure 2-3](#). Hence, when the switch is open, the V_{CC} (logic HIGH) is available on the input pin of the MCU.

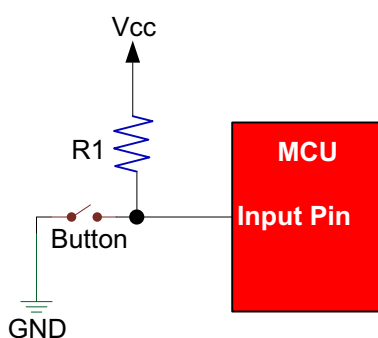


Figure 2-3 Pull-up Resistor for Input Pin

Similarly, a pull-down resistor defaults a hardware connection to ground, 0V as shown in [Figure 2-4](#). When the switch is open, 0V GND (logic LOW) is available on the input pin of the MCU.

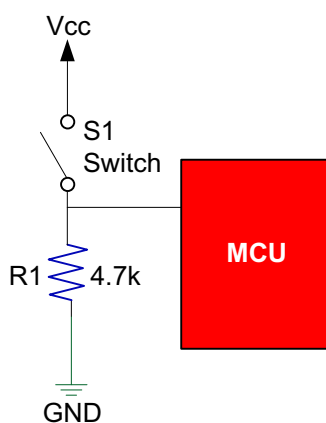


Figure 2-4 Pull-down Resistor for Input Pin

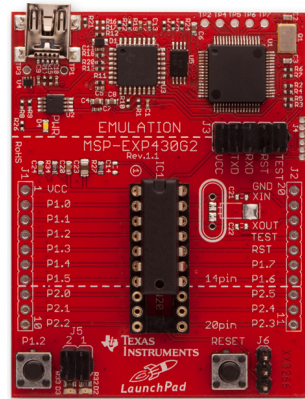
2.3 Component Requirements

2.3.1 Software Requirement

[Code Composer Studio](#)

2.3.2 Hardware Requirement

Table 2-1: Components Required for the Experiment

S.No	Components	Specifications	Images
1.	MSP430G2 LaunchPad	MSP-EXP430G2XL	
2.	USB cable		

2.4 Software

The software for the experiment is written in C and developed using the CCS Integrated Development Environment (IDE). Refer to “[Project Creation and Build](#)” in the Getting Started section of this manual for project build and debug using CCS. The software is programmed into the target device MSP430G2553 on the MSP-EXP430G2 LaunchPad using the USB interface.

2.4.1 Flowchart

The flowchart for the code is shown in [Figure 2-5](#). The program code first disables the watchdog timer to prevent a restart on expiry of the WDT count. The port pin P1.6 connected to the green LED is configured as output and port pin P1.3 connected to the left push button switch (SW2) is configured as input with pull up resistor.

The push button input at P1.3 is read continuously using a while(1) infinite loop. When the push button switch is open, input at P1.3 will be HIGH because of the pull up resistor, and when the button is pressed, the switch is closed and the input at pin P1.3 is LOW.

If the push button switch is open, the code turns the green LED ON with a HIGH on output pin P1.6. Else, if the push button switch is closed, the code turns the green LED OFF with a LOW on output pin P1.6.

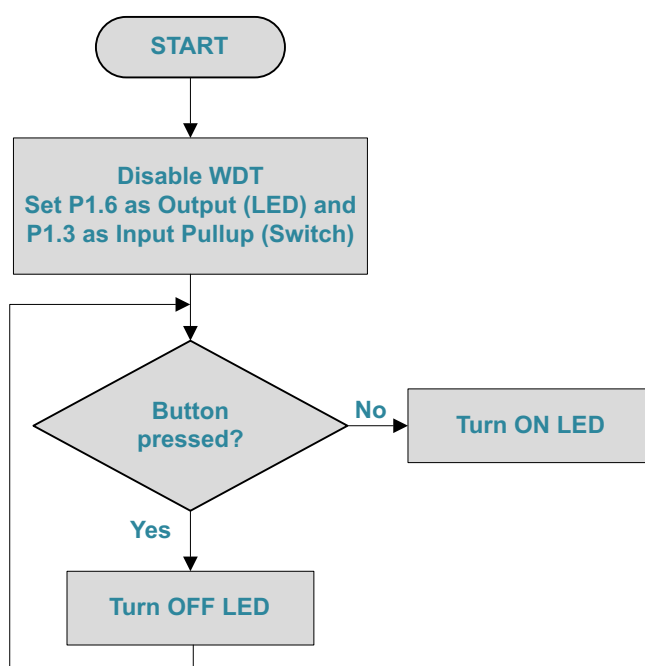


Figure 2-5 Flowchart for Controlling LED with a Switch

2.4.2 C Program Code for Controlling the LED with a Switch

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

```
int main(void) {
    WDTCTL = WDTPW | WDTHOLD;           // Stop watchdog timer
    P1DIR |= 0x40;                       // Set P1.6 to output direction
    P1REN |= 0x08;
    P1OUT |= 0x08;
    while(1) {
        if ((P1IN & BIT3)) {             // If button is open(P1.3 HIGH)
            P1OUT = P1OUT | BIT6;        // ... turn on LED
            // or P1OUT |= BIT0;
        }
        else {
            P1OUT = P1OUT & ~BIT6;       // ... else turn it off.
            // or P1OUT &= ~BIT0
        }
    }
}
```

2.4.3 Registers Used

The registers used in this program are:

- **WDTCTL: 16-bit Watchdog Timer Control Register** - The Watchdog Timer (WDT) is typically used to trigger a system reset after a certain amount of time. In most examples, the timer is stopped during the first line of code to prevent a continuous restart loop.
- **P1DIR: 8-bit Direction Register** - Each bit in the direction register selects the direction of the corresponding pin as an input or an output. Here, bit 6 is set to 1 directing the Port Pin 1.6 as an output.
- **P1REN: Resistor Enable Register** - It is used to specify configuration of pull-up or pull-down to the input pin. Here, bit 3 is set to enable a pull-up resistor on switch input pin P1.3.
- **P1OUT: 8-bit Output Register** - Each bit in the P1OUT register is the value to be output on the corresponding I/O pin when the pin is configured as an output, and the pull-up/down resistor is disabled. A bit 1 sets the pin HIGH, while a 0 resets the pin to LOW. In our program, since the pull-up resistor is enabled on pin P1.3, the corresponding input pin on the P1OUT register is set.

2.5 Procedure

1. Connect the MSP-EXP430G2 LaunchPad to the PC using the USB cable supplied.
2. Build, program and debug the code into the LaunchPad using CCS.

2.6 Observation

Initially, the green LED is ON as shown in [Figure 2-6](#). When the left push button switch is pressed, the green LED turns OFF.



Figure 2-6 Green LED Turned On in MSP-EXP430G2 LaunchPad

2.7 Summary

In this experiment, we have learnt to configure and program the GPIO pins of MSP430G2553 as input and output. We have successfully programmed to read input from the on-board switch of MSP430G2 Launchpad connected to pin P1.3 to blink the on-board green led of MSP430G2 Launchpad connected to pin P1.6.

2.8 Exercise

1. Alter the code to turn the LED ON when the button is pressed and OFF when it is released.
2. Alter the code to make the green LED stay ON for around 1 second every time the button is pressed.
3. Alter the code to turn the red LED ON when the button is pressed and the green LED ON when the button is released.