# 2DX4: Microprocessor System
# Lab 5

## Instructor: Drs.Boursalie, Doyle, Haddara

Lab TAs: Fatemeh Derakshani (derakhsf), Han Zhou (zhouh115)

Junbo Wang – L01 – wangj430

Yichen Lu – L01 – luy191

As a future member of the engineering profession, the student is responsible for performing the required work in an honest manner, without plagiarism and cheating. Submitting this work with my name and student number is a statement and understanding that this work is our own and adheres to the Academic Integrity Policy of McMaster University and the Code of Conduct of the Professional Engineers of Ontario. Submitted by **[Junbo Wang  wangj430  400249823]**

As a future member of the engineering profession, the student is responsible for performing the required work in an honest manner, without plagiarism and cheating. Submitting this work with my name and student number is a statement and understanding that this work is our own and adheres to the Academic Integrity Policy of McMaster University and the Code of Conduct of the Professional Engineers of Ontario. Submitted by **[Yichen Lu  luy191  400247938]**

## 1. Purpose

The purpose of this lab is to start interfacing peripherals and integrating embedded concepts. We will decode the keypad input in this experiment. The keyboard must be connected by hardware and software to create the functional user interface.

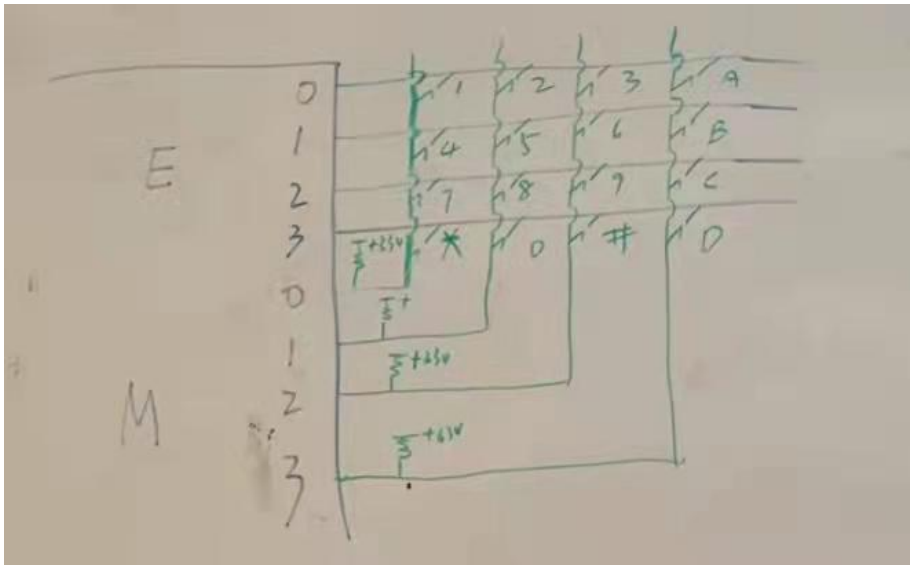
## 2. BackGround

There are three milestones in this lab and they are all related to the keypad. In studio 5, it shows how different keypads work to control the onboard LEDs. We interface a row of 2 buttons with the microcontroller and know that keys are in active low, when we press the button, a 0 will be driven to a certain port and light up the LED.

## 3. Method

**Milestone 1**

We identify the scan input and output code through the following figure, that is, the 4*4 keyboard interface. Since the key is active low, when we press the input button, a 0 will be driven to that port. For example, when we press the value 1, the scanning input code for port M is 1110, and the scanning output code for port E is 1110.



**Milestone 2 & Milestone 3**

According to the reference code, we saw that the input port M, output port E, and the function of lighting the LED were given. We created variables to represent the return values and wrote several while loops for all the input keys in the keyboard. We ran the code and entered debug mode, and when we pressed the key, we could see that the return value was the same as what we pressed. At the same time, the LEDs corresponding to the input binary value were illuminated. For example, when we press the value 1, which is active low 1110, LED D1 will light up.

## 4. Results

In milestone 1, we can determine the value of all key presses through the 4*4 keyboard interface to get scanning input and output codes. For milestones 2 and 3, we ran the code successfully and turned into debug mode. When we pressed the buttons on the keypad, the values were shown in the table. For instance, when we pressed the button 7, the value of 7 was shown in this table and onboard LEDs D1, D2, D3 shined, it represented the binary value 7 which is 0111.

