|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Course Code** | | **MEE306 Microcontroller** | | | |
| **Term Project** | | **Electric Motor Velocity Controller** | | | |
| **Related Learning Outcome:** | | **5** | | | |
| **Group Number:** | |  | | | |
| **Group Member ID** | **Group Member Name** | | **Group Member Surname** | | **Group Member Signature** |
| **200412042** | **Sultan** | | **Etyemez** | |  |
| **200412061** | **Halil Ümit** | | **Turgut** | |  |
| **200412009** | **Enes** | | **Fide** | |  |
| **200412032** | **Fidel Utku** | | **Bulut** | |  |
| **Grading Policy** | | | | | |
| Pre-Work (20%)  Necessary simulation / software application / code | | Lab Work (50%)  Application outcomes & properly-working circuit | | Post-Lab Quiz (30%)  Quiz aiming to test fulfillments of Lab Work | |
| **Pre-work Term Project Results (filled by Lab assistants)** | | | | | |
| **Pre-Work** (100%) | | | | | |
|  | | | | | |

**PRE- WORK TERM PROJECT - Electric Motor Velocity Controller**

İNTRODUCTİON  
  
In this project, we carried out a study on PI control in a microcontroller. We aimed to drive the motor for certain fixed values ​​by writing a code for P and I controllers. By inserting the codes we wrote into the microcontroller, we enabled communication with the Ardiunho, thus aiming to drive the engine for the KP and K values ​​we entered.

Code Part

For the first code we right receiver for mikrocontrroller.

#include <16F877A.h>

#device ADC=10

#include <stdlib.h>

#FUSES XT, NOWDT, NOPROTECT, NOBROWNOUT, NOLVP, NOPUT, NODEBUG, NOCPD

#use delay(crystal=16000000)

#include <lcd.c>

#use rs232 (baud=9600,xmit=PIN\_C6, rcv=PIN\_C7, parity=N, stop=1) // configuretion for serial communication

#DEFINE IN1 PIN\_C3

#DEFINE IN2 PIN\_C4

int counter = 0; // counter to be used

char strInput[4]; // string array for input characters

unsigned long int inputString; // inputString value to be used

unsigned long int revAngle = 0.0f;unsigned long int prevAngle = 0.0f;

unsigned long int integralError = 0.0f;

int8 dt = 0.2;

float Ki = 0.0f;

float Kp = 0.0f;

signed int16 error = 0;

float dx\_dt = 0;

int i = 0;

#int\_ext

void external\_interrupt()

{

revAngle++;

}

#int\_timer0

void tmr\_int()

{

set\_timer0(60);

i++;

if (i==16)

{

integralError = integralError + error \* dt;

dx\_dt = (revAngle - prevAngle)\*18.75f; // Caution here

prevAngle = revAngle;

i = 0;

}

}

void main()

{

unsigned int16 realSpeed = 0;

unsigned long int pot = 0;

unsigned long int referance = 0;

unsigned long int controlOut =0;

setup\_psp(PSP\_DISABLED)

setup\_timer\_1(T1\_DISABLED);

setup\_timer\_2(T1\_DISABLED,0,1);

setup\_CCP1(CCP\_OFF);

setup\_CCP2(CCP\_OFF);

lcd\_init();

lcd\_cursor\_on(TRUE);

port\_b\_pullups(TRUE);

enable\_interrupts(GLOBAL);

clear\_interrupt(int\_ext);

setup\_timer\_0(RTCC\_INTERNAL | RTCC\_DIV\_256);

set\_timer0(60);

enable\_interrupts(int\_timer0);

enable\_interrupts(int\_ext);

setup\_adc\_ports(AN0\_AN1\_AN3); //A0 A1 A3 are configured for analog input pin

setup\_adc(ADC\_CLOCK\_DIV\_32); //enable ADC and set clock for ADC

//set\_tris\_c(0x10000000); //set all portb pins as output

setup\_ccp1(CCP\_PWM); //4kHz PWM signal output at CCP1 pin 17

setup\_timer\_2(T2\_DIV\_BY\_16, 255, 1);

set\_pwm2\_duty(0);

output\_low(IN1);

output\_high(IN2);

while(TRUE)

