UNIVERSITY OF WOLVERHAMPTON
KNOWLEDGE ▪ INNOVATION ▪ ENTERPRISE

Faculty of Science and Engineering

Assignment front cover sheet

SECTION A – TO BE COMPLETED BY THE STUDENT

Module Title	Embedded Systems	Module Code	5MA020
Assignment Title	DC Motor Control	Assignment Number	1
Student's Name	Nilangi Maheesha Sithumini Edirisinghe	Student's Number	1632456
Lecturer's Name	Dr. Buddhika Annasiwaththa		
I confirm that: • I have been given a copy of the assessment criteria relating to this assignment • I understand the meaning of the terms Cheating, Collusion and Plagiarism • All the work submitted is my own work			
Signature:		Date	

SECTION B – TO BE COMPLETED BY MODULE LEADER/TEAM

Assignment Feedback comments:

Signed:…….. Print Name:…….....

Date: …… Unconfirmed Mark ……..

Table of Contents

Question No.1

The physical parameters of the motor that I selected are;

$Motor\ Constant\ (K)$ $= 14 \times\ 10^{-3}\ Nm/A$

$Input\ Motor\ Circuit\ Resistance\ (R)$ $=\ 9.6\ \Omega$

$Input\ Motor\ Circuit\ Inductance\ (L)$ $= 440 \times 10^{-6}\ H$

$Motor\ and\ Load\ Moment\ of\ Inertia\ (J)$ $= 2.2 \times 10^{-7}\ kg\ m^2$

$Friction\ Coefficient\ (B)$ $= \dfrac{Friction\ Torque}{No\ Load\ Speed} = \dfrac{0.1 \times 10^{-3} Nm}{858.7\ rads^{-1}}$

$= \ \ 1.1646 \times 10^{-7}\ Nms$

$Gearbox\ Ratio\ (G)$ $= 1:34$

(DC-Micromotors)

(DC Micromotors Series 2233 Datasheet)

Question No.2

In order to obtain the transfer function of the system it was broken into its theoretical parts and then the transfer function for each part was derived.

For the first part, the motor was considered as a voltage to torque device in spite of its inertia.

$$G_{motor} = \frac{T_L(s)}{V_a(s)}$$

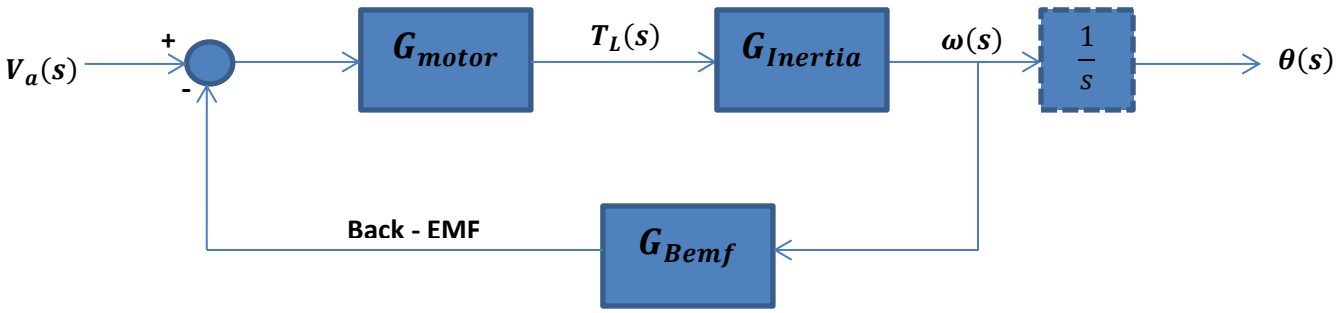
For the next part, all the inertia was gathered into an inertia block. It represented the relationship between torque and angular velocity.

$$G_{inertia} = \frac{\omega(s)}{T_L(s)}$$

As the last part, the back-emf generated by the armature coils was considered.

$$G_{Bemf} = \frac{V_{Bemf}}{\omega(s)}$$

Due to the back-emf, we have a negative feedback system acting within the motor. Hence, the block diagram can be represented as follows:

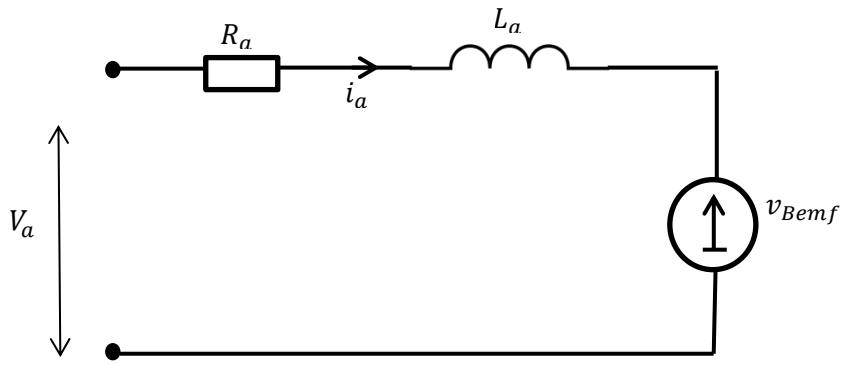


Then, in order to obtain the transfer function for the entire system, the above mentioned transfer functions were analyzed further more.

Analyzing $G_{motor} = \frac{T_L(s)}{V_a(s)}$

$$T_L(s) = \quad K_m \; I_a(s)$$

In order to derive an expression for $I_a(s)$, electrical analysis of the system was considered.



The current in the winding could be found from Kirchchoff's Voltage Law,

$$v_a(t) = i_a \, R_a + \; L_a \frac{di_a}{dt} + v_{Bemf}$$

Nevertheless, as the back emf is considered separately, here it was omitted.

$$v_a(t) = i_a \, R_a + \; L_a \frac{di_a}{dt}$$

Then, the Laplace Transform was taken.

$$V_a(s) = I_a(s) \;\; R_a + L_a \, s \, I_a(s)$$

(All initial conditions were assumed to be zero as the ultimate result is a transfer function)

$$V_a(s) = I_a(s)(R_a + L_a s)$$

$$I_a(s) = \frac{V_a(s)}{R_a + L_a s}$$

By substituting the expression for $I_a(s)$ into the $T_L(s)$ equation,

$$T_L(s) = K_m \frac{V_a(s)}{R_a + L_a s}$$

Finally, it can be arranged as,

$$G_{motor} = \frac{T_L(s)}{V_a(s)} = \frac{K_m}{R_a + \; L_a s}$$

Analyzing $G_{inertia} = \frac{\omega(s)}{T_L(s)}$

The torque produced by the motor is used to rotate the total mass of the system and to counterbalance its viscous friction. Hence, the equation is;

$$\tau_M(t) = \tau_J(t) + \tau_B(t)$$

$\tau_M(t)$ = the torque generated by the motor
$\tau_J(t)$ = the torque necessary to rotate the mass (the armature and the load) of the system
$\tau_B(t)$ = the torque required to overcome the viscous friction of the bearings

This can be rewritten in terms of $T_L(t)$;

$$T_L(t) = \tau_J(t) + \tau_B(t)$$

And each of those torques can be written in terms of ω as follows:

$$\tau_B(t) = B\omega$$

$$\tau_J(t) = J\frac{d\omega}{dt}$$

Thus;

$$T_L(t) = J\frac{d\omega}{dt} + B\omega$$

Then the Laplace Transform can be taken;

$$T_L(s) = Js\omega(s) + B\omega(s)$$

Hence;

$$G_{inertia} = \frac{\omega(s)}{T_L(s)} = \frac{1}{Js + B}$$

Analyzing $G_{Bemf} = \frac{V_{Bemf}}{\omega(s)}$

The back emf is proportional to the angular velocity of the shaft by a constant factor K_B. Therefore;

$$V_{Bemf}(t) = K_B\omega(t)$$

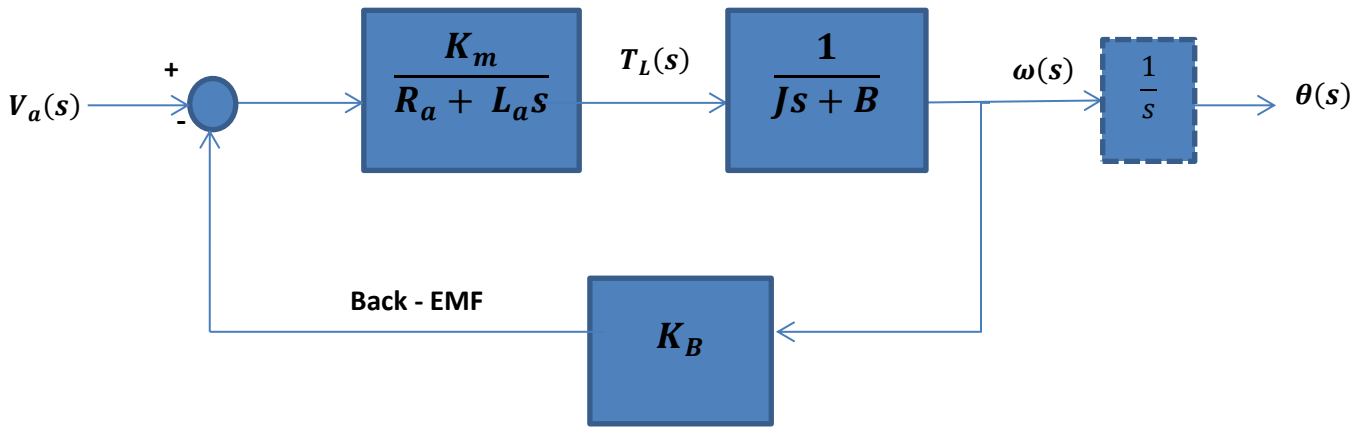
Then the Laplace Transform was taken;

$$V_{Bemf}(s) = K_B\omega(s)$$

Thus;

$$G_{Bemf} = \frac{V_{Bemf}}{\omega(s)} = K_B$$

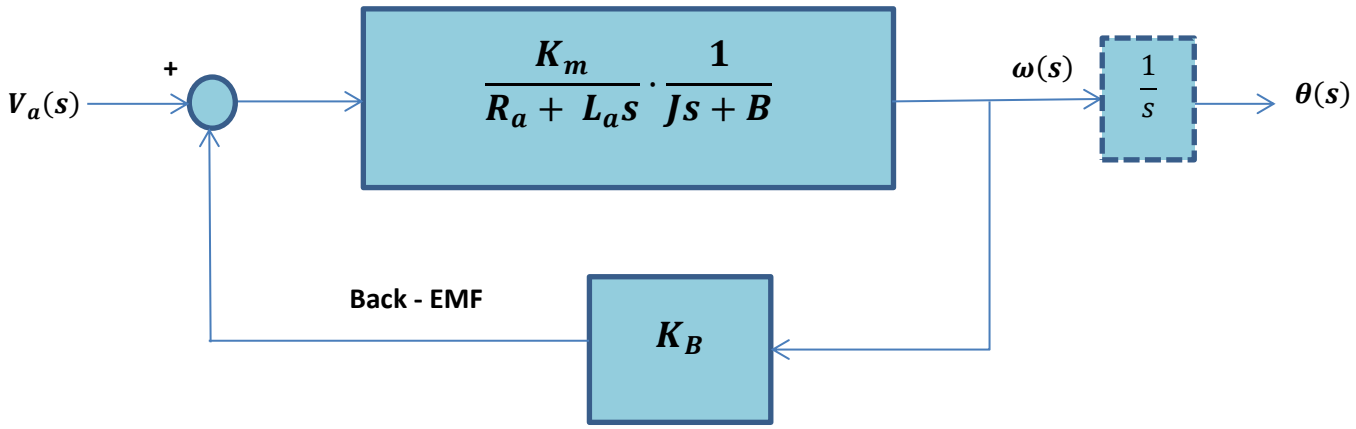
Consequently, the block diagram was obtained as follows:



Simplification of the Block Diagram

The combined transfer function of series blocks is taken by multiplying each transfer function together. Since the combined transfer function was taken as:

$$G_1 = G_{motor} \cdot G_{inertia} = \frac{K_m}{R_a + L_a s} \cdot \frac{1}{Js + B}$$



$$G_{dc-motor} = \frac{G_1}{1 + G_1 K_B}$$

According to the above equation;

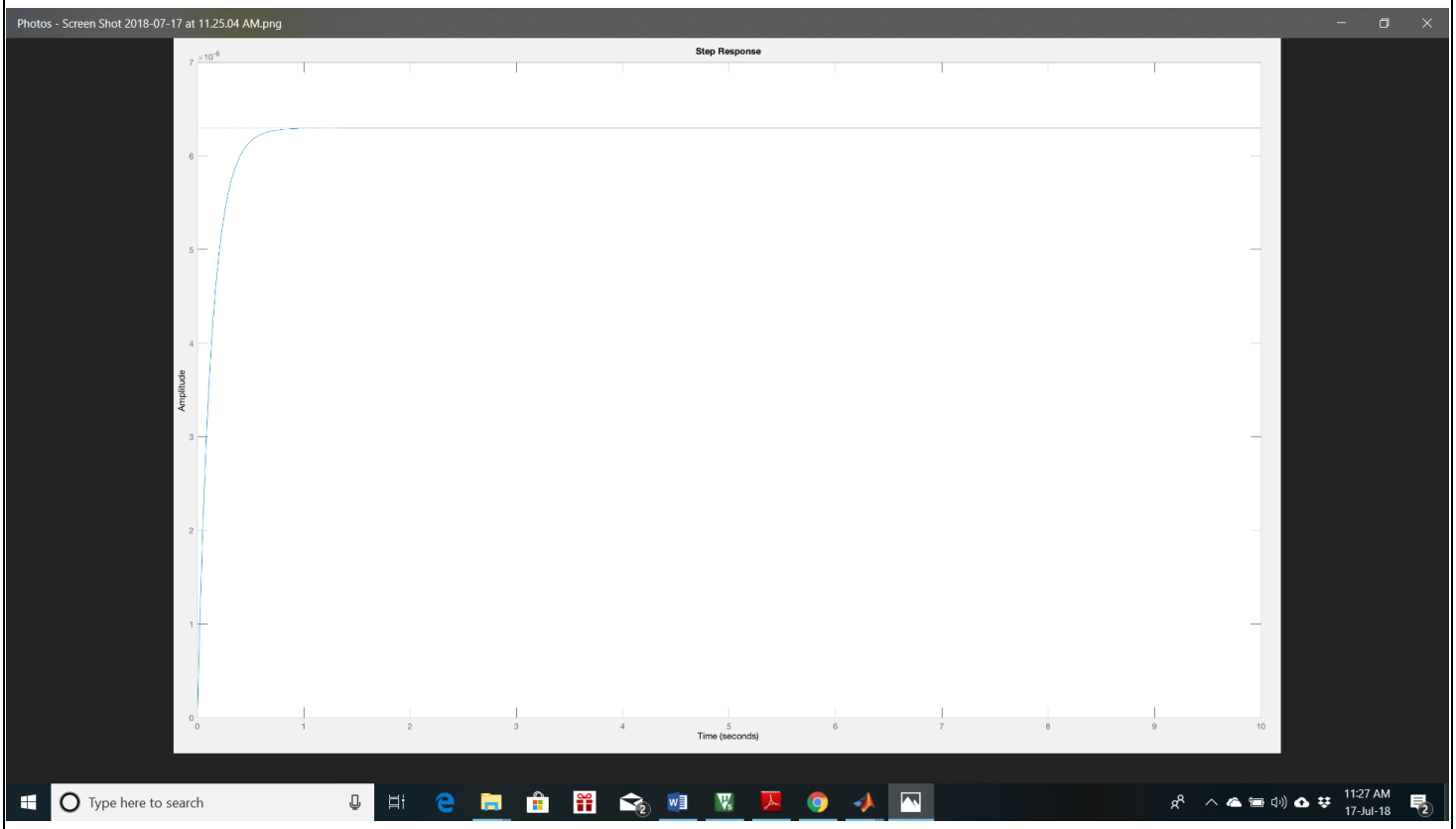
$$G_{dc-motor} = \frac{\frac{K_m}{(R_a + L_a s)(J_s + B)}}{1 + \frac{K_m K_B}{(R_a + L_a s)(J_s + B)}}$$

$$G_{dc-motor} = \frac{K_m}{(R_a + L_a s)(J_s + B) + K_m K_B}$$

$$Y_s = \frac{K_m}{(R_a + L_a s)(J_s + B) + K_m K_B} \times \frac{1}{s}$$

$$y(t) = L^{-1}\left[\frac{K_m}{(R_a + L_a s)(J_s + B) + K_m K_B} \times \frac{1}{s}\right]$$

The above step response was derived using inverse Laplace Transform and then the appropriate values were substituted and the step response was obtained as follows.