Table 6.3: Key Press Decoding

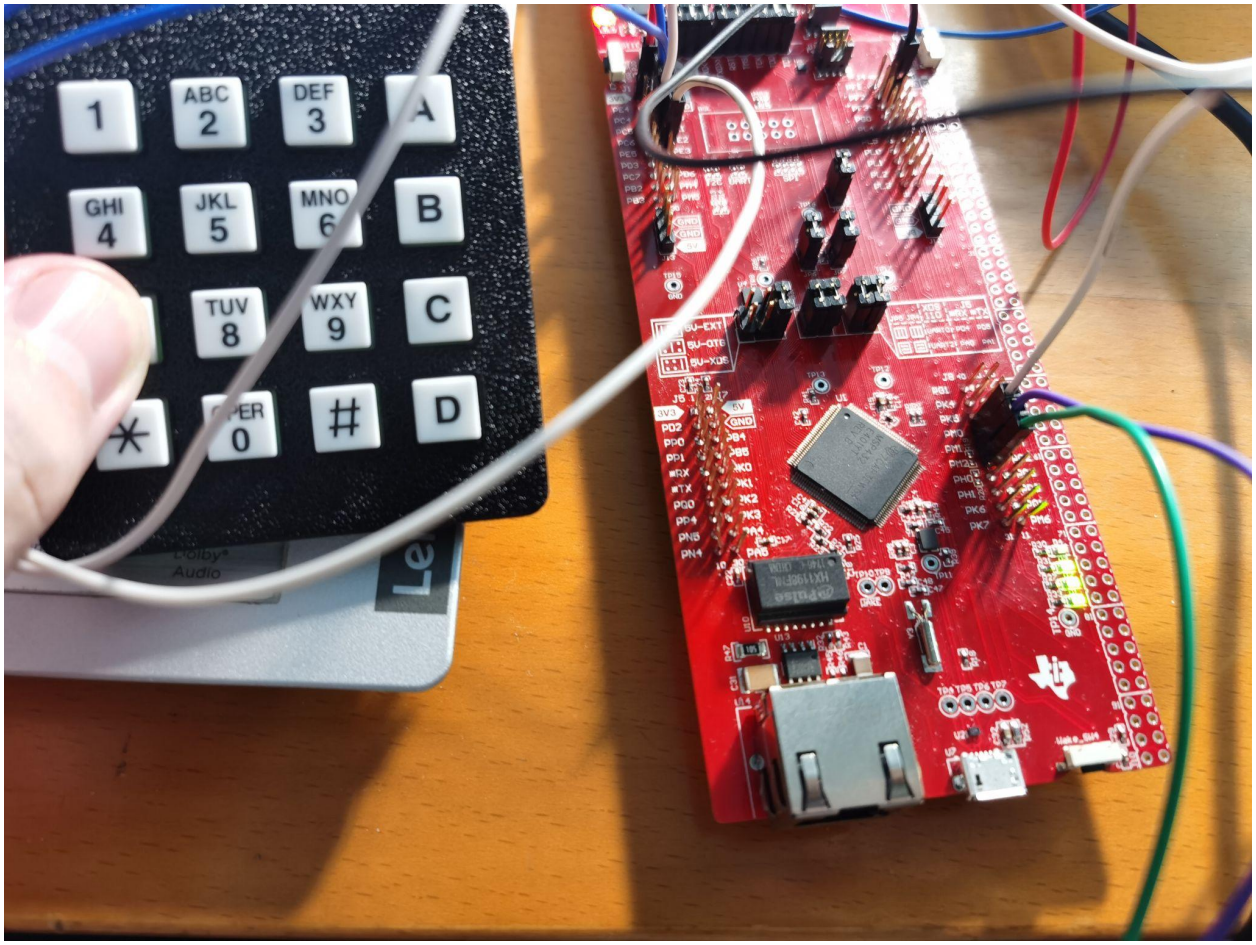
| Key Pressed | Binary Key Value | Scanning Input Code PM[3:0] | Scanning Output Code PE[3:0] |
|---|---|---|---|
| 0 | 0000 | | |
| 1 | 0001 | | |
| 2 | 0010 | | |
| 3 | 0011 | | |
| 4 | 0100 | | |
| 5 | 0101 | | |
| 6 | 0110 | | |
| 7 | 0111 | | |
| 8 | 1000 | | |
| 9 | 1001 | | |
| A | 1010 | | |
| B | 1011 | | |
| C | 1100 | | |
| D | 1101 | | |
| * | 1110 | | |
| # | 1111 | | |