{

if(kbhit()) // if data has been received

{

char i = getc(); // UART read

if (i == '\*') // special character for serial input. If the received character is \*, then this condition is called

{

counter = 0; // counter to be zero again

inputString = atol(strInput); // change string array to long variable

Kp = inputString/100.0f;

//printf("\fKp: %f",Kp); // print string array to the screen

printf(LCD\_PUTC,"\fKp=%f",Kp); // print inputString value to the LCD

memset(strInput, 0, 4); // clear the string array

}

else if(i== '#')

{

counter = 0; // counter to be zero again

inputString = atol(strInput); // change string array to long variable

Ki = inputString/100.0f;

//printf("\nKi: %f",Ki); // print string array to the screen

printf(LCD\_PUTC,"\nKi=%f",Ki); // print inputString value to the LCD

memset(strInput, 0, 4); // clear the string array

}

else

{

strInput[counter] = i; // attend input character to the string array

counter++; // increase the counter by 1

}

}

set\_adc\_channel(0); // next analog reading will be from channel 0

delay\_us(10);

pot = read\_adc();

delay\_us(10);

referance = pot \* 7.82f;

realSpeed = dx\_dt;

error = referance - realSpeed;

controlOut = Kp \* error + Ki \* integralError ;

if (controlOut >=1023){

controlOut = 1023;

}

else if (controlOut <=0){

controlOut = 0;

}

set\_pwm1\_duty(controlOut); //set pulse-width during which signal is high

/\*

printf(lcd\_putc, "\tref\_vel:%lu",referance);

printf(lcd\_putc, "\nact\_vel:%lu",realSpeed);

printf(lcd\_putc, "\terror:%lu",error);

printf(lcd\_putc, "\tsamp\_tm:%d",dt);

delay\_ms(50);

\*/

printf("/nKp: %f",Kp);

printf("\nKi: %f",Ki);

printf("\nReferance\_Velocity:%lu",referance);

printf("\nActual\_Velocity:%lu",realSpeed);

printf("\nError:%lu",error);

printf("\nSampling\_Time:%d",dt);

delay\_ms(1200);  
  
For the reciver code we firstly calculate the reference velocity. Than we try to code P cpntrooler with given information. And we add an I controller.For PI controller we use the Kp and KI values for the code and we try to define them with specific values for every each one.

For the second code transmitter.

#include <16F877A.h>

#FUSES XT, NOWDT, NOPROTECT, NOBROWNOUT, NOLVP, NOPUT, NODEBUG, NOCPD

#use delay(crystal=16000000)

#use rs232 (baud=9600,xmit=PIN\_C6, rcv=PIN\_C7, parity=N, stop=1) // configuretion for serial communication

void main()

{

setup\_psp(PSP\_DISABLED);

setup\_timer\_1(T1\_DISABLED);

setup\_timer\_2(T2\_DISABLED,0,1);

setup\_adc\_ports(NO\_ANALOGS);

setup\_adc(ADC\_OFF);

setup\_CCP1(CCP\_OFF);

setup\_CCP2(CCP\_OFF);

int16 dt = 1000;

while(1)

{

printf("5");

delay\_ms(dt);

printf("\*");

delay\_ms(dt);

printf("1");