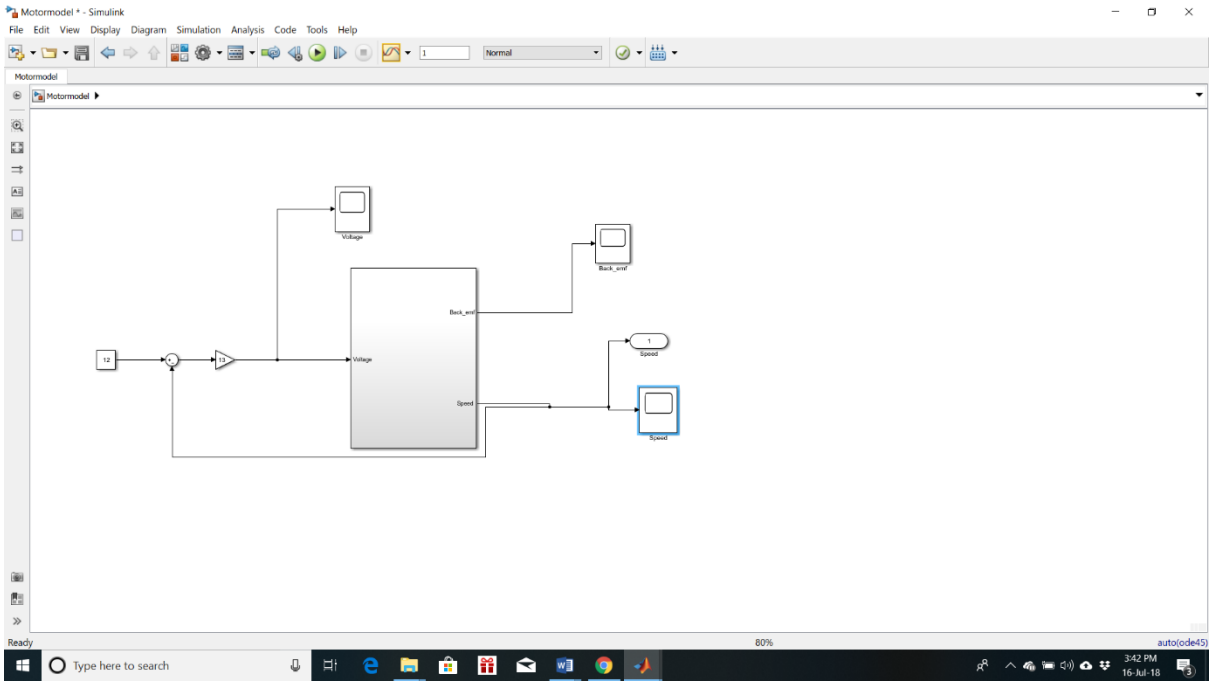


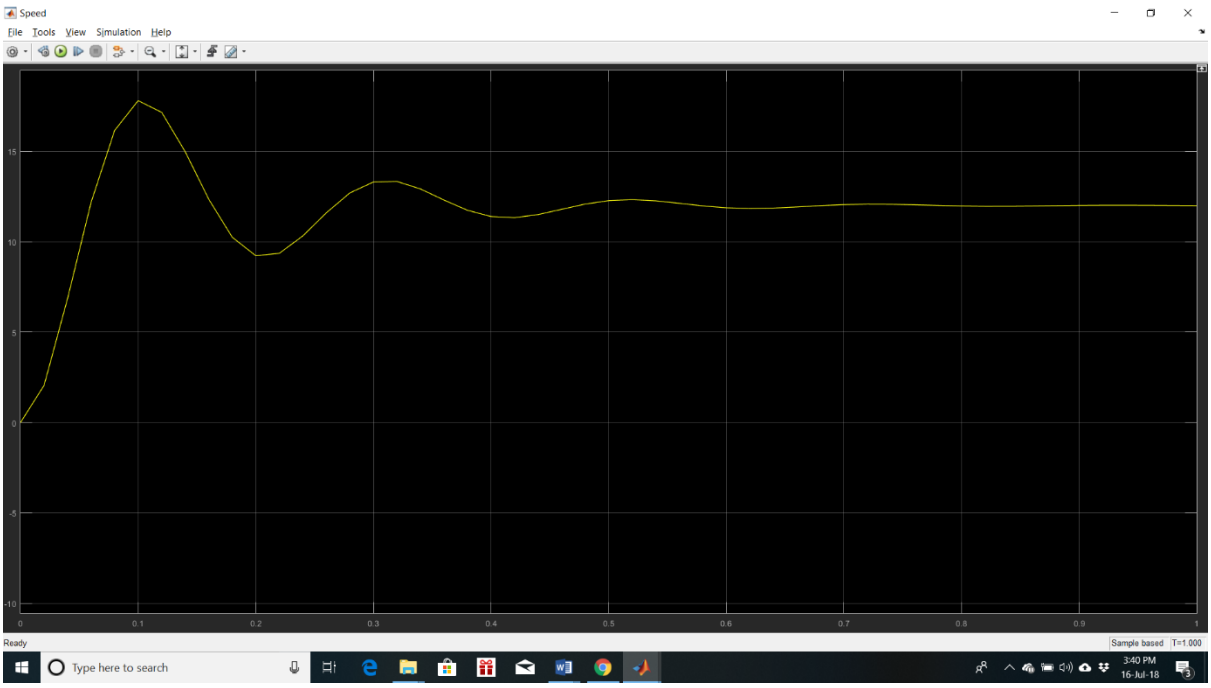
(Sillitoe, 2017)

Question No.3

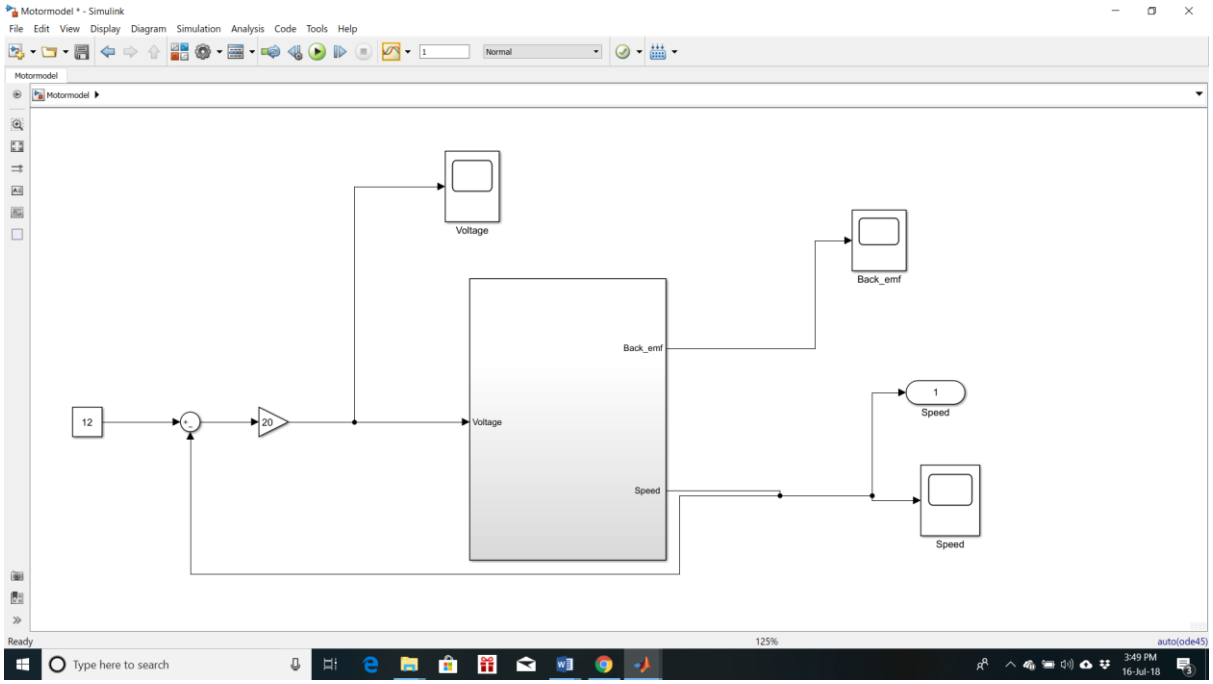
Block Diagram Modified with Different Amplifier Gains

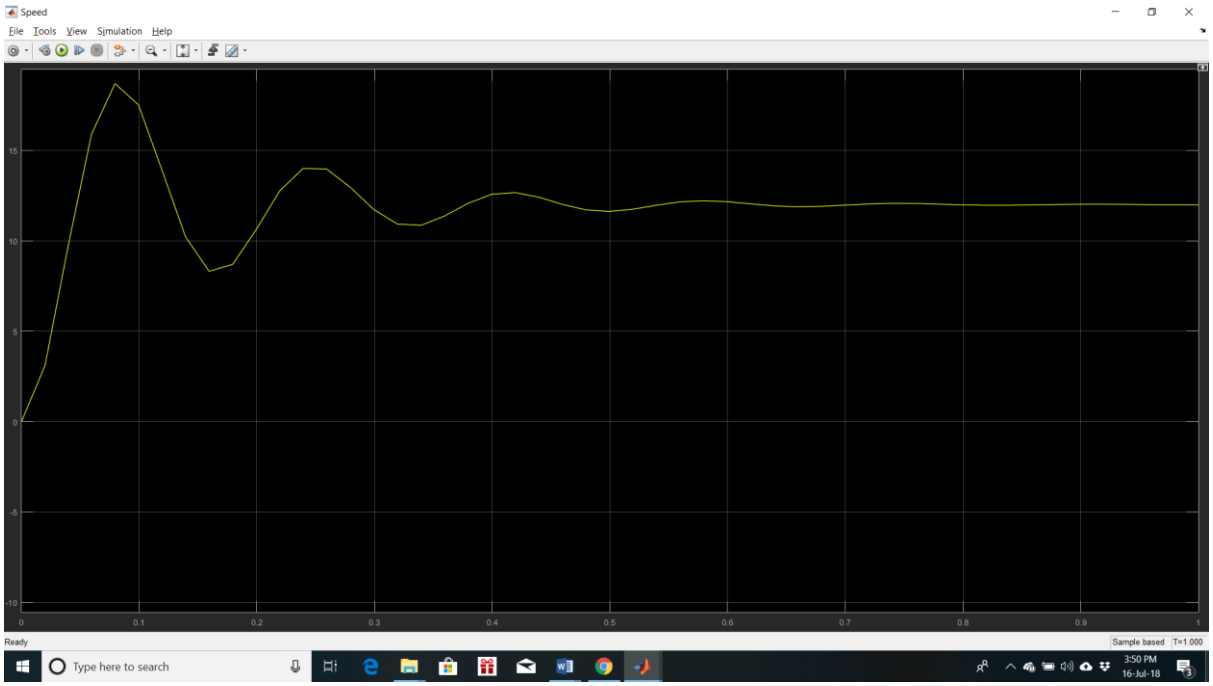
Gain = 13



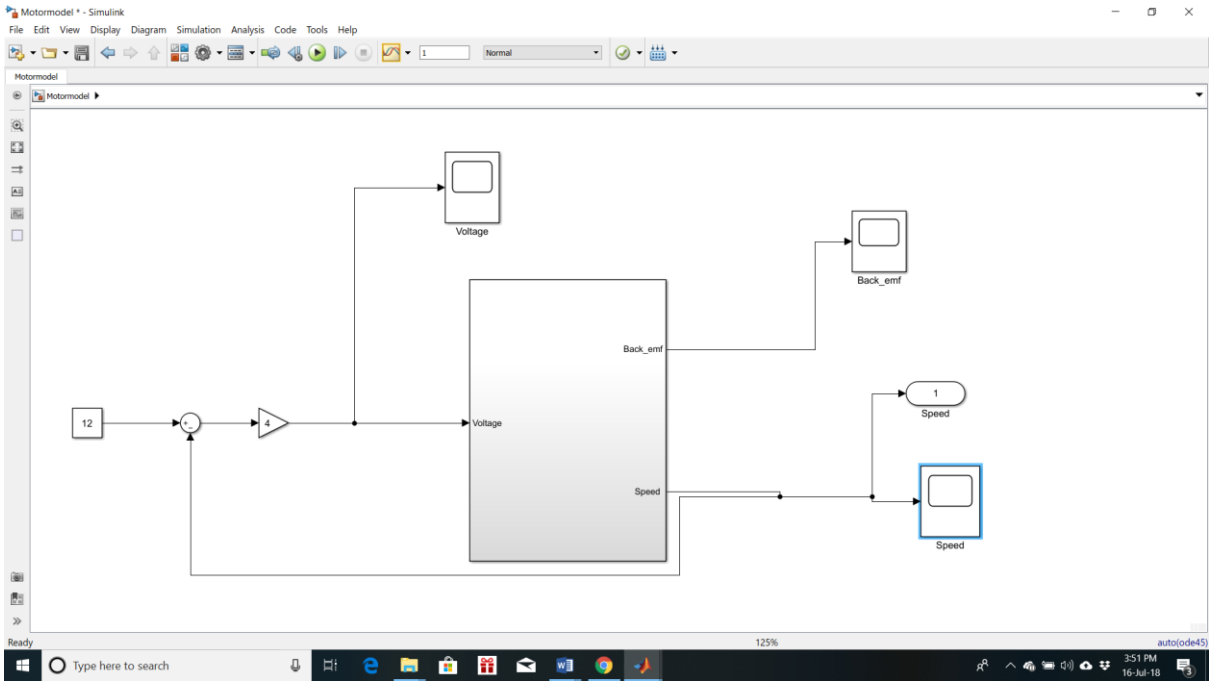


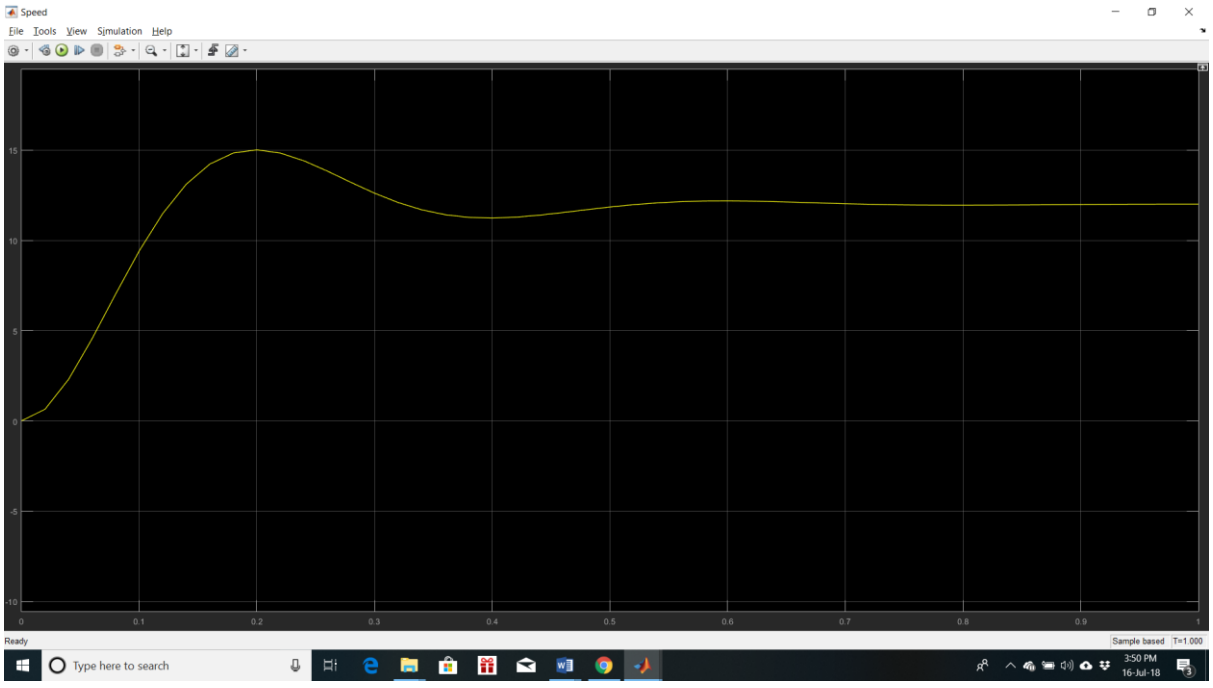
Gain=20


Motormodel * - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
Motormodel
Voltage
Back_emf
12
20
Voltage
Back_emf
Speed
1
Speed
Speed
Ready
125%
auto(ode45)
Type here to search
3:49 PM
16-Jul-18

