MSP430 Stopwatch and LCD Display Project

Embedded Systems Lab Report

September 22, 2025

Chapter 1

Introduction

This lab exercise focuses on two applications using the MSP430FR2433 microcontroller:

- A stopwatch system using **Timer_A Capture Mode** and pushbutton input.
- Keypad and LCD interfacing to display a user's phone number.

The goals are to gain understanding of timer capture functionality, GPIO interrupts, and peripheral interfacing.

Chapter 2

Part 1: Stopwatch Using Pushbutton and Timer_A Capture

Design Requirements

- Use pushbutton on **P2.7** to control stopwatch.
- Stopwatch starts on button press and stops on button release.
- Use jumper wire to connect **P2.7** to **P1.2**, which is the `TA0.CCI2A` capture input.
- Light up:
  - Red LED (P1.0) when stopwatch starts.
  - Green LED (P6.6) when stopwatch stops.

Implementation

The timer is configured in **capture mode** on `TA0CCR2`. On the button press (rising edge), the current timer value is stored, and on release (falling edge), the timer value is captured again. The difference gives elapsed time.

```
1  /* Stopwatch using pushbutton P2.7 and Timer_A Capture
2   * Red LED (P1.0) ON at start
3   * Green LED (P6.6) ON at stop
4   */
5
6  #include <msp430.h>
7
8  volatile unsigned int start_time = 0;
9  volatile unsigned int stop_time = 0;
10 volatile unsigned int elapsed_time = 0;
11
12 int main(void)
13 {
14     WDTCTL = WDTPW | WDTHOLD;    // Stop watchdog timer
15     PM5CTL0 &= ~LOCKLPM5;        // Enable GPIO
16
```

```
    // LEDs
    P1DIR |= BIT0;  // Red LED
    P6DIR |= BIT6;  // Green LED
    P1OUT &= ~BIT0;
    P6OUT &= ~BIT6;

    // Configure P1.2 for TA0.CCI2A capture input
    P1SEL0 |= BIT2;
    P1SEL1 &= ~BIT2;

    // Configure Timer_A
    TA0CTL = TASSEL__SMCLK | MC__CONTINUOUS | TACLR; // SMCLK, continuous mode
    TA0CCTL2 = CM_3 | CCIS_0 | CAP | CCIE;  // Capture on both edges, CCI2A, enable interrupt

    __bis_SR_register(GIE);   // Enable global interrupts

    while (1) {
        __no_operation();     // Wait for capture ISR
    }
}

// Timer_A Capture ISR
#pragma vector = TIMER0_A1_VECTOR
__interrupt void Timer_A_Capture(void)
{
    if (TA0IV == TA0IV_TACCR2) {
        if (P2IN & BIT7) { // Button pressed
            start_time = TA0CCR2;
            P1OUT |= BIT0;   // Red LED ON
            P6OUT &= ~BIT6;  // Green LED OFF
        } else { // Button released
            stop_time = TA0CCR2;
            elapsed_time = stop_time - start_time;
            P6OUT |= BIT6;   // Green LED ON
            P1OUT &= ~BIT0;  // Red LED OFF
        }
    }
}
```

Explanation

- `CM_3` configures capture on both rising and falling edges.
- On press, the start time is recorded, red LED lights.
- On release, stop time is recorded, elapsed time is calculated, green LED lights.
- The elapsed time can later be displayed on an LCD if needed.

Chapter 3

Part 2: Display Phone Number on LCD Using Keypad

Design Requirements

- Use 4x4 matrix keypad connected to GPIOs.
- Accept numeric key inputs.
- Display pressed digits on character LCD.
- Enter complete phone number and show it in real-time.

Implementation Approach

1. Configure GPIO rows as outputs and columns as inputs with pull-ups.
2. Perform keypad scanning: set one row low at a time and read columns.
3. Map pressed key to corresponding digit.
4. Send character to LCD via 4-bit or 8-bit interface.

```
/* Keypad and LCD phone number display */

#include <msp430.h>
#include "lcd.h"    // Assume lcd.h contains LCD init and send
   functions

char phone_number[11];
unsigned int index = 0;

char scan_keypad(void); // Prototype

int main(void)
{
    WDTCTL = WDTPW | WDTHOLD;    // Stop watchdog timer
    PM5CTL0 &= ~LOCKLPM5;        // Enable GPIO
```

```
15
16      LCD_init();
17      LCD_clear();
18
19      while (1) {
20          char key = scan_keypad();
21          if (key != 0 && index < 10) {
22              phone_number[index++] = key;
23              LCD_data(key); // Display on LCD
24          }
25      }
26  }
27
28  char scan_keypad(void)
29  {
30      // Example: configure P3.0-P3.3 as rows, P3.4-P3.7 as columns
31      // Return digit character '0'-'9' if pressed
32      // Implementation depends on specific wiring
33      return 0; // Placeholder
34  }
```

Explanation

- Keypad scanning detects which button is pressed.
- The corresponding digit is stored and sent to LCD.
- The user enters digits of their phone number, which appear sequentially on LCD.

Chapter 4

Conclusion

This lab combined two important microcontroller concepts:

- Using **Timer_A Capture** to measure button press duration and implement a stopwatch.
- Interfacing with peripherals: **keypad scanning** and **LCD display**.

The stopwatch application demonstrated timer interrupts and GPIO control, while the keypad-LCD application demonstrated data input and user feedback on embedded systems.