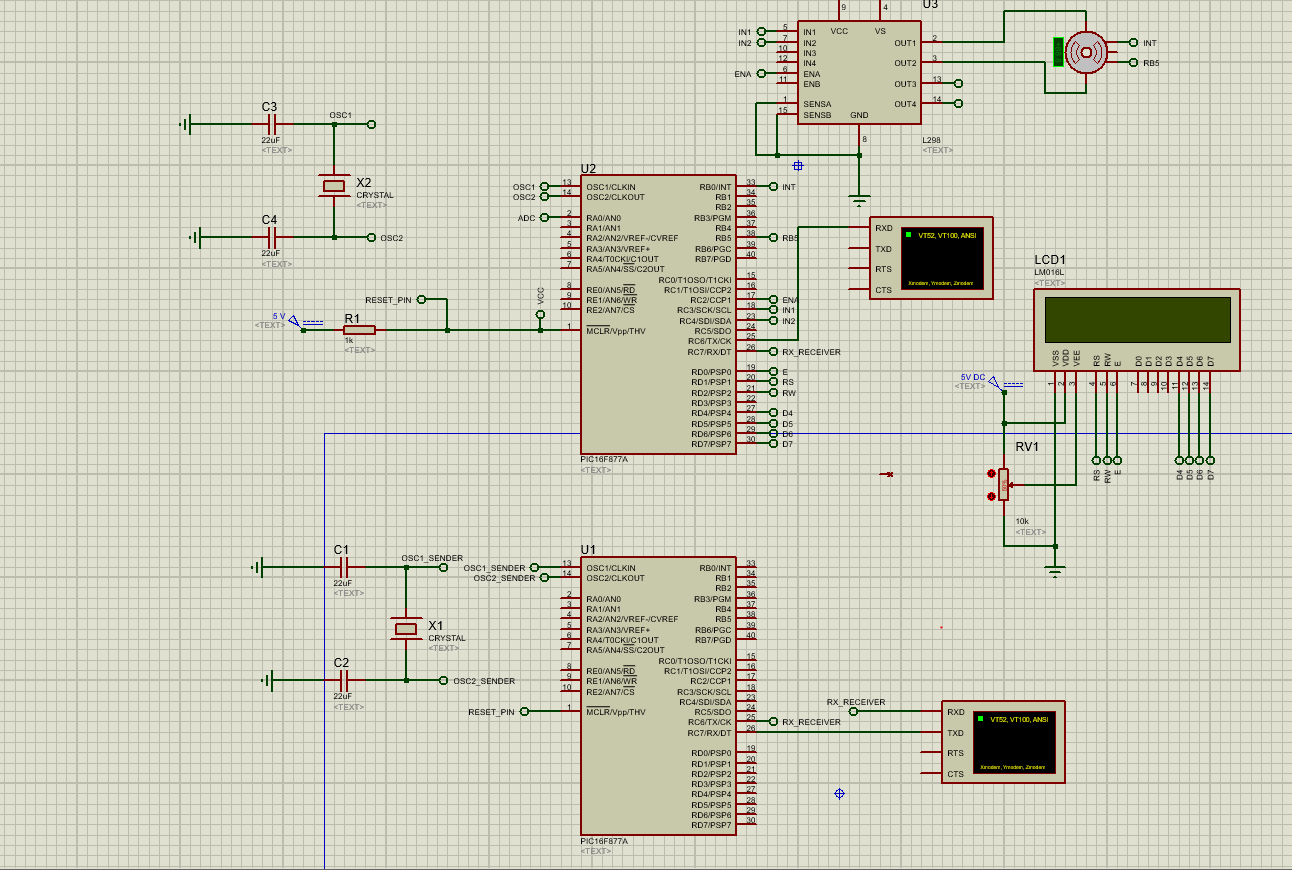
delay\_ms(dt);

printf("2");

delay\_ms(dt);

printf("#");

delay\_ms(dt); }

For the transmitter code we define for ardiunho uno atn we try the communicate the uno and microcontroller. We set the Kp and Kı values for communicate with each other and try to drive and encoder DC motor.

CONCULUTİON  
  
In this project, we learned how to design a PI controller and how to control the motor using Kp Ki values. During these operations, we also learned about communication and value control. While designing the controller, we analyzed, observed and learned how to write code.