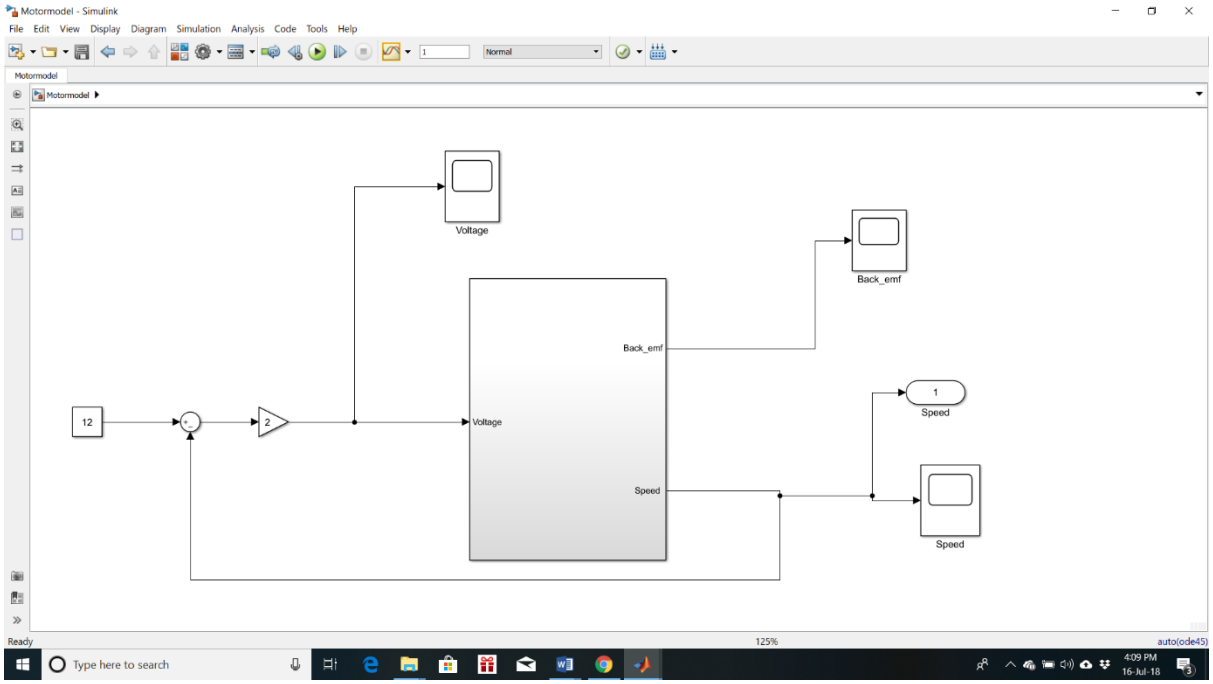

Speed
File Tools View Simulation Help
15
10
5
0
-5
-10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Ready
Sample based T=1.000
Type here to search
3:50 PM
16-Jul-18

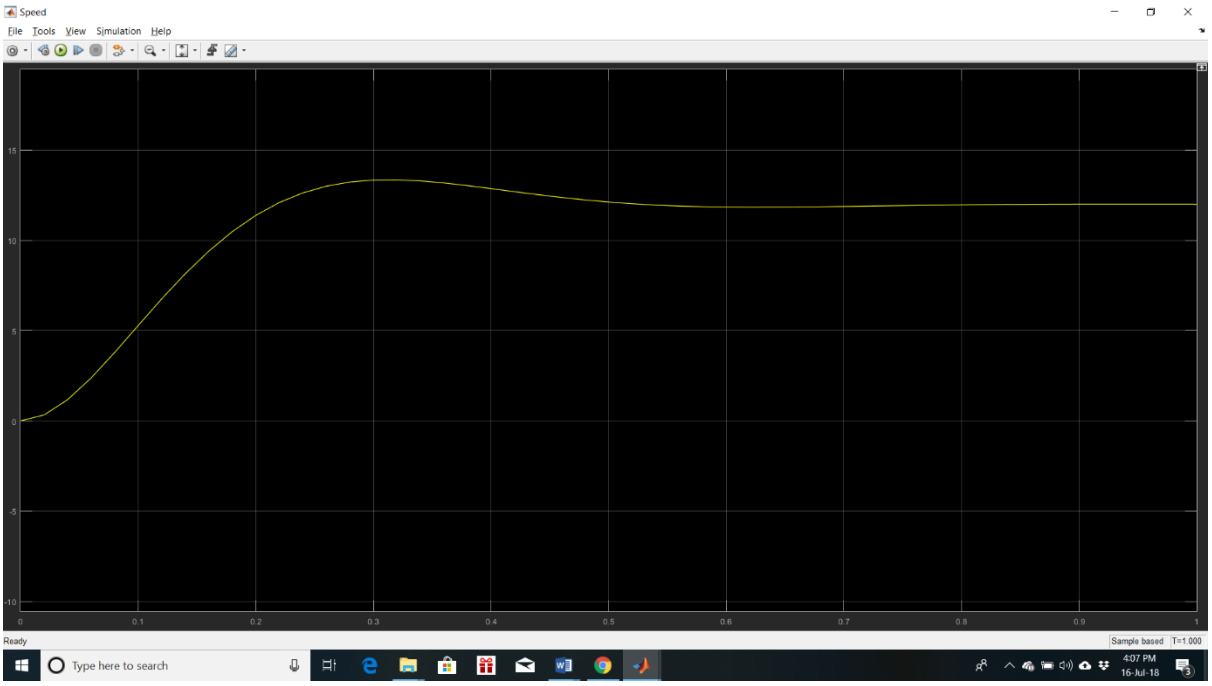
Gain = 4


Motormodel * - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
Motormodel
Voltage
Back_emf
12
4
Voltage
Back_emf
Speed
1
Speed
Speed
Ready
125%
auto(ode45)
Type here to search
3:51 PM
16-Jul-18


Speed
File Tools View Simulation Help
15
10
5
0
-5
-10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Ready
Sample based T=1.000
Type here to search
3:50 PM
16-Jul-18

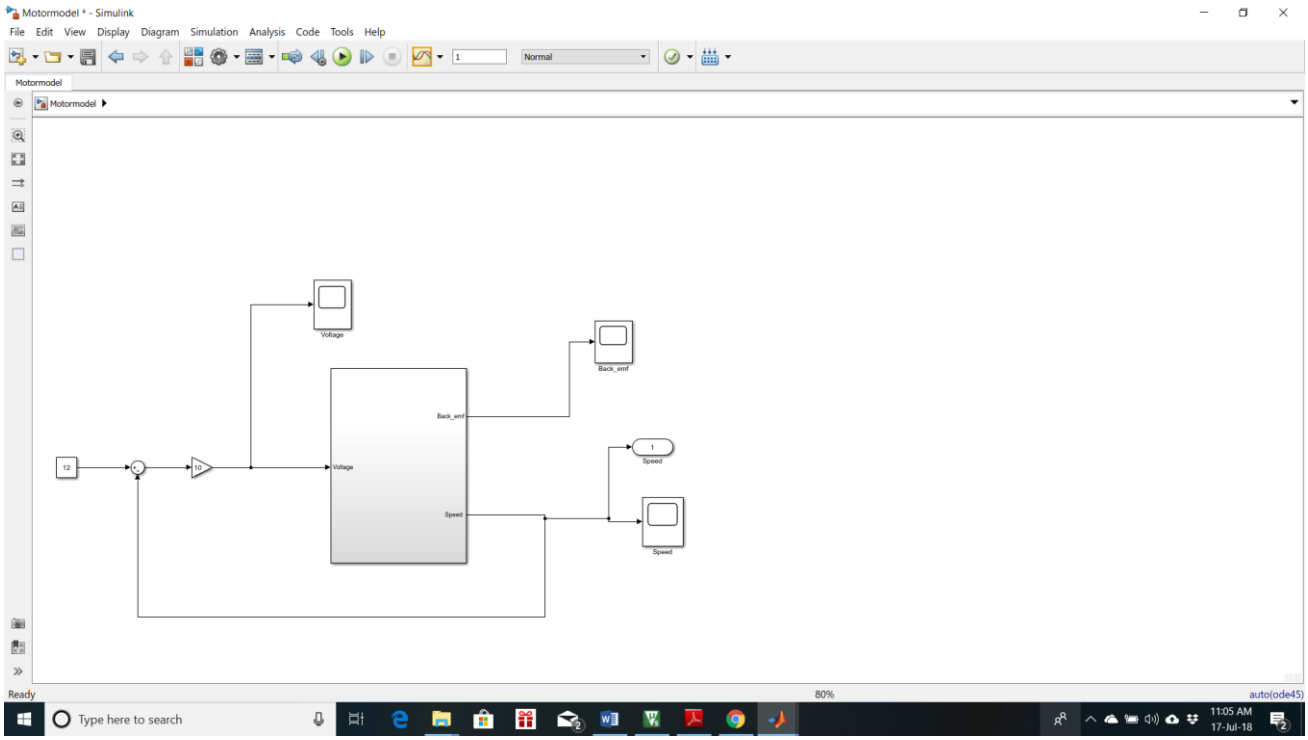
Gain = 2

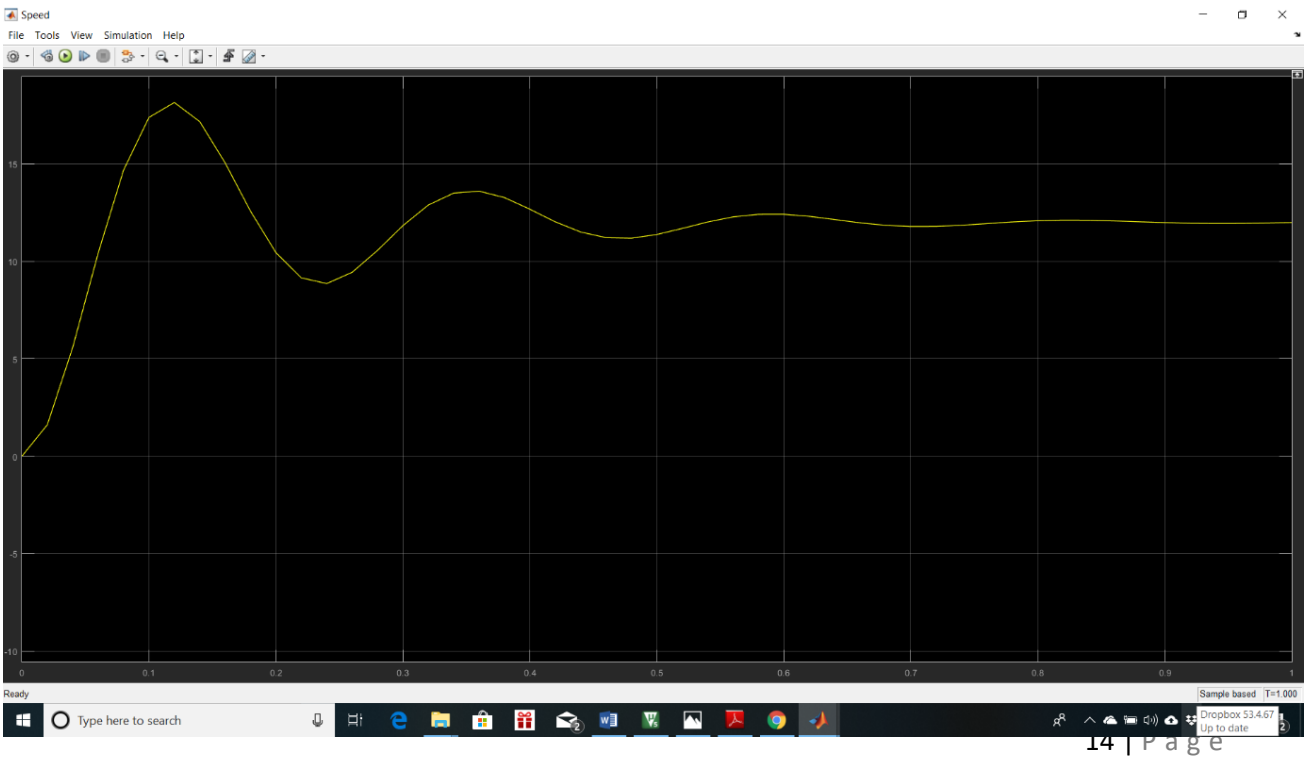

Motormodel - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
Motormodel
Voltage
Back_emf
Back_emf
Voltage
Speed
1
Speed
Speed
12
2
Ready
125%
auto(ode45)
Type here to search
4:09 PM
16-Jul-18


Speed
File Tools View Simulation Help
15
10
5
0
-5
-10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Ready
Sample based
T=1.000
Type here to search
4:07 PM
16-Jul-18

Question 4

In Order for the system to exhibit an adequate, speedy and precise response, after applying many gains, through trial and error it was found out the suitable underdamped response can be obtained when the gain is equal to 10. It can be displayed as follows:





Question No.5 – Report on Motor Control

1) Objective

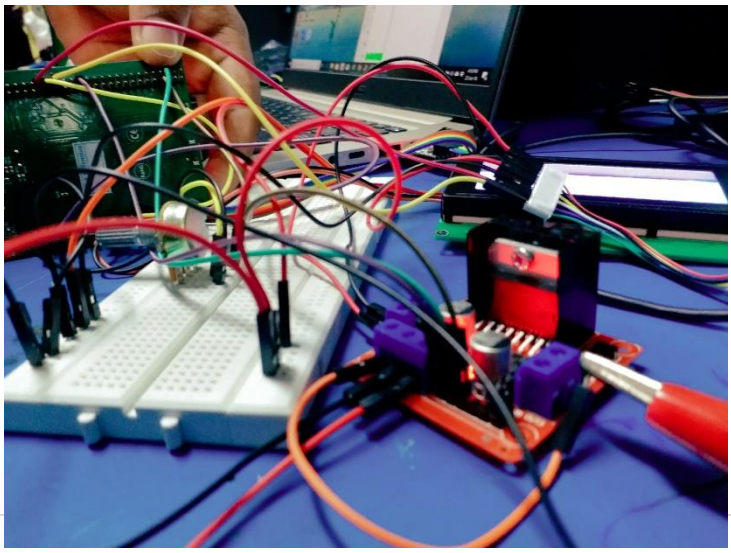
The main aim of this report is to depict the theories, apparatus, procedure, observations, calculations and results obtained by implementing dc motor speed control. The goal of the experiment was to maintain a desired motor speed with minimum error.

2) Introduction

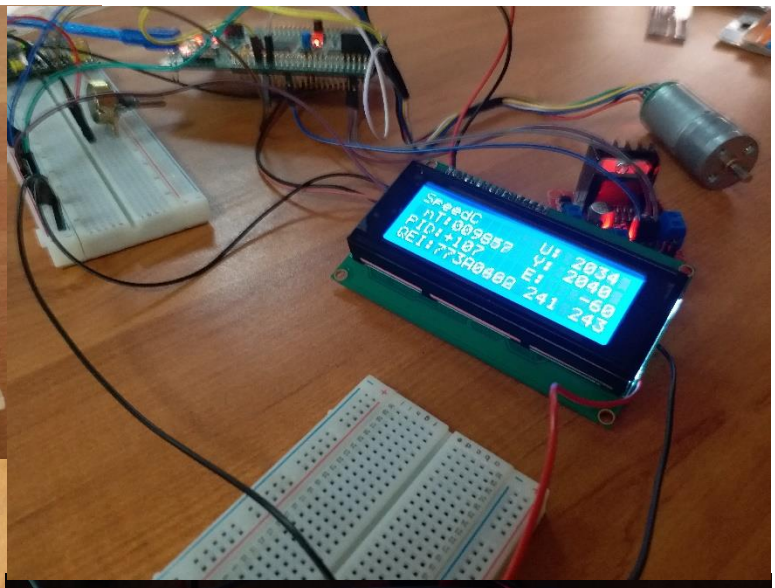
This experimentation was carried out with the objective of maintaining a desired motor speed by utilizing PID control as the controller and pulse width modulation in the Keil uVision code. The necessary coding was done to the STM32F407 board using Keil uVision 5. The PID controller was implemented using MATLAB and then converted to a C code and added to the IDE.

