Embedded Systems
Postlab 7 ADC

By Rajendra Singh (111601017), final year, CSE, IIT PALAKKAD

27 September 2019

Objective

To learn how to setup and use adc pin in the KL25Z Microcontroller using ARM Cortex M0+.

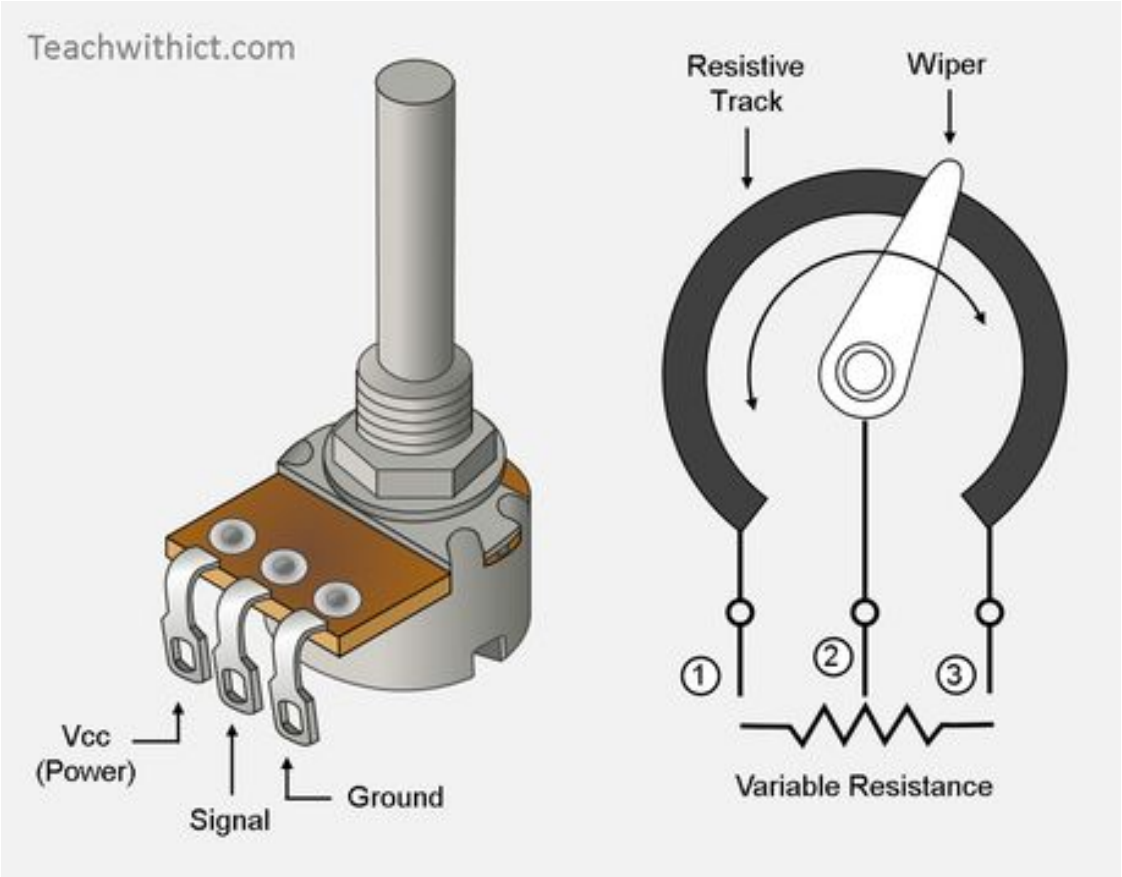
Abstract

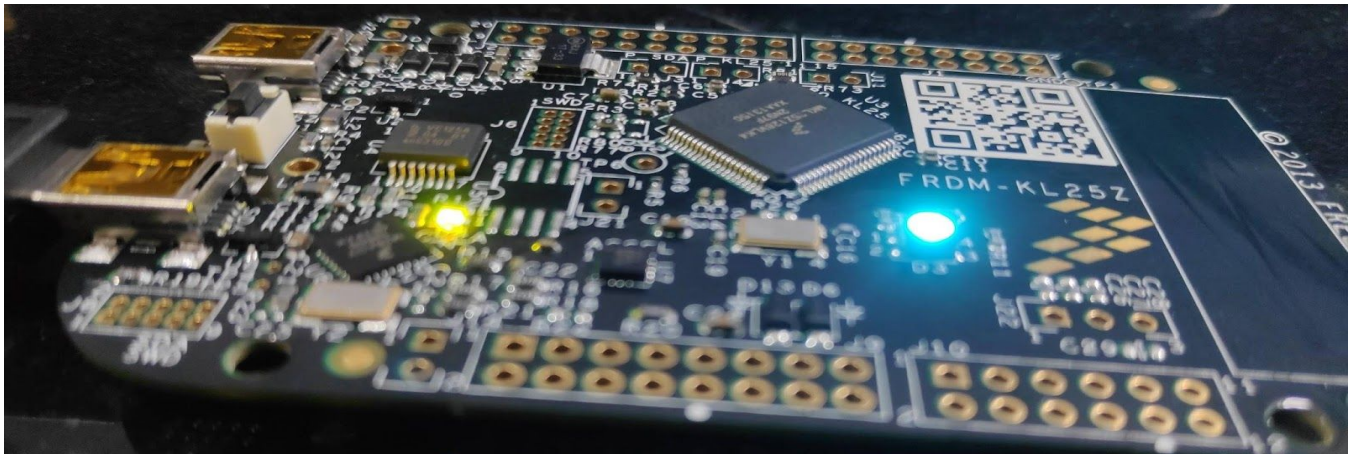
The Cortex-M0+ processor supports adc input and output. In this exercise we read the adc data from the potentiometer and based on this data we controlled the on board leds.

Introduction

Potentiometer output the adc data from 0 to 1024. We divided this range into three parts i.e

1. (0, 1024/3) => green led
2. (1024/3, 2*1024/3) => blue led
3. (2*1024, 1024) => red led


Teachwithict.com
Resistive Track
Wiper
Vcc (Power)
Signal
Ground
1
2
3
Variable Resistance


FRDM-KL25Z


Kindly look the code for more information regarding the port and pin selection.

EXERCISE

1 : Control the LED based on the potentiometer reading.

=> In below is detailed code with comments for each steps for controlling the led using potentiometer reading.

CODE:

