EECS 3215 Digital Logic Design
Lab 4
Michael Founk
213641204
Sean O’Brien
213735741

Problem Statement

The objective of this lab was to design and create a control system for a mixer. The basis of it is that there are 2 liquids that are being mixed together. The system uses 3 sensors to measure the height of the liquid. The first sensor is triggers at 80% of the container height, the second sensor is triggered at 90% of the container height and the last sensor is triggered when the container is overflowing. We are tasked to create this system and show the different stages with LEDs being turned on.

If the container is less than 80% full then there are no LEDs that are to be turned on. When the container is more than 80% full and the first sensor is triggered, then the green LED is turned on. When the container is more than 90% full and the second sensor is also triggered then both the red and green LEDs are turned on. When the last sensor triggers, when the container is overflowing, then both the green and red LEDs are supposed to be blinking. In any other case, when the sensors are malfunctioning, the red LED should blink.

Top-Level Design

The design we opted for was a series of else-if statements which would handle the various combinations of sensor inputs. The cases which had specific outcomes were handled explicitly, with the else statement at the end handling the invalid combinations of inputs. We utilized the two on-board push-buttons to represent two of the sensors, as well as a wire connecting a pin and a ground on the board to represent the other sensor. The push-buttons were active-low but to simplify the problem we inverted the values in the logic so that when no buttons were pressed, it was the case that no sensors were active. When the wire is disconnected from the ground, our logic treated it as if the sensor was warning of overflow.

Instead of the Pin Assignment tool, we opted to use the actual names of the pins in our assignments. We enabled the clock for the ports we used, assigned the pins we were using as GPIO and defined the data direction of our pins. In order to blink our LEDs we used a self-defined delay function, and we used the PSOR and PCOR to clear and set our bits.

Code

```
#include <stdio.h>
#include "board.h"
#include "peripherals.h"
#include "pin_mux.h"
#include "clock_config.h"
#include "MKL43Z4.h"
#include "fsl_debug_console.h"
#include<stdlib.h>
void delay(unsigned int mseconds)
{
        int c, d;

          for (c = 1; c <= 1000; c++)
```

```
            for (d = 1; d <= 1000; d++)
            {}

        return 0;
}


int main (void) {

        SysTick->LOAD = 0xFFFFFF;
        SysTick->CTRL = 0x05;

        SIM->SCGC5 |= SIM_SCGC5_PORTC_MASK;
        SIM->SCGC5 |= SIM_SCGC5_PORTD_MASK;
        SIM->SCGC5 |= SIM_SCGC5_PORTE_MASK;
        SIM->SCGC5 |= SIM_SCGC5_PORTA_MASK;

        PORTE->PCR[31] = 0x100;
        PORTD->PCR[5] = 0x100;

        PTE->PDDR |= 1<<31u;
        PTD->PDDR |= 1<<5u;

        PTE->PSOR |= 1<<31u;
        PTD->PSOR |= 1<<5u;

        PTC->PDDR &= 1<<3u;
        PORTC->PCR[3] = 0x103;

        PTA->PDDR &= 1<<4u;
        PORTA->PCR[4] = 0x103;

        PTA->PDDR &= 0<<1u;
        PORTA->PCR[1] = 0x103;


        while (1) {
                if (PTC->PDIR & 0b1000 && PTA->PDIR & 0b10000 && !(PTA->PDIR &
0b10)){
                        PTD -> PSOR |= 1<<5u;
                        PTE -> PSOR |= 1<<31u;
                }else if(!(PTA->PDIR & 0b10000) && PTC->PDIR & 0b1000  && !(PTA->PDIR
& 0b10)){
                        PTD -> PCOR |= 1<<5u;
                        PTE -> PSOR |= 1<<31u;
                }else if(!(PTA->PDIR & 0b10000) && !(PTC->PDIR & 0b1000)  &&
!(PTA->PDIR & 0b10)){
                        PTD -> PCOR |= 1<<5u;
```

```
                PTE -> PCOR |= 1<<31u;
        }else if(PTA->PDIR & 0b10 && !(PTA->PDIR & 0b10000) && !(PTC->PDIR &
0b1000)){
                PTD -> PCOR |= 1<<5u;
                PTE -> PCOR |= 1<<31u;
                delay(1000);
                PTD -> PSOR |= 1<<5u;
                PTE -> PSOR |= 1<<31u;
                delay(1000);
        }else{
                PTD -> PSOR |= 1<<5u;
                PTE -> PCOR |= 1<<31u;
                delay(1000);
                PTE -> PSOR |= 1<<31u;
                delay(1000);
        }
    }
}
```

Reference Material

SupplementaryTextbook
https://electrovolt.ir/wp-content/uploads/2017/07/Freescale_ARM_Cortex_M_Embedded_ElectroVolt.ir_.pdf

FRDM kl43z User Manual
https://wiki.eecs.yorku.ca/course_archive/2018-19/W/3215/_media/frdmkl43zug.pdf

FRDM kl43 Family Reference Manual
https://www.nxp.com/docs/en/reference-manual/KL43P64M48SF6RM.pdf

FRDM kl43z Board Schematic
https://wiki.eecs.yorku.ca/course_archive/2018-19/W/3215/_media/frdm-kl43z_sch.pdf