3) Apparatus

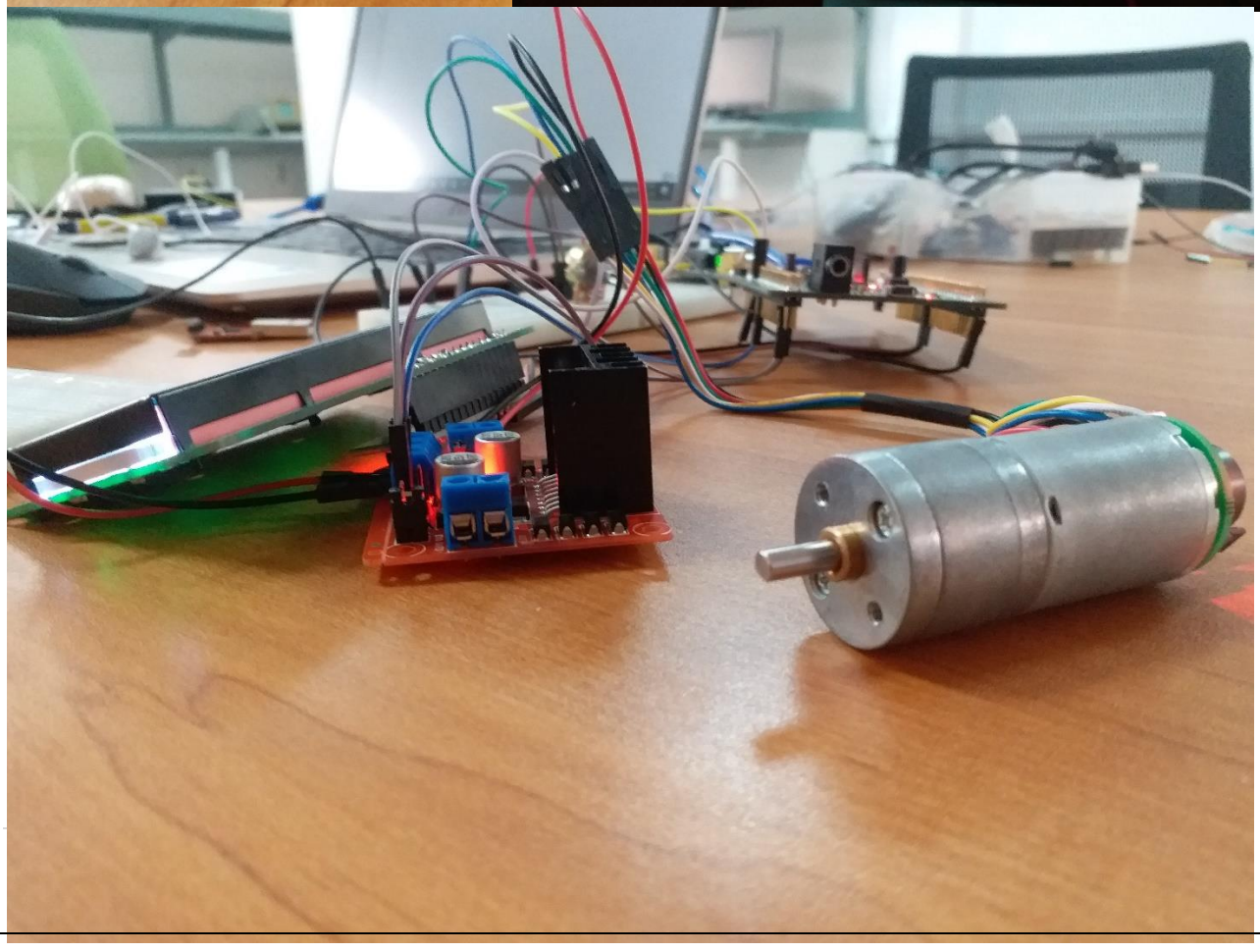
1. STM32F407 Discovery Board
2. DC Motor with Gear box and Encoder
3. L298 Motor Driver
4. Bread Board
5. Bread Board Wires
6. LCD Display
7. I2C Converter
8. USB to TTL Converter Module
9. DC Power Supply
10. Bread Board Power Supply




JGA25测速电机
光电测速
霍尔测速
YFROBOT


SpeedC
nT:009857
PID:+107
QEI:7739A0BE
U: 2034
Y: 2040
E: -60
241 243


SpeedC
nT:009361
PID:+108
QEI:7739A37A
U: 2034
Y: 2040
E: -60
241 243



4) Methodology

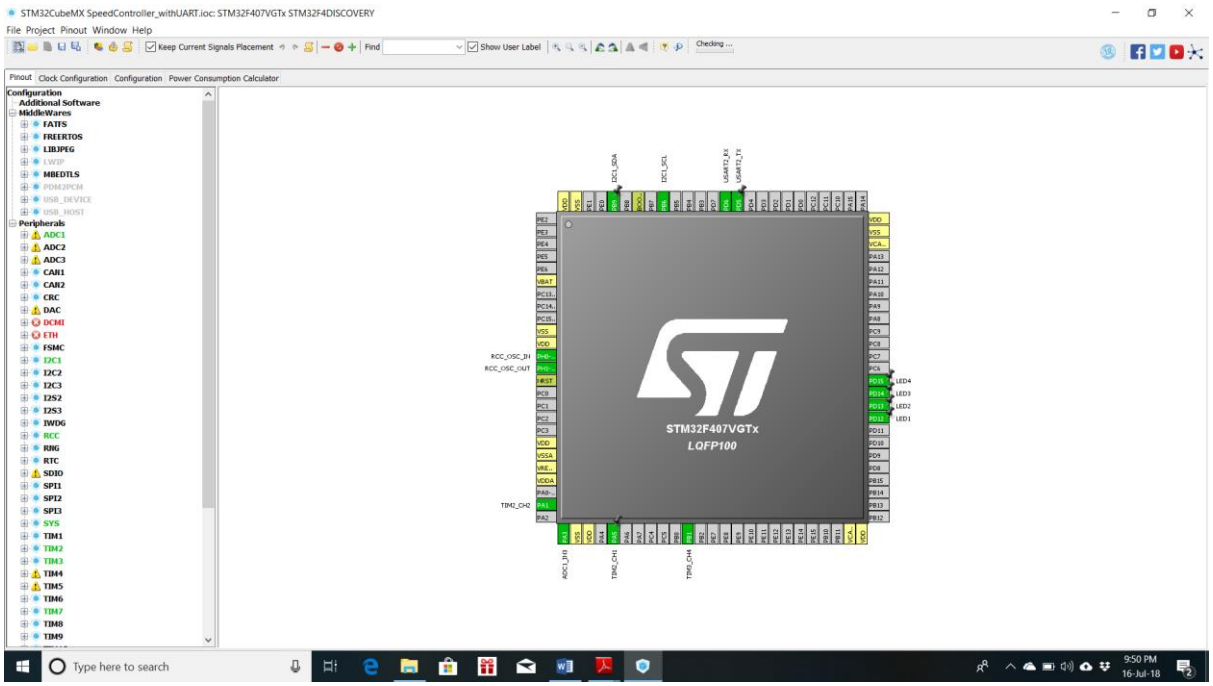
In real time, the procedure was a lengthy process obtained through a long period of time.

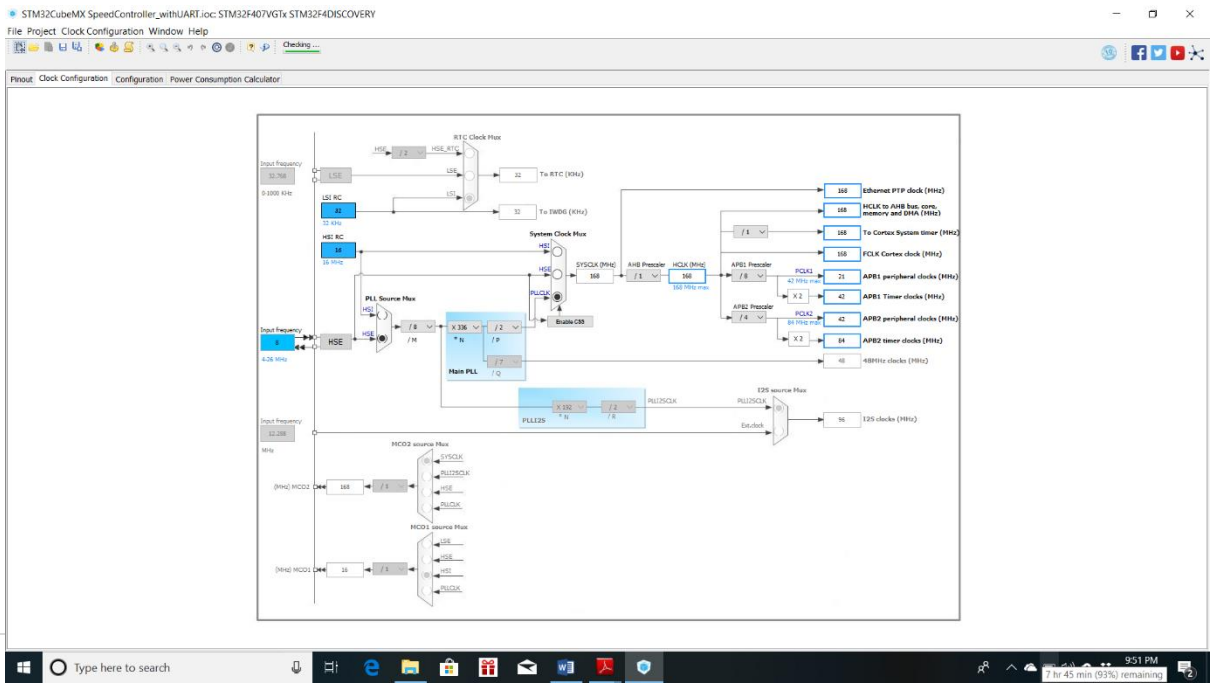
Step 1

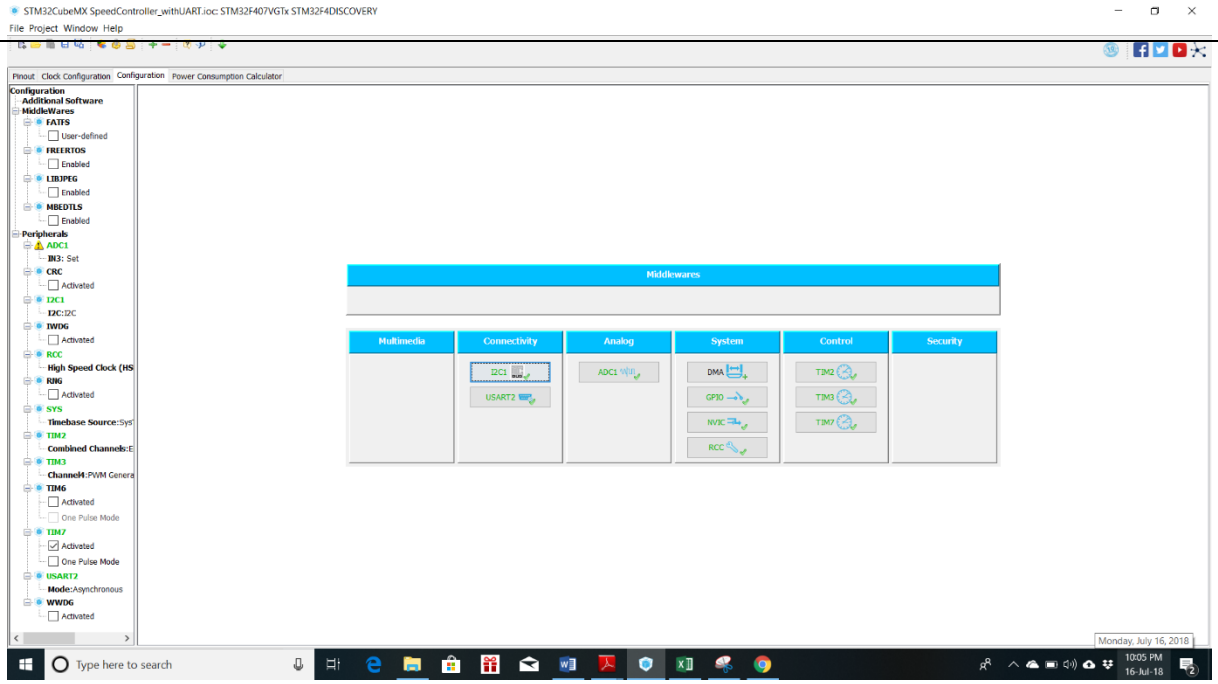
A suitable DC motor was selected and bought along with the other necessary equipment.

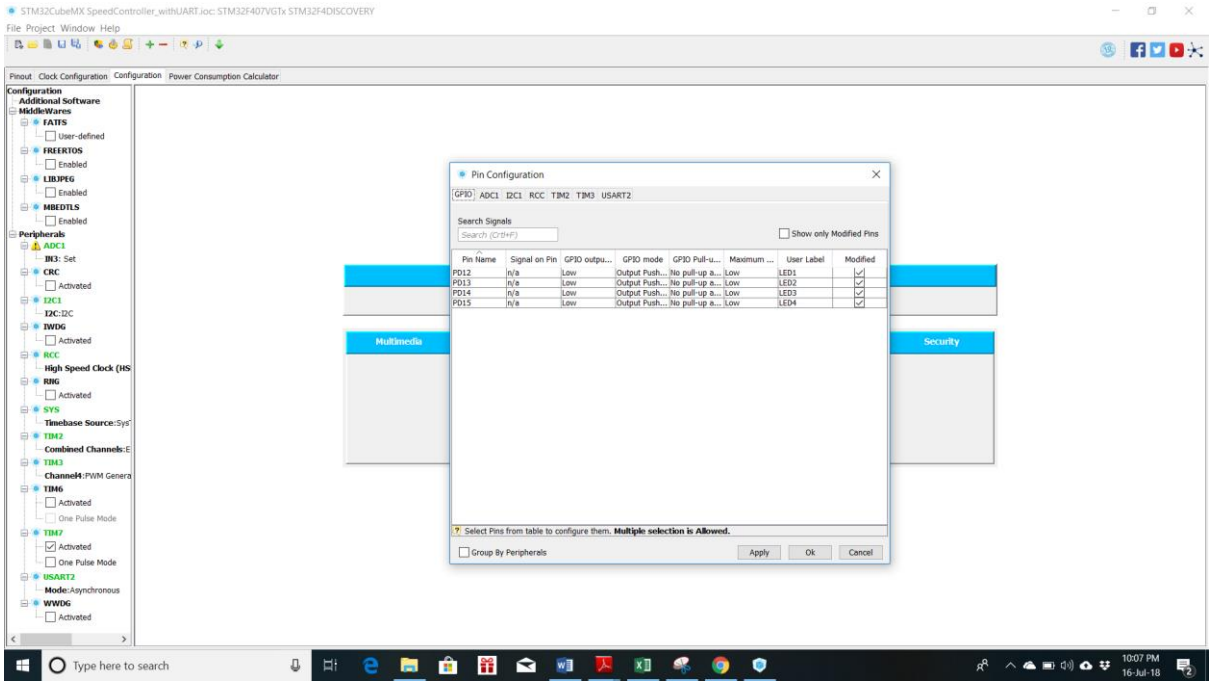
Step 2

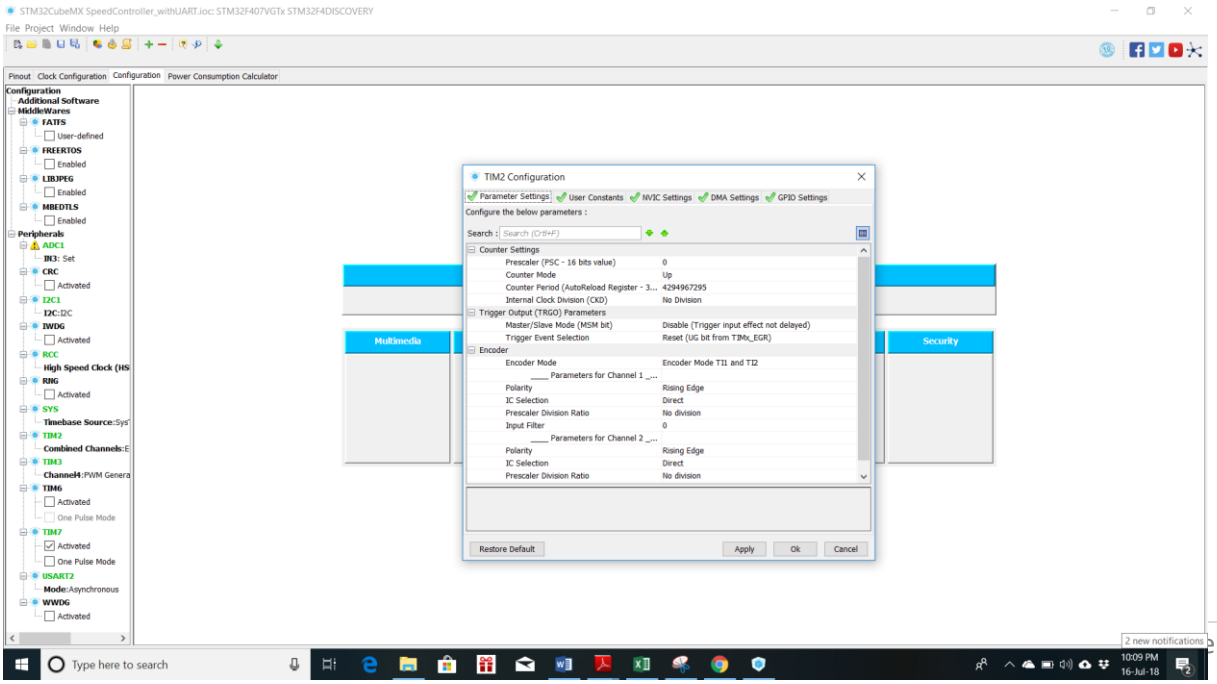
The code which was needed to run the STM32F407 discover board was generated via Cube MX.