Watch 1

| Name | Value | Type |
| --- | --- | --- |
| b | 0x00000007 | int |
| <Enter expression> | | |

Call Stack + Locals | Watch 1 | Memory 1

CMSIS-DAP Debugger t1: 149.02696820 sec



## 5. Observations and conclusions

In this lab, we connected the keypad with the microcontroller and computed the codes in software. The keypad was connected by the hardware and software to create interface and work. In milestone 1, we determined the scanning input and output codes through the 4*4 keypad interface. For milestones 2 and 3, we created several functions in while loops and decoded the input buttons from the keypad to show in the table. Also, we observed that the onboard LEDs shined which matched the input we pushed.

# 6. Reference

## Code Appendix

```
// 2DX4StudioW30E1_Decoding a Button Press
// This program illustrates detecting a single button press.
// This program uses code directly from your course textbook.
//
// This example will be extended for in W21E0 and W21E1.
//
//  Written by Ama Simons
//  January 30, 2020
//  Last Update: January 21, 2020

// Name: Yichen Lu Junbo Wang
// Student id: 400247938 400249823
// Date: Feb, 28th, 2022


#include <stdint.h>
#include "tm4c1294ncpdt.h"
#include "Systick.h"
#include "PLL.h"

void PortE0E1_Init(void){
    SYSCTL_RCGCGPIO_R |= SYSCTL_RCGCGPIO_R4;                        // activate
the clock for Port E
    while((SYSCTL_PRGPIO_R&SYSCTL_PRGPIO_R4) == 0){};            // allow time
for clock to stabilize
  GPIO_PORTE_DEN_R = 0b00001111;                                    // Enabled
both as digital outputs
    return;
    }


void PortM0M1_Init(void){
    SYSCTL_RCGCGPIO_R |= SYSCTL_RCGCGPIO_R11;                  //activate the
clock for Port M
    while((SYSCTL_PRGPIO_R&SYSCTL_PRGPIO_R11) == 0){};         //allow time for
clock to stabilize
    GPIO_PORTM_DIR_R = 0b00000000;                                          //
make PM0 an input, PM0 is reading if the button is pressed or not
  GPIO_PORTM_DEN_R = 0b00001111;
    return;
}


//Turns on D2, D1
void PortN0N1_Init(void){
```

```
    SYSCTL_RCGCGPIO_R |= SYSCTL_RCGCGPIO_R12;                  //activate the
clock for Port N
    while((SYSCTL_PRGPIO_R&SYSCTL_PRGPIO_R12) == 0){};
    GPIO_PORTN_DIR_R=0b00000011;
    GPIO_PORTN_DEN_R=0b00000011;
    return;
}

//Turns on D3, D4
void PortF0F4_Init(void){
  SYSCTL_RCGCGPIO_R |= SYSCTL_RCGCGPIO_R5;                  //activate the
clock for Port F
    while((SYSCTL_PRGPIO_R&SYSCTL_PRGPIO_R5) == 0){};
    GPIO_PORTF_DIR_R=0b00010001;
    GPIO_PORTF_DEN_R=0b00010001;
    return;
}




int main(void){
    //PLL_Init();
    //SysTick_Init();
    PortE0E1_Init();
    PortM0M1_Init();
    PortN0N1_Init();
    PortF0F4_Init();
    volatile int a=0b00000000;
    volatile int b=0b00000000;

    while(1){


//Row 1

   GPIO_PORTE_DIR_R  =  0b00000001;            // To drive you use the data
direction register
   GPIO_PORTE_DATA_R =  0b00000000;

    //Checks If Button 3 is pressed - D2 lights up
    //Unique code is:   1010 - In the order of PE1 PE0 PM1 PM0
    while((GPIO_PORTM_DATA_R&0b00000001)==0){
    GPIO_PORTN_DATA_R = 0b00000010;
    a=0b11101110;
    b=0b0001;
    }
    a=0;
    b=0;
```

```
    //Checks If Button 4 is pressed - D1 lights up
    //Unique code is: 1001 - In the order of PE1 PE0 PM1 PM0
  while((GPIO_PORTM_DATA_R&0b00000010)==0){
      GPIO_PORTN_DATA_R = 0b00000001;
        a=0b11011110;