Below code can also be found at github.

```
/*

	*	Author : Rajendra singh

	*	Roll no: 111601017

	*/



	#include<MKL25Z4.h> //INCLUDING LIBRARY
```

```
static volatile short result; //To store 16 bit value




//====================red B18==============================//

//=======init========//

void led_red_init(){

   SIM->SCGC5 |=(1<<10); //TO ACTIVATE PORT B OR ACTIVE PORT B CLOCK

   //SET 8,9,10 = 001 TO GPIO

   PORTB->PCR[18]|=(1<<8); //SETTING 8TH BIT TO 1

   PORTB->PCR[18]&=0xFFFFF9FF; //SETTING 9TH, 10TH BIT TO 0, OTHER UNCHANGED

   PTB->PDDR |= (1<<18); //18TH BIT = 1, TO ACTIVATE 18 PIN

}

//=======ON========//

void led_red_on(){ //!!! on on low

   PTB->PCOR |= (1<<18 ); //CLEAR 18PIN VALUE

}
```

```
//=======OFF========//

void led_red_off(){

   PTB->PDOR |= (1<<18); //CLEAR 18PIN VALUE

}

//=======TOGGLE========//

void led_red_toggle(){

   PTB->PTOR |= (1<<18); //CLEAR 18PIN VALUE

}




//==============================green B19============================//

//=======init========//

void led_green_init(){

        SIM->SCGC5 |=(1<<10); //TO ACTIVATE PORT B

        PORTB->PCR[19]|=(1<<8); //SETTING 8TH BIT TO 1

        PORTB->PCR[19]&=0xFFFFF9FF; //SETTING 9TH, 10TH BIT TO 0, OTHER UNCHANGED

        PTB->PDDR |= (1<<19); //18TH BIT = 1

}
```

```
//=======ON========//

void led_green_on(){

        PTB->PCOR |= (1<<19 ); //CLEAR 18PIN VALUE

}

//=======OFF========//

void led_green_off(){

        PTB->PDOR |= (1<<19); //CLEAR 18PIN VALUE

}

//=======TOGGLE========//

void led_green_toggle(){

        PTB->PTOR |= (1<<19); //CLEAR 18PIN VALUE

}




//============================blue D1============================//

//=======init========//

void led_blue_init(){

        SIM->SCGC5 |=(1<<12); //TO ACTIVATE PORT B
```

```
        PORTD->PCR[1]|=(1<<8); //SETTING 8TH BIT TO 1

        PORTD->PCR[1]&=0xFFFFF9FF; //SETTING 9TH, 10TH BIT TO 0, OTHER UNCHANGED

        PTD->PDDR |= (1<<1); //18TH BIT = 1

}

//=======ON========//

void led_blue_on(){

        PTD->PCOR |= (1<<1 ); //CLEAR 18PIN VALUE

}

//=======OFF========//

void led_blue_off(){

        PTD->PDOR |= (1<<1); //CLEAR 18PIN VALUE

}

//=======TOGGLE========//

void led_blue_toggle(){

        PTD->PTOR |= (1<<1); //CLEAR 18PIN VALUE

}
```

```
//============================DELAY old============================//

void delay_old(long long int d){

        while(d--);

}



//============================ADC0_init============================//

void ADC0_init(){

   //disable clock gating of adc0

   SIM->SCGC6 |= (1<<27);



   SIM->SCGC5 |= (1<<10); //set port B, enable clock to Port B

   //set 9th, 10th, and unset 8th

   PORTB->PCR[1] &= 0xFFFFF8FF; //set 8th, 9th , 10th bit to 0

   PORTB->PCR[1] |= (1<<8); //Unset 8th again



   //configure adc0 sc2

   ADC0->SC2 &=0xFFFFFFBC; //6th bit = 0 and 0th and 1th bit = 0 (for default voltage)
```

```
    //0th,1th   //00 Bus clock

    //2th,3th   //01 Selects the ADC resolution mode.

    //4th       //0 Short sample time.

    //5th,6th   //10 The divide ratio is 4 and the clock rate is (input clock)/4

    //7th       //0 Normal power configuration.

    //rest all  //0

    ADC0->CFG1 =0b01000100;

}

//============================MAIN============================//

int main(void){

    SystemCoreClockUpdate(); //updating clock from PLL



    ADC0_init();//initialise adc0



    //initialise all led

    led_red_init();
```

```
    led_blue_init();

    led_green_init();



    //initially turn off all led

    led_red_off();

    led_blue_off();

    led_green_off();



    long long int n=1e6; //number of delay



    while(1){



        ADC0->SC1[0] = 9;//selecting ad9 for coco flag of ptb1

        while(!(ADC0->SC1[0] & (1<<7))); //check coco flag of adc0_sc1 to see if we read the data
or not

        result = ADC0->R[0] & 0xFFF; //0 for A



        //dividing the range in 3 parts and checking
```

```
            if(result > (1<<13)/3){

                led_red_on();

                delay_old(n);

                led_red_off();

            }else if(result > (1<<12)/3){

                led_green_on();

                delay_old(n);

                led_green_off();

            }else{

                led_blue_on();

                delay_old(n);

                led_blue_off();

            }

        }

    }
```

Conclusion

- In this exercise we learnt more about the adc pin available on the board.
- We learnt to use the potentiometer.

- Overall it was nice exercise.

Reference

1) Google
2) **Cortex M0+ Generic User's Guide**
3) **Cortex M0+ Technical Reference Manual**