STM32CubeMX SpeedController_withUART.ioc: STM32F407VGTx STM32F4DISCOVERY
File Project Window Help
Pinout Clock Configuration Configuration Power Consumption Calculator
Middlewares
Multimedia
Connectivity
Analog
System
Control
Security
I2C1
USART2
ADC1
DMA
GPIO
NVIC
RCC
TIM2
TIM3
TIM7
Monday, July 16, 2018
Type here to search
10:05 PM
16-Jul-18


STM32CubeMX SpeedController_withUART.ioc: STM32F407VGTx STM32F4DISCOVERY
File Project Window Help
Pinout Clock Configuration Configuration Power Consumption Calculator
Pin Configuration
GPIO ADC1 I2C1 RCC TIM2 TIM3 USART2
Search Signals
Search (Crtl+F)
Show only Modified Pins
Pin Name Signal on Pin GPIO outpu... GPIO mode GPIO Pull-u... Maximum ... User Label Modified
PD12 n/a Low Output Push... No pull-up a... Low LED1
PD13 n/a Low Output Push... No pull-up a... Low LED2
PD14 n/a Low Output Push... No pull-up a... Low LED3
PD15 n/a Low Output Push... No pull-up a... Low LED4
Select Pins from table to configure them. Multiple selection is Allowed.
Group By Peripherals
Apply
Ok
Cancel
Multimedia
Security
Type here to search
10:07 PM
16-Jul-18


STM32CubeMX SpeedController_withUART.ioc: STM32F407VGTx STM32F4DISCOVERY
File Project Window Help
Pinout Clock Configuration Configuration Power Consumption Calculator
TIM2 Configuration
Parameter Settings
User Constants
NVIC Settings
DMA Settings
GPIO Settings
Configure the below parameters :
Search :
Counter Settings
Prescaler (PSC - 16 bits value) 0
Counter Mode Up
Counter Period (AutoReload Register - 3... 4294967295
Internal Clock Division (CKD) No Division
Trigger Output (TRGO) Parameters
Master/Slave Mode (MSM bit) Disable (Trigger input effect not delayed)
Trigger Event Selection Reset (UG bit from TIMx_EGR)
Encoder
Encoder Mode Encoder Mode TI1 and TI2
Parameters for Channel 1 ...
Polarity Rising Edge
IC Selection Direct
Prescaler Division Ratio No division
Input Filter 0
Parameters for Channel 2 ...
Polarity Rising Edge
IC Selection Direct
Prescaler Division Ratio No division
Restore Default
Apply
Ok
Cancel
Multimedia
Security
2 new notifications
Type here to search
10:09 PM
16-Jul-18

As the above screen capture depicts, the timers were allocated accordingly and the settings were setup as per the table below.

TIM3, TIM4	General purpose	16bit	16bit	4	SysClk/4	SysClk, SysClk/2	1
TIM9	General purpose	16bit	16bit	2	SysClk/2	SysClk	2
TIM10, TIM11	General purpose	16bit	16bit	1	SysClk/2	SysClk	2
TIM12	General purpose	16bit	16bit	2	SysClk/4	SysClk, SysClk/2	1
TIM13, TIM14	General purpose	16bit	16bit	1	SysClk/4	SysClk, SysClk/2	1
TIM6, TIM7	Basic	16bit	16bit	0	SysClk/4	SysClk, SysClk/2	1

Steps of Generating the code via Cube MX

a) Firstly, TIMER 2 was allocated and setup in order to measure the current motor speed from the Quadrature Encoder. In other words, as the Quadrature Encoder Interface (QEI). It was connected to pins PA1 and PA5. The part of code which initiated TIMER 2 was as follows:

```
/* TIM2 init function */
static void QEI_MX_TIM2_Init(void)
{

  TIM_Encoder_InitTypeDef sConfig;
  TIM_MasterConfigTypeDef sMasterConfig;

  htim2.Instance = TIM2;
  htim2.Init.Prescaler = 0;
  ht
  ht
  ht
  s(
  s(
  s(
  s(
  s(
  s(
  s(
  s(
  ii
  {
  }
  sI
  sI
  ii
  {
  }
  }

    TIM2->EGR = 1;           // Generate an update event
    TIM2->CR1 = 1;           // Enable the counter

}
```

```
/* TIM7 init function */
static void ControlTick_MX_TIM7_Init(void)
{

  TIM_MasterConfigTypeDef sMasterConfig;

  htim7.Instance = TIM7;
  htim7.Init.Prescaler = 209;
  htim7.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim7.Init.Period = 4999;
  if (HAL_TIM_Base_Init(&htim7) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim7, &sMasterConfig) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

  HAL_TIM_Base_Start_IT(&htim7);
```

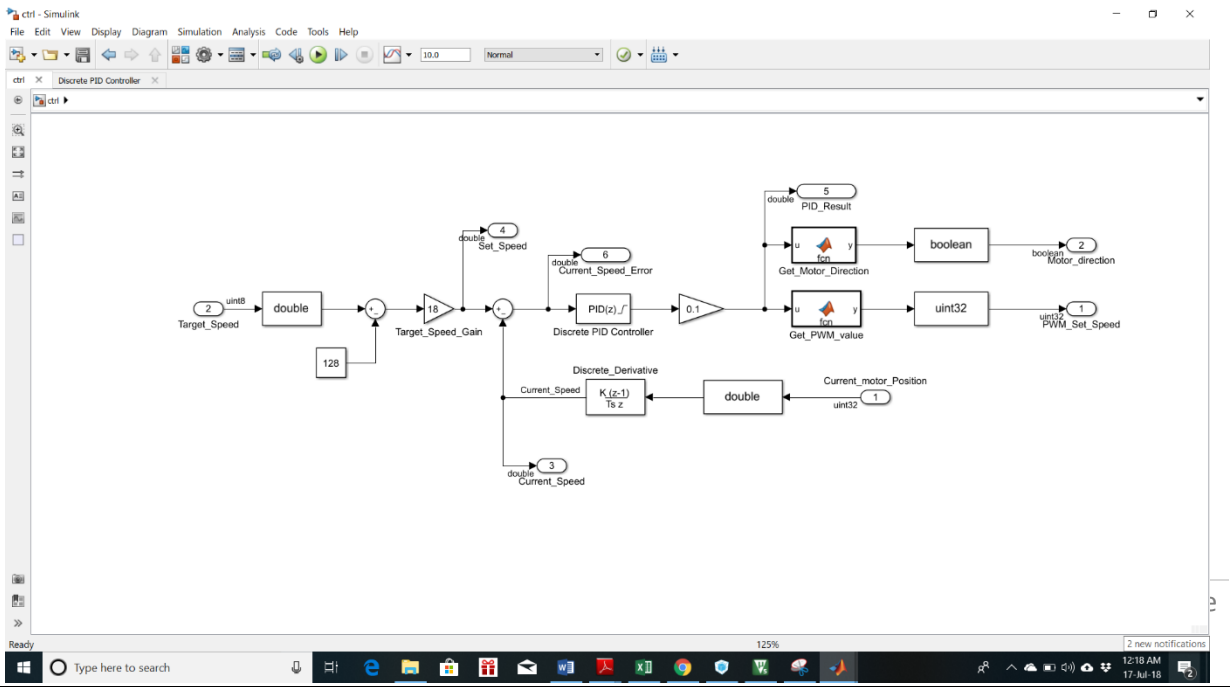
d) After that, for Pulse Width Modulation, TIMER 3 was allocated. The pin PB1 was used for PWM generation. Two GPIO pins were assigned to decide direction of motor.

```
/* TIM3 init function */
static void PWM_MX_TIM3_Init(void)
{

  TIM_MasterConfigTypeDef sMasterConfig;
  TIM_OC_InitTypeDef sConfigOC;

  htim3.Instance = TIM3;
  htim3.Init.Prescaler = 21;
  htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim3.Init.Period = 200;
  htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  if (HAL_TIM_PWM_Init(&htim3) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim3, &sMasterConfig) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

  sConfigOC.OCMode = TIM_OCMODE_PWM1;
  sConfigOC.Pulse = 0;
  sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
  sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
  if (HAL_TIM_PWM_ConfigChannel(&htim3, &sConfigOC, TIM_CHANNEL_4) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

  HAL_TIM_MspPostInit(&htim3);

}
```

e) Next, the LCD was setup through the I2C converter. The I2C SCL pin was connected to pin PB6 and I2C SDA pin was connected to pin PB9.

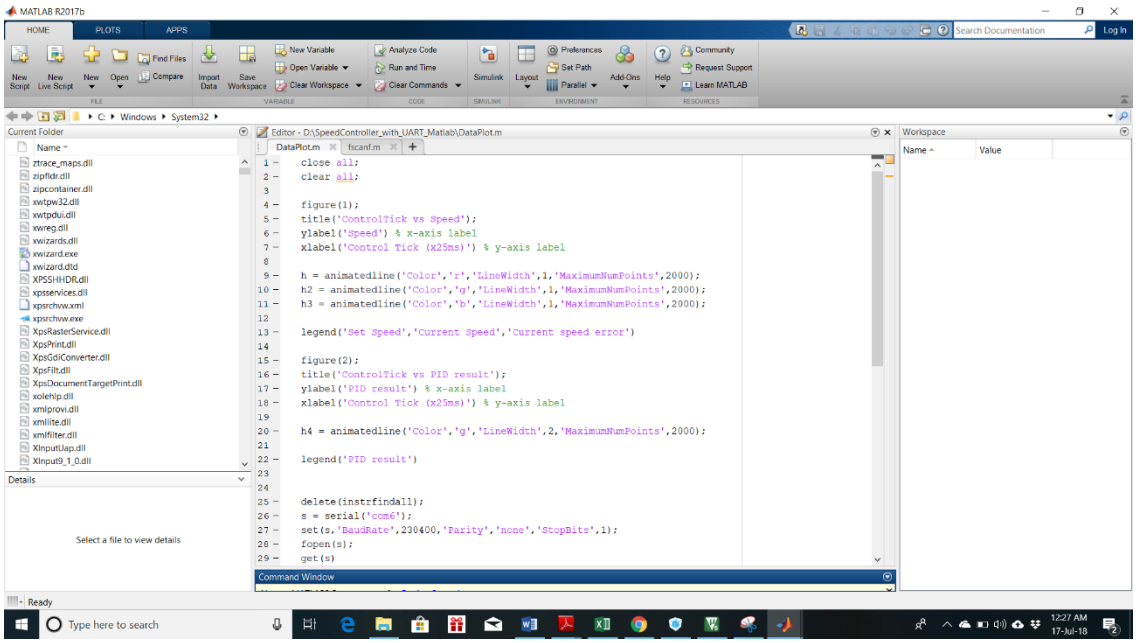
Step 3

The control code was generated by developing a model in Simulink and generating a C file from it. That controller code was then added to the rest of the code in Keil.



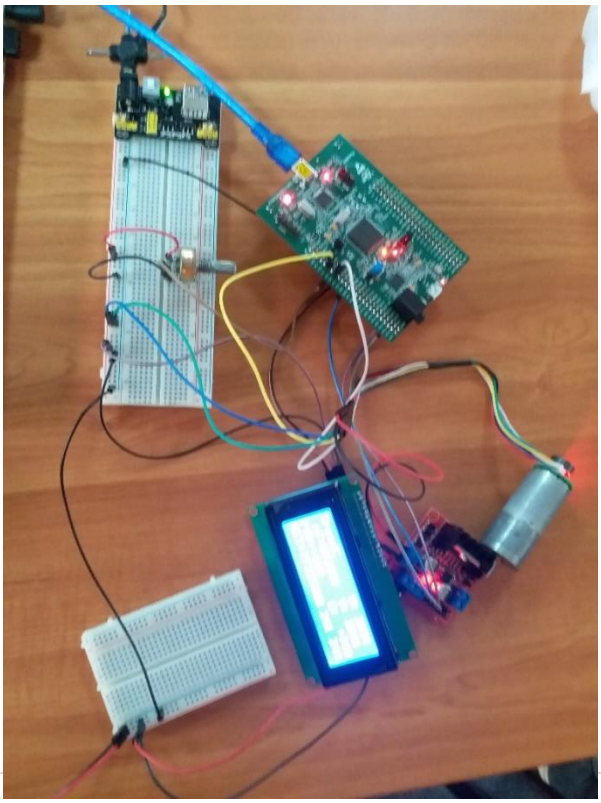
Step 4

The *DataPlot.m* file was created in MATLAB in order to get results as graphs.



Step 5

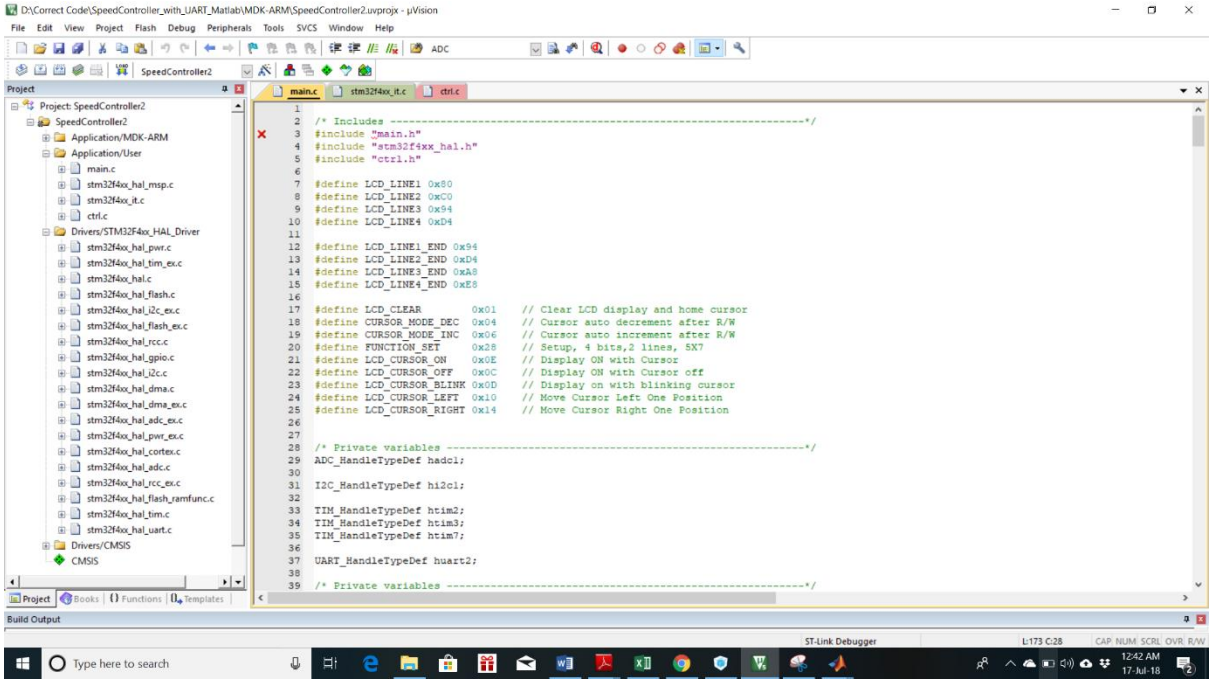
The I2C was soldered onto the LCD and the motor, STM32 board, the USB to TTL converter etc. were wired and the experiment was conducted. The results were obtained.

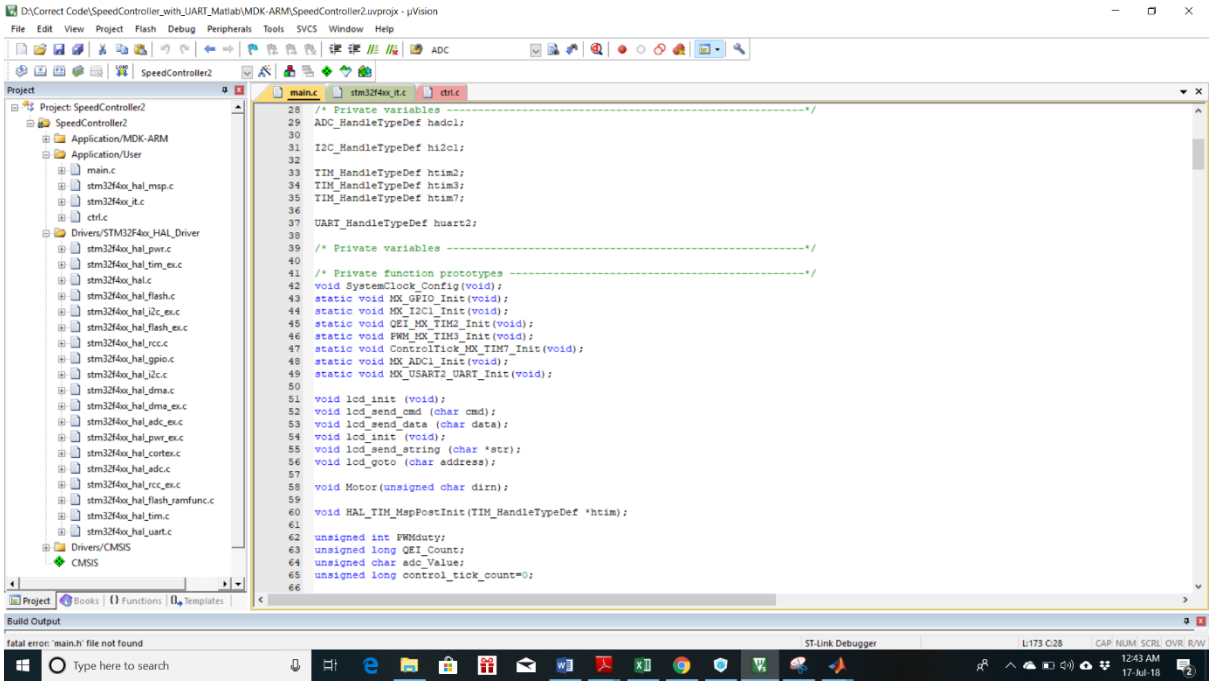


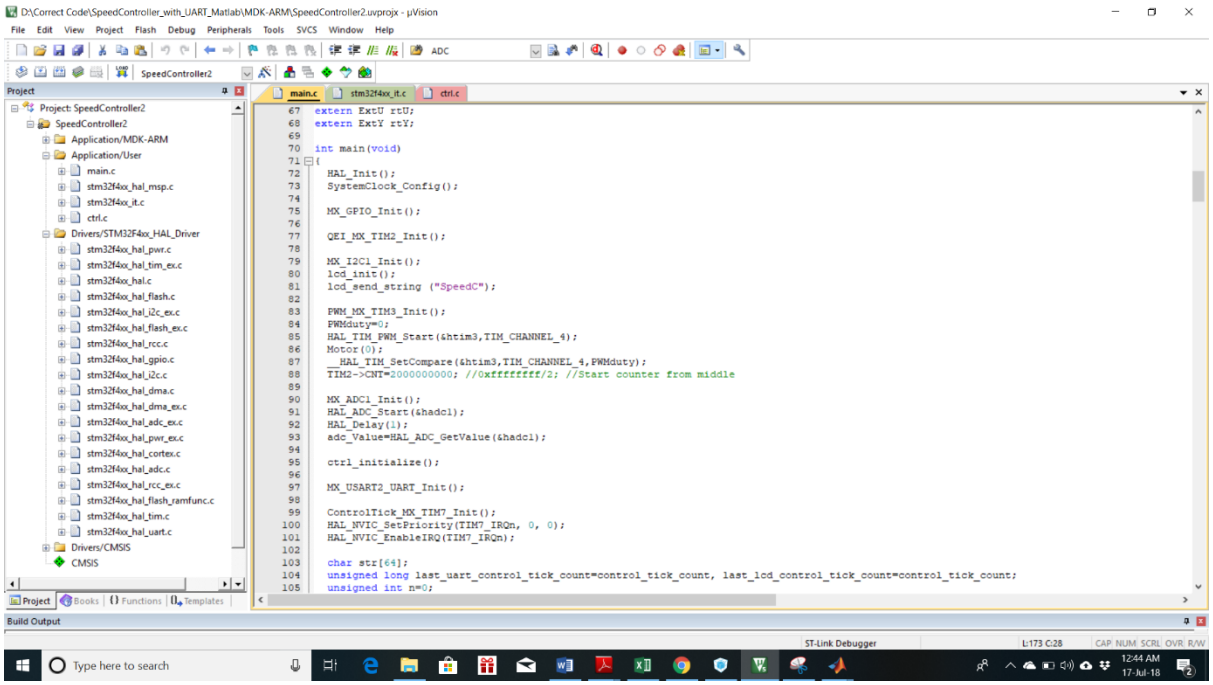
5) Coding

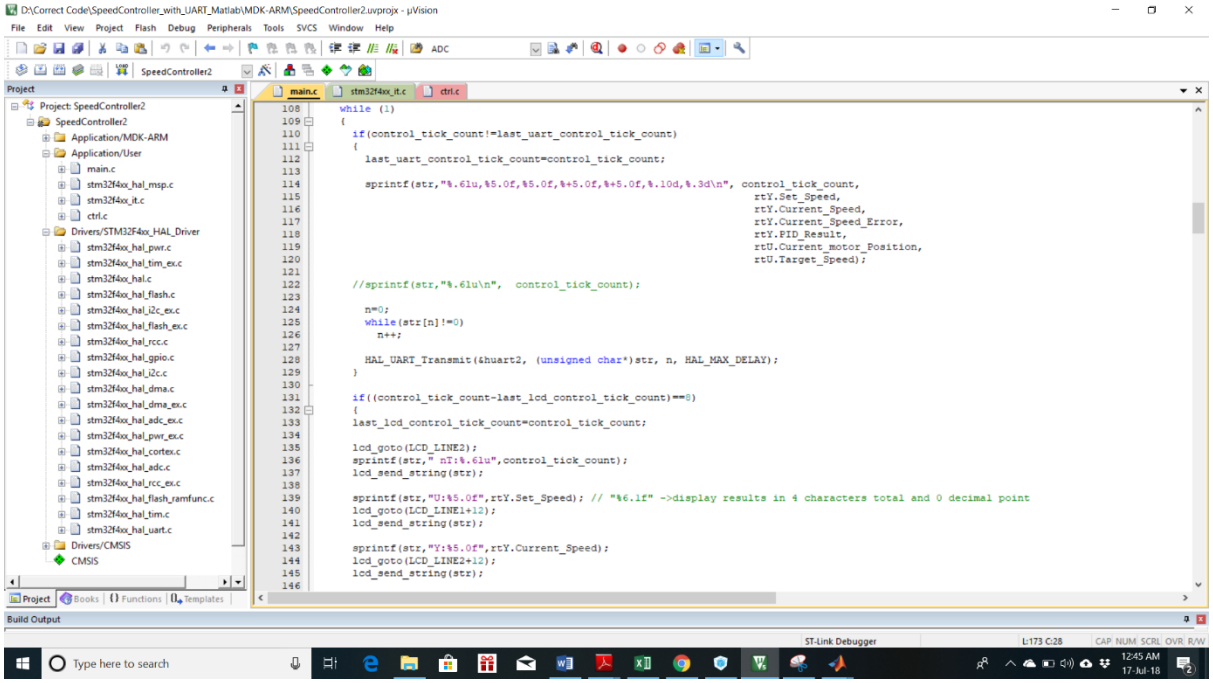
The main.c file

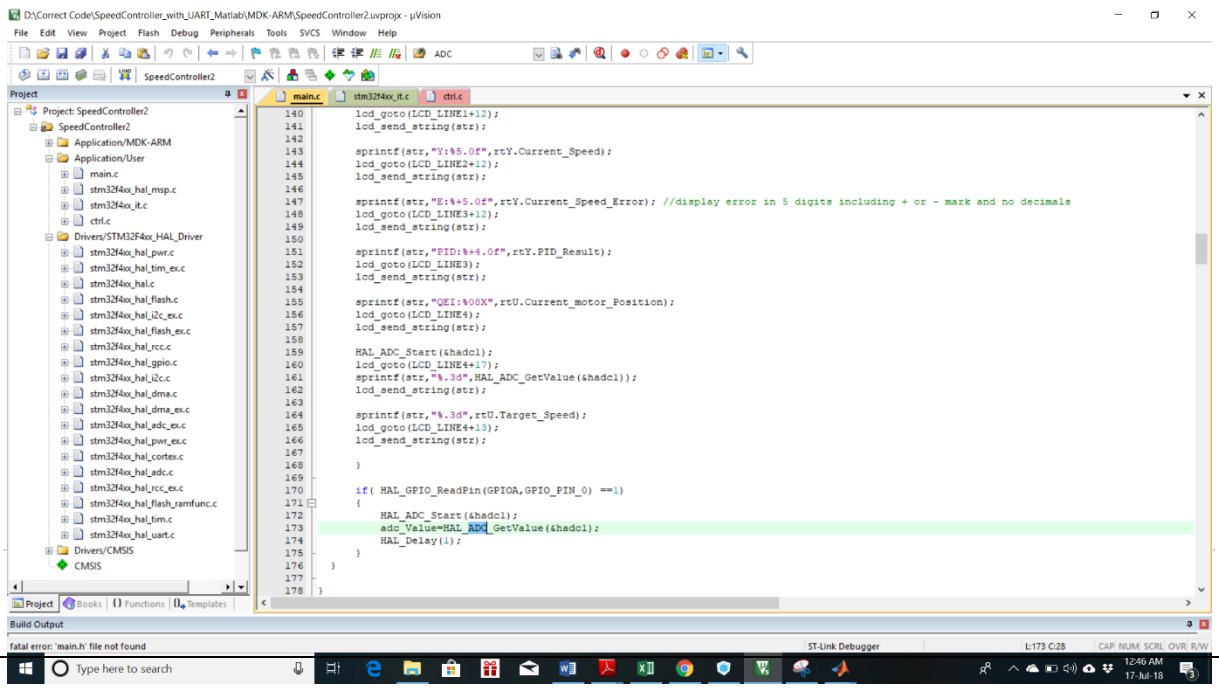
The main code (main.c file) that was used to run and control the speed of motor was as follows.

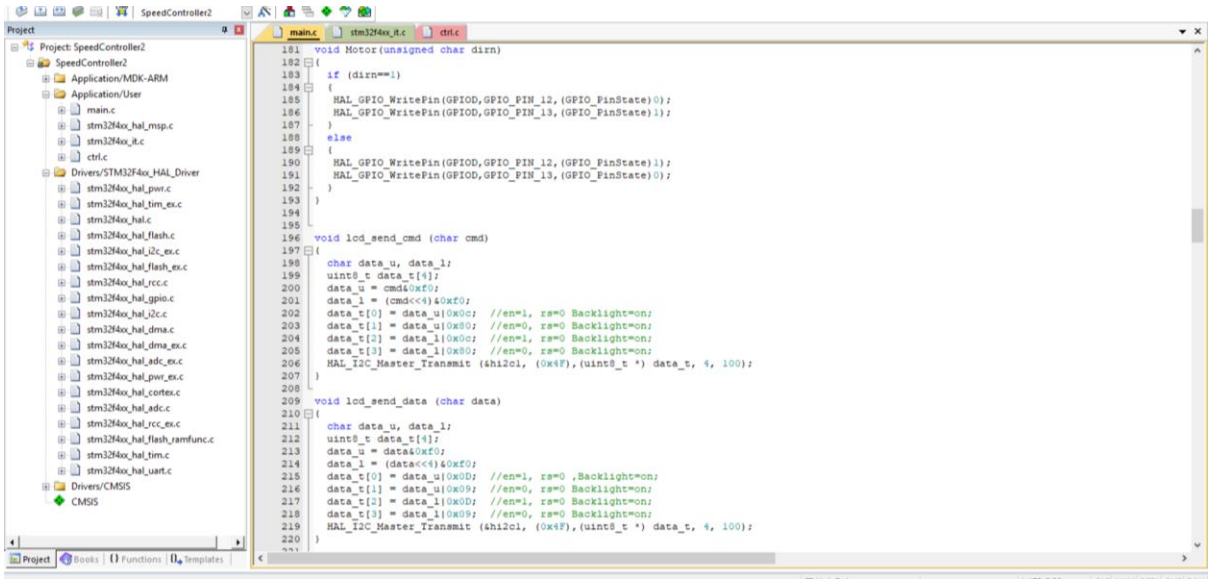




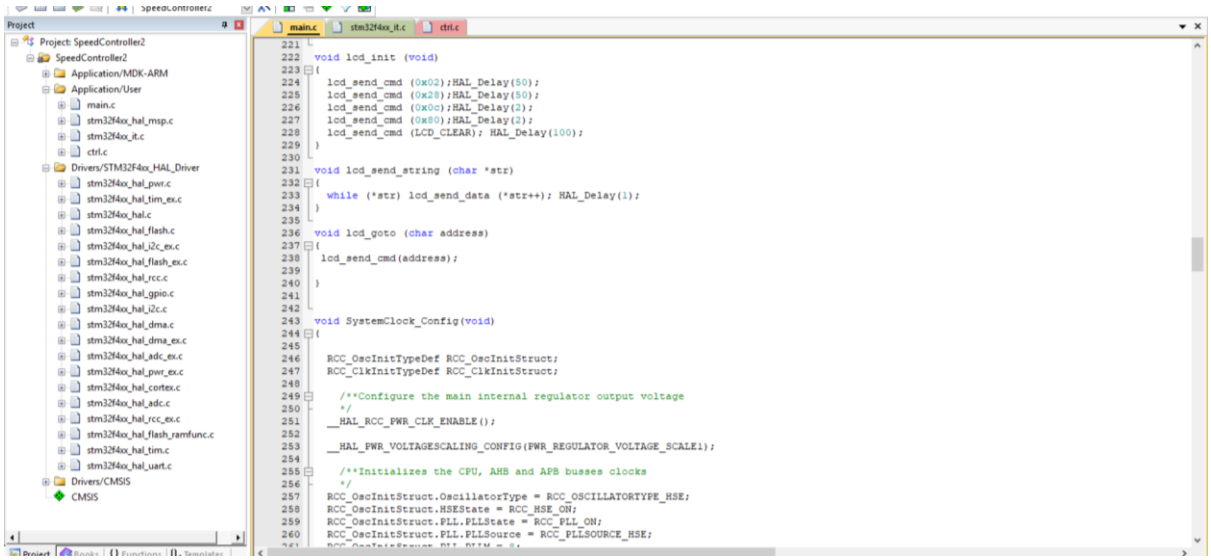

D:\Correct Code\SpeedController_with_UART_Matlab\MDK-ARM\SpeedController2.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
SpeedController2
main.c
stm32f4xx_it.c
ctrl.c
Project: SpeedController2
SpeedController2
Application/MDK-ARM
Application/User
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
ctrl.c
Drivers/STM32F4xx_HAL_Driver
stm32f4xx_hal_pwr.c
stm32f4xx_hal_tim_ex.c
stm32f4xx_hal.c
stm32f4xx_hal_flash.c
stm32f4xx_hal_i2c_ex.c
stm32f4xx_hal_flash_ex.c
stm32f4xx_hal_rcc.c
stm32f4xx_hal_gpio.c
stm32f4xx_hal_i2c.c
stm32f4xx_hal_dma.c
stm32f4xx_hal_dma_ex.c
stm32f4xx_hal_adc_ex.c
stm32f4xx_hal_pwr_ex.c
stm32f4xx_hal_cortex.c
stm32f4xx_hal_adc.c
stm32f4xx_hal_rcc_ex.c
stm32f4xx_hal_flash_ramfunc.c
stm32f4xx_hal_tim.c
stm32f4xx_hal_uart.c
Drivers/CMSIS
CMSIS
67 extern ExtU rtU;
68 extern ExtY rtY;
69
70 int main(void)
71 {
72 HAL_Init();
73 SystemClock_Config();
74
75 MX_GPIO_Init();
76
77 QEI_MX_TIM2_Init();
78
79 MX_I2C1_Init();
80 lcd_init();
81 lcd_send_string ("SpeedC");
82
83 PWM_MX_TIM3_Init();
84 PWMduty=0;
85 HAL_TIM_PWM_Start(&htim3,TIM_CHANNEL_4);
86 Motor(0);
87 __HAL_TIM_SetCompare(&htim3,TIM_CHANNEL_4,PWMduty);
88 TIM2->CNT=2000000000; //0xffffffff/2; //Start counter from middle
89
90 MX_ADC1_Init();
91 HAL_ADC_Start(&hadc1);
92 HAL_Delay(1);
93 adc_Value=HAL_ADC_GetValue(&hadc1);
94
95 ctrl_initialize();
96
97 MX_USART2_UART_Init();
98
99 ControlTick_MX_TIM7_Init();
100 HAL_NVIC_SetPriority(TIM7_IRQn, 0, 0);
101 HAL_NVIC_EnableIRQ(TIM7_IRQn);
102
103 char str[64];
104 unsigned long last_uart_control_tick_count=control_tick_count, last_lcd_control_tick_count=control_tick_count;
105 unsigned int n=0;
Project Books Functions Templates
Build Output
ST-Link Debugger
L:173 C:28
CAP NUM SCRL OVR R/W
Type here to search
12:44 AM
17-Jul-18


D:\Correct Code\SpeedController_with_UART_Matlab\MDK-ARM\SpeedController2.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
SpeedController2
main.c
stm32f4xx_it.c
ctrl.c
Project: SpeedController2
SpeedController2
Application/MDK-ARM
Application/User
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
ctrl.c
Drivers/STM32F4xx_HAL_Driver
stm32f4xx_hal_pwr.c
stm32f4xx_hal_tim_ex.c
stm32f4xx_hal.c
stm32f4xx_hal_flash.c
stm32f4xx_hal_i2c_ex.c
stm32f4xx_hal_flash_ex.c
stm32f4xx_hal_rcc.c
stm32f4xx_hal_gpio.c
stm32f4xx_hal_i2c.c
stm32f4xx_hal_dma.c
stm32f4xx_hal_dma_ex.c
stm32f4xx_hal_adc_ex.c
stm32f4xx_hal_pwr_ex.c
stm32f4xx_hal_cortex.c
stm32f4xx_hal_adc.c
stm32f4xx_hal_rcc_ex.c
stm32f4xx_hal_flash_ramfunc.c
stm32f4xx_hal_tim.c
stm32f4xx_hal_uart.c
Drivers/CMSIS
CMSIS
108 while (1)
109 {
110 if(control_tick_count!=last_uart_control_tick_count)
111 {
112 last_uart_control_tick_count=control_tick_count;
113
114 sprintf(str,"%.6lu,%5.0f,%5.0f,%+5.0f,%+5.0f,%.10d,%.3d\n", control_tick_count,
115 rtY.Set_Speed,
116 rtY.Current_Speed,
117 rtY.Current_Speed_Error,
118 rtY.PID_Result,
119 rtU.Current_motor_Position,
120 rtU.Target_Speed);
121
122 //sprintf(str,"%.6lu\n", control_tick_count);
123
124 n=0;
125 while(str[n]!=0)
126 n++;
127
128 HAL_UART_Transmit(&huart2, (unsigned char*)str, n, HAL_MAX_DELAY);
129 }
130
131 if((control_tick_count-last_lcd_control_tick_count)==8)
132 {
133 last_lcd_control_tick_count=control_tick_count;
134
135 lcd_goto(LCD_LINE2);
136 sprintf(str," nT:%.6lu",control_tick_count);
137 lcd_send_string(str);
138
139 sprintf(str,"U:%5.0f",rtY.Set_Speed); // "%6.1f" ->display results in 4 characters total and 0 decimal point
140 lcd_goto(LCD_LINE1+12);
141 lcd_send_string(str);
142
143 sprintf(str,"Y:%5.0f",rtY.Current_Speed);
144 lcd_goto(LCD_LINE2+12);
145 lcd_send_string(str);
146
Project Books Functions Templates
Build Output
ST-Link Debugger
L:173 C:28
CAP NUM SCRL OVR R/W
Type here to search
12:45 AM
17-Jul-18

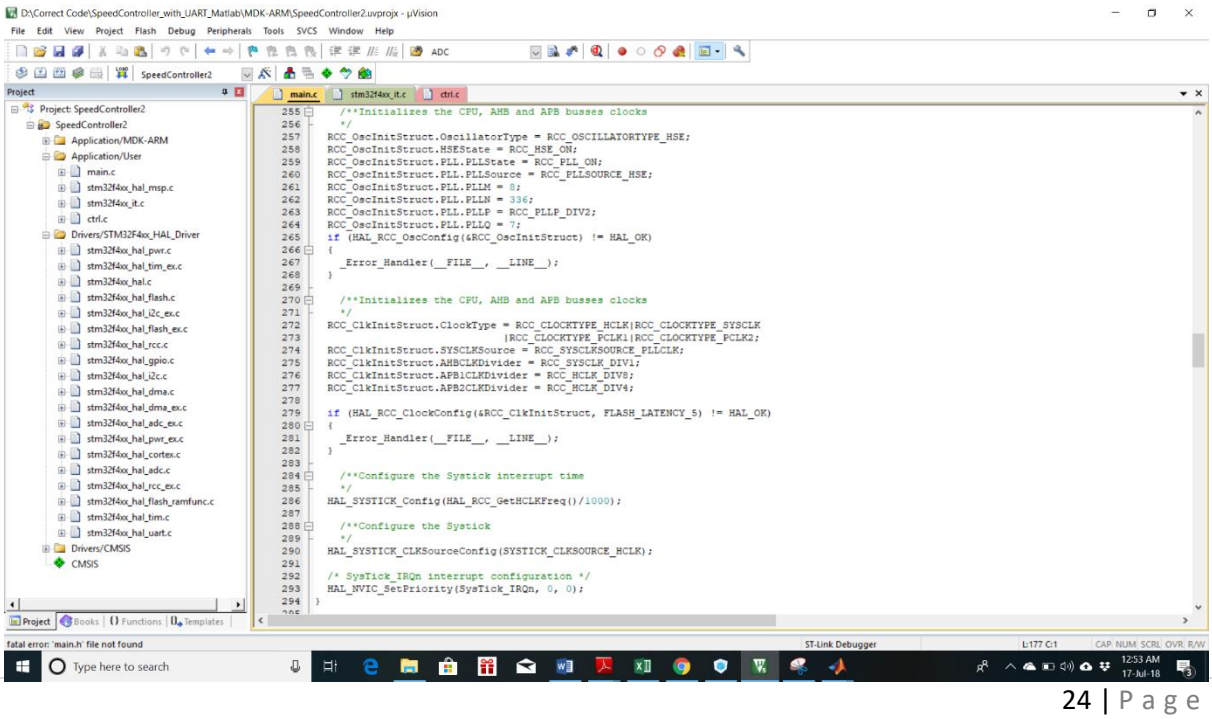

D:\Correct Code\SpeedController_with_UART_Matlab\MDK-ARM\SpeedController2.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
SpeedController2
main.c
stm32f4xx_it.c
ctrl.c
Project: SpeedController2
SpeedController2
Application/MDK-ARM
Application/User
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
ctrl.c
Drivers/STM32F4xx_HAL_Driver
stm32f4xx_hal_pwr.c
stm32f4xx_hal_tim_ex.c
stm32f4xx_hal.c
stm32f4xx_hal_flash.c
stm32f4xx_hal_i2c_ex.c
stm32f4xx_hal_flash_ex.c
stm32f4xx_hal_rcc.c
stm32f4xx_hal_gpio.c
stm32f4xx_hal_i2c.c
stm32f4xx_hal_dma.c
stm32f4xx_hal_dma_ex.c
stm32f4xx_hal_adc_ex.c
stm32f4xx_hal_pwr_ex.c
stm32f4xx_hal_cortex.c
stm32f4xx_hal_adc.c
stm32f4xx_hal_rcc_ex.c
stm32f4xx_hal_flash_ramfunc.c
stm32f4xx_hal_tim.c
stm32f4xx_hal_uart.c
Drivers/CMSIS
CMSIS
140 lcd_goto(LCD_LINE1+12);
141 lcd_send_string(str);
142
143 sprintf(str,"Y:%5.0f",rtY.Current_Speed);
144 lcd_goto(LCD_LINE2+12);
145 lcd_send_string(str);
146
147 sprintf(str,"E:%+5.0f",rtY.Current_Speed_Error); //display error in 5 digits including + or - mark and no decimals
148 lcd_goto(LCD_LINE3+12);
149 lcd_send_string(str);
150
151 sprintf(str,"PID:%+4.0f",rtY.PID_Result);
152 lcd_goto(LCD_LINE3);
153 lcd_send_string(str);
154
155 sprintf(str,"QEI:%08X",rtU.Current_motor_Position);
156 lcd_goto(LCD_LINE4);
157 lcd_send_string(str);
158
159 HAL_ADC_Start(&hadc1);
160 lcd_goto(LCD_LINE4+17);
161 sprintf(str,"%.3d",HAL_ADC_GetValue(&hadc1));
162 lcd_send_string(str);
163
164 sprintf(str,"%.3d",rtU.Target_Speed);
165 lcd_goto(LCD_LINE4+13);
166 lcd_send_string(str);
167
168 }
169
170 if(HAL_GPIO_ReadPin(GPIOA,GPIO_PIN_0) ==1)
171 {
172 HAL_ADC_Start(&hadc1);
173 adc_Value=HAL_ADC_GetValue(&hadc1);
174 HAL_Delay(1);
175 }
176 }
177
178 }
Project Books Functions Templates
Build Output
fatal error: 'main.h' file not found
ST-Link Debugger
L:173 C:28
CAP NUM SCRL OVR R/W
Type here to search
12:46 AM
17-Jul-18



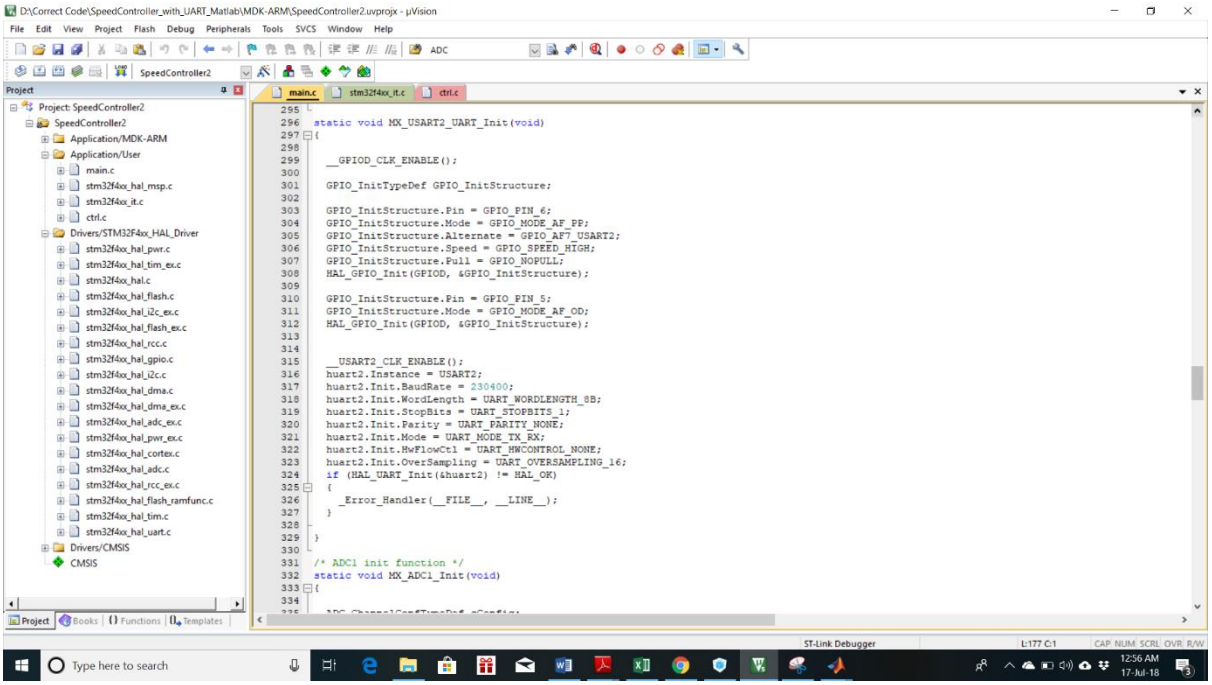
```
void Motor(unsigned char dirn)
{
  if (dirn==1)
  {
   HAL_GPIO_WritePin(GPIOD,GPIO_PIN_12,(GPIO_PinState)0);
   HAL_GPIO_WritePin(GPIOD,GPIO_PIN_13,(GPIO_PinState)1);
  }
  else
  {
   HAL_GPIO_WritePin(GPIOD,GPIO_PIN_12,(GPIO_PinState)1);
   HAL_GPIO_WritePin(GPIOD,GPIO_PIN_13,(GPIO_PinState)0);
  }
}


void lcd_send_cmd (char cmd)
{
  char data_u, data_l;
  uint8_t data_t[4];
  data_u = cmd&0xf0;
  data_l = (cmd<<4)&0xf0;
  data_t[0] = data_u|0x0C;  //en=1, rs=0 Backlight=on;
  data_t[1] = data_u|0x80;  //en=0, rs=0 Backlight=on;
  data_t[2] = data_l|0x0C;  //en=1, rs=0 Backlight=on;
  data_t[3] = data_l|0x80;  //en=0, rs=0 Backlight=on;
  HAL_I2C_Master_Transmit (&hi2c1, (0x4F),(uint8_t *) data_t, 4, 100);
}

void lcd_send_data (char data)
{
  char data_u, data_l;
  uint8_t data_t[4];
  data_u = data&0xf0;
  data_l = (data<<4)&0xf0;
  data_t[0] = data_u|0x0D;  //en=1, rs=0 ,Backlight=on;
  data_t[1] = data_u|0x09;  //en=0, rs=0 Backlight=on;
  data_t[2] = data_l|0x0D;  //en=1, rs=0 Backlight=on;
  data_t[3] = data_l|0x09;  //en=0, rs=0 Backlight=on;
  HAL_I2C_Master_Transmit (&hi2c1, (0x4F),(uint8_t *) data_t, 4, 100);
}
```

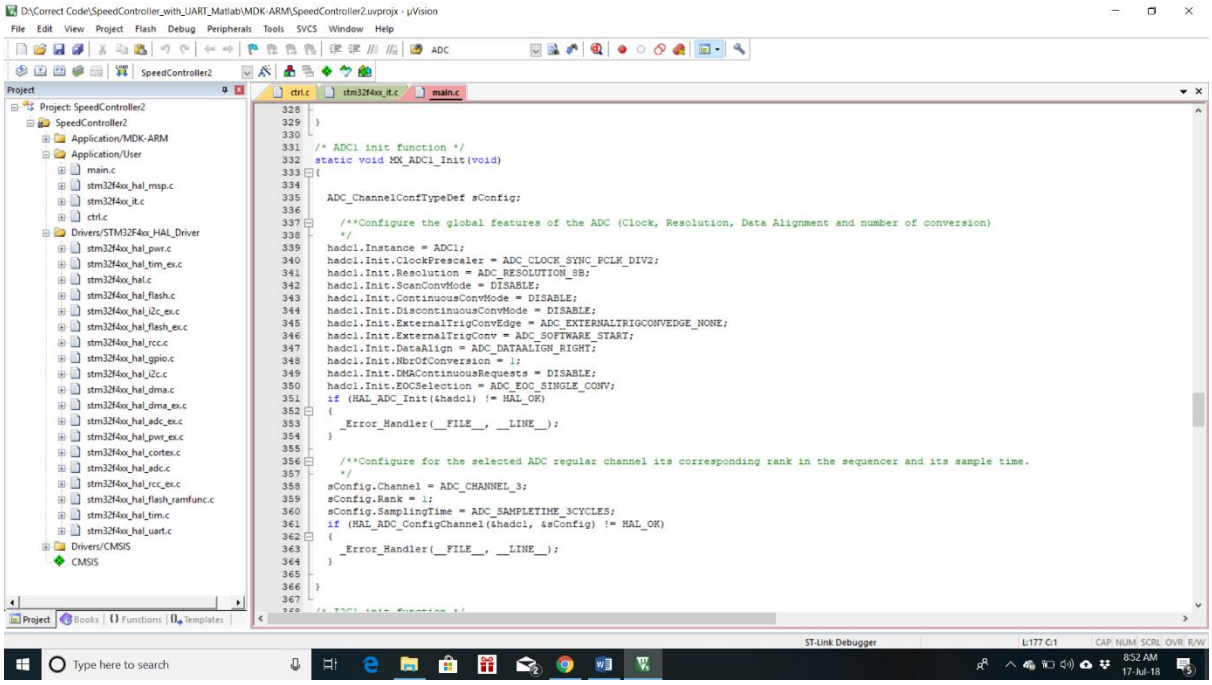


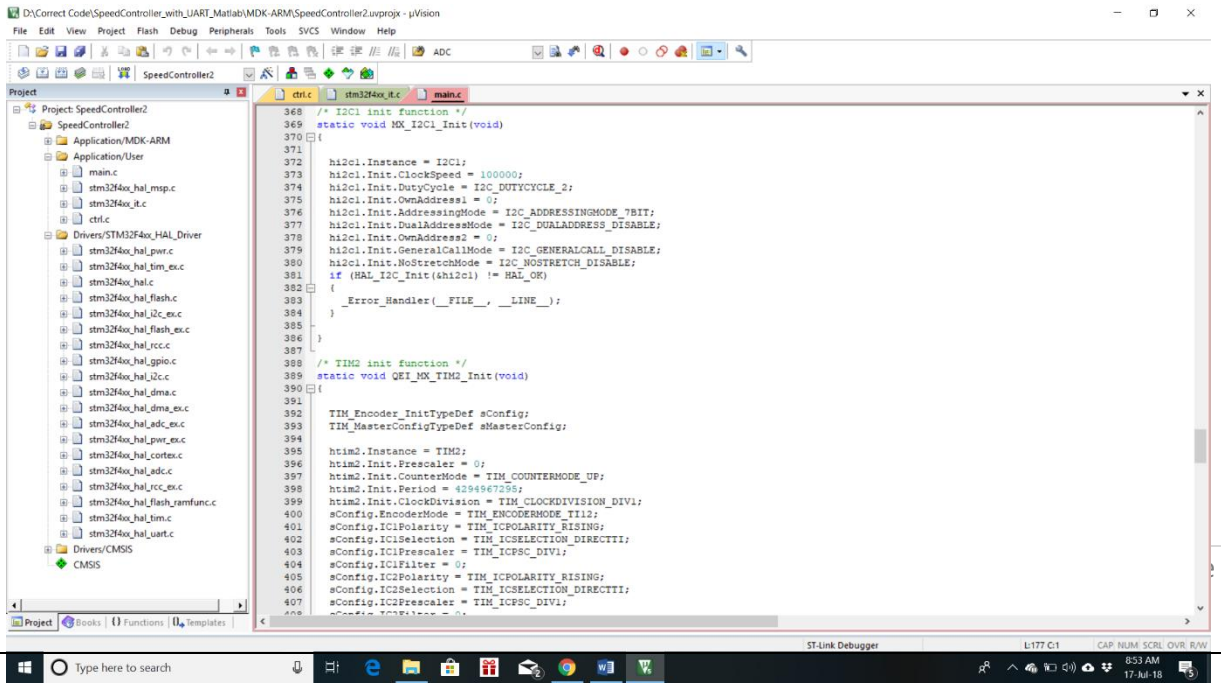
```
void lcd_init (void)
{
  lcd_send_cmd (0x02);HAL_Delay(50);
  lcd_send_cmd (0x28);HAL_Delay(50);
  lcd_send_cmd (0x0c);HAL_Delay(2);
  lcd_send_cmd (0x80);HAL_Delay(2);
  lcd_send_cmd (LCD_CLEAR); HAL_Delay(100);
}

void lcd_send_string (char *str)
{
  while (*str) lcd_send_data (*str++); HAL_Delay(1);
}

void lcd_goto (char address)
{
 lcd_send_cmd(address);

}


void SystemClock_Config(void)
{

  RCC_OscInitTypeDef RCC_OscInitStruct;
  RCC_ClkInitTypeDef RCC_ClkInitStruct;

    /**Configure the main internal regulator output voltage
    */
  __HAL_RCC_PWR_CLK_ENABLE();

  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);

    /**Initializes the CPU, AHB and APB busses clocks
    */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
```

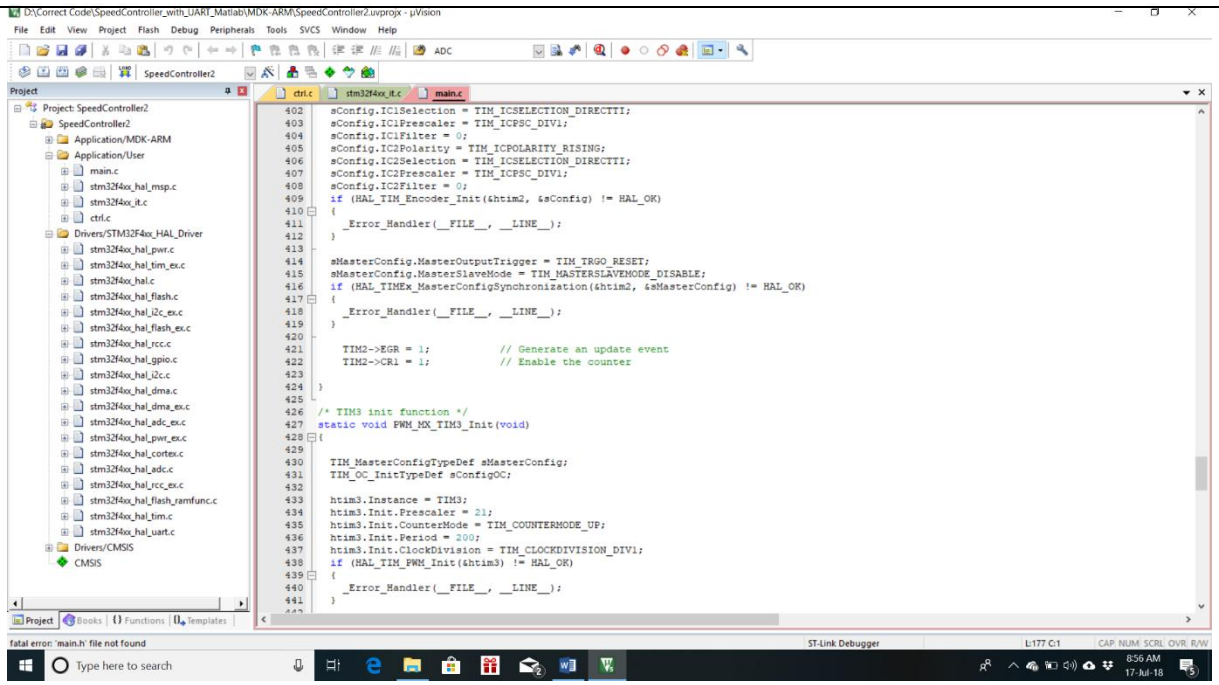


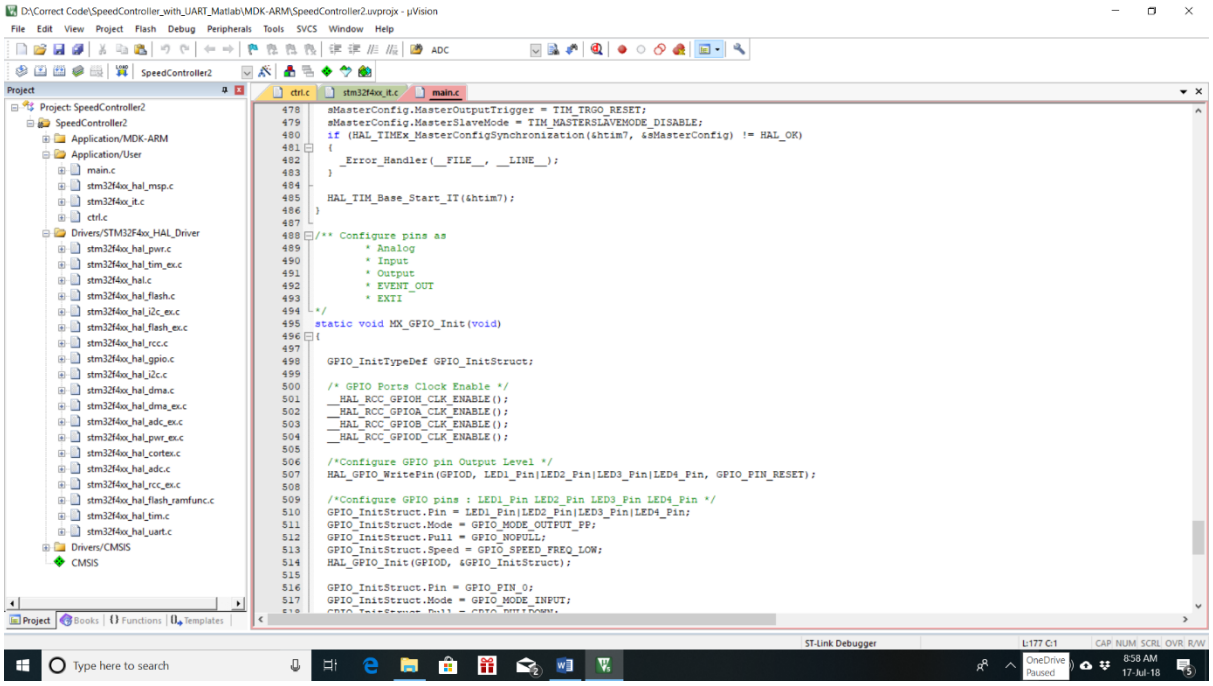
```
    /**Initializes the CPU, AHB and APB busses clocks
    */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
  RCC_OscInitStruct.PLL.PLLM = 8;
  RCC_OscInitStruct.PLL.PLLN = 336;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
  RCC_OscInitStruct.PLL.PLLQ = 7;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

    /**Initializes the CPU, AHB and APB busses clocks
    */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV8;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV4;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_5) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

    /**Configure the Systick interrupt time
    */
  HAL_SYSTICK_Config(HAL_RCC_GetHCLKFreq()/1000);

    /**Configure the Systick
    */
  HAL_SYSTICK_CLKSourceConfig(SYSTICK_CLKSOURCE_HCLK);

  /* SysTick_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(SysTick_IRQn, 0, 0);
}
```

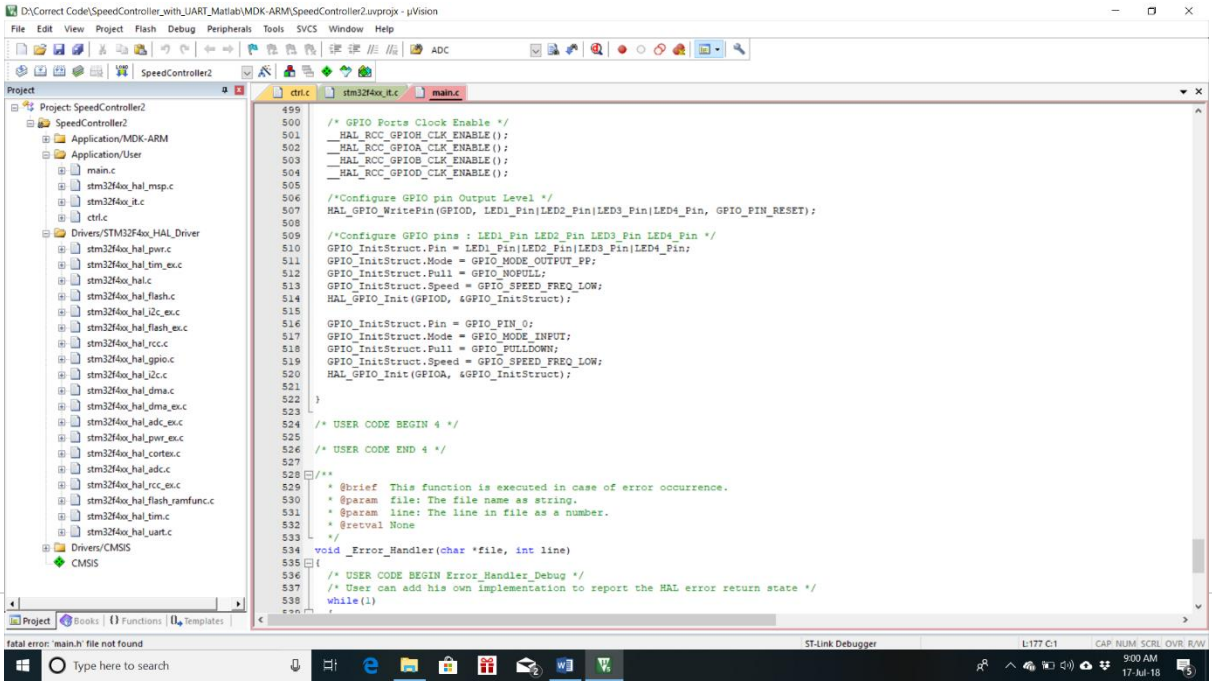

D:\Correct Code\SpeedController_with_UART_Matlab\MDK-ARM\SpeedController2.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Project: SpeedController2
SpeedController2
Application/MDK-ARM
Application/User
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
ctrl.c
Drivers/STM32F4xx_HAL_Driver
stm32f4xx_hal_pwr.c
stm32f4xx_hal_tim_ex.c
stm32f4xx_hal.c
stm32f4xx_hal_flash.c
stm32f4xx_hal_i2c_ex.c
stm32f4xx_hal_flash_ex.c
stm32f4xx_hal_rcc.c
stm32f4xx_hal_gpio.c
stm32f4xx_hal_i2c.c
stm32f4xx_hal_dma.c
stm32f4xx_hal_dma_ex.c
stm32f4xx_hal_adc_ex.c
stm32f4xx_hal_pwr_ex.c
stm32f4xx_hal_cortex.c
stm32f4xx_hal_adc.c
stm32f4xx_hal_rcc_ex.c
stm32f4xx_hal_flash_ramfunc.c
stm32f4xx_hal_tim.c
stm32f4xx_hal_uart.c
Drivers/CMSIS
CMSIS
main.c
stm32f4xx_it.c
ctrl.c
static void MX_USART2_UART_Init(void)
{
__GPIOD_CLK_ENABLE();
GPIO_InitTypeDef GPIO_InitStructure;
GPIO_InitStructure.Pin = GPIO_PIN_6;
GPIO_InitStructure.Mode = GPIO_MODE_AF_PP;
GPIO_InitStructure.Alternate = GPIO_AF7_USART2;
GPIO_InitStructure.Speed = GPIO_SPEED_HIGH;
GPIO_InitStructure.Pull = GPIO_NOPULL;
HAL_GPIO_Init(GPIOD, &GPIO_InitStructure);
GPIO_InitStructure.Pin = GPIO_PIN_5;
GPIO_InitStructure.Mode = GPIO_MODE_AF_OD;
HAL_GPIO_Init(GPIOD, &GPIO_InitStructure);
__USART2_CLK_ENABLE();
huart2.Instance = USART2;
huart2.Init.BaudRate = 230400;
huart2.Init.WordLength = UART_WORDLENGTH_8B;
huart2.Init.StopBits = UART_STOPBITS_1;
huart2.Init.Parity = UART_PARITY_NONE;
huart2.Init.Mode = UART_MODE_TX_RX;
huart2.Init.HwFlowCtl = UART_HWCONTROL_NONE;
huart2.Init.OverSampling = UART_OVERSAMPLING_16;
if (HAL_UART_Init(&huart2) != HAL_OK)
{
_Error_Handler(__FILE__, __LINE__);
}
}
/* ADC1 init function */
static void MX_ADC1_Init(void)
{
ST-Link Debugger
L:177 C:1
CAP NUM SCRL OVR R/W
Type here to search
12:56 AM
17-Jul-18


D:\Correct Code\SpeedController_with_UART_Matlab\MDK-ARM\SpeedController2.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Project: SpeedController2
SpeedController2
Application/MDK-ARM
Application/User
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
ctrl.c
Drivers/STM32F4xx_HAL_Driver
stm32f4xx_hal_pwr.c
stm32f4xx_hal_tim_ex.c
stm32f4xx_hal.c
stm32f4xx_hal_flash.c
stm32f4xx_hal_i2c_ex.c
stm32f4xx_hal_flash_ex.c
stm32f4xx_hal_rcc.c
stm32f4xx_hal_gpio.c
stm32f4xx_hal_i2c.c
stm32f4xx_hal_dma.c
stm32f4xx_hal_dma_ex.c
stm32f4xx_hal_adc_ex.c
stm32f4xx_hal_pwr_ex.c
stm32f4xx_hal_cortex.c
stm32f4xx_hal_adc.c
stm32f4xx_hal_rcc_ex.c
stm32f4xx_hal_flash_ramfunc.c
stm32f4xx_hal_tim.c
stm32f4xx_hal_uart.c
Drivers/CMSIS
CMSIS
ctrl.c
stm32f4xx_it.c
main.c
}
/* ADC1 init function */
static void MX_ADC1_Init(void)
{
ADC_ChannelConfTypeDef sConfig;
/**Configure the global features of the ADC (Clock, Resolution, Data Alignment and number of conversion)
*/
hadc1.Instance = ADC1;
hadc1.Init.ClockPrescaler = ADC_CLOCK_SYNC_PCLK_DIV