        b=0b0010;
    }
    a=0;
    b=0;

    while((GPIO_PORTM_DATA_R&0b00000100)==0){
      GPIO_PORTN_DATA_R = 0b00000011;
        a=0b10111110;
        b=0b0011;
    }
    a=0;
    b=0;

    while((GPIO_PORTM_DATA_R&0b00001000)==0){
      GPIO_PORTN_DATA_R = 0b00000001;
        GPIO_PORTF_DATA_R = 0b00000001;
        a=0b01111110;
        b=0b1010;
    }
    a=0;
    b=0;
    //Row 2

    GPIO_PORTE_DIR_R  =  0b00000010;              //Drive Row 2
    GPIO_PORTE_DATA_R =  0b00000000;
//Data is still as registers


    //Checks if Button 1 is pressed - D3 lights up
    //Unique code is: 0110 - In order of PE1 PE0 PM1 PM0
  while((GPIO_PORTM_DATA_R&0b00000001)==0){
    GPIO_PORTF_DATA_R=0b000010000;
        a=0b11101101;
        b=0b0100;
    }
    a=0;
    b=0;


//Checks if Button 2 is pressed - D4 Lights up
    //Unique Code is: 0101  - In order of PE1 PE0 PM1 PM0
```

```
 while((GPIO_PORTM_DATA_R&0b00000010)==0){
        GPIO_PORTF_DATA_R=0b000010000;
      GPIO_PORTN_DATA_R=0b000000010;
        a=0b11011101;
        b=0b0101;
    }
    a=0;
    b=0;

    while((GPIO_PORTM_DATA_R&0b00000100)==0){
        GPIO_PORTF_DATA_R=0b000010000;
      GPIO_PORTN_DATA_R=0b000000001;
        a=0b10111101;
        b=0b0110;
    }
    a=0;
    b=0;

    while((GPIO_PORTM_DATA_R&0b00001000)==0){
        GPIO_PORTF_DATA_R=0b000000001;
      GPIO_PORTN_DATA_R=0b000000011;
        a=0b01111101;
        b=0b1011;
    }
    a=0;
    b=0;

//Row 3
      GPIO_PORTE_DIR_R  =  0b00000100;             //Drive Row 2
    GPIO_PORTE_DATA_R =  0b00000000;

        while((GPIO_PORTM_DATA_R&0b00000001)==0){
            GPIO_PORTF_DATA_R=0b000010000;
            GPIO_PORTN_DATA_R=0b000000011;
            a=0b11101011;
            b=0b0111;
        }
        a=0;
        b=0;

        while((GPIO_PORTM_DATA_R&0b00000010)==0){
            GPIO_PORTF_DATA_R=0b000000001;
            a=0b11011011;
            b=0b1000;
        }
        a=0;
        b=0;

        while((GPIO_PORTM_DATA_R&0b00000100)==0){
```

```
            GPIO_PORTF_DATA_R=0b000000001;
            GPIO_PORTN_DATA_R=0b000000010;
            a=0b10111011;
            b=0b1001;
        }
        a=0;
        b=0;

        while((GPIO_PORTM_DATA_R&0b00001000)==0){
            GPIO_PORTF_DATA_R=0b000010001;
            a=0b01111011;
            b=1100;
        }
        a=0;
        b=0;

//Row 4
        GPIO_PORTE_DIR_R  =  0b00001000;              //Drive Row 2
    GPIO_PORTE_DATA_R =  0b00000000;

        while((GPIO_PORTM_DATA_R&0b00000001)==0){
            GPIO_PORTF_DATA_R=0b000010001;
            GPIO_PORTN_DATA_R=0b000000001;
            a=0b11100111;
            b=0b1110;
        }
        a=0;
        b=0;

        while((GPIO_PORTM_DATA_R&0b00000010)==0){
            GPIO_PORTF_DATA_R=0b000000000;
            a=0b11010111;
            b=0b0000;
        }
        a=0;
        b=0;

        while((GPIO_PORTM_DATA_R&0b00000100)==0){
            GPIO_PORTF_DATA_R=0b000010001;
            GPIO_PORTN_DATA_R=0b000000011;
            a=0b10110111;
            b=0b1111;
        }
        a=0;
        b=0;

        while((GPIO_PORTM_DATA_R&0b00001000)==0){
            GPIO_PORTF_DATA_R=0b000010001;
            GPIO_PORTN_DATA_R=0b000000010;
```

```
            a=0b01110111;
            b=0b1101;
        }
        a=0;
        b=0;


GPIO_PORTN_DATA_R=0b00000000;
GPIO_PORTF_DATA_R=0b00000000;

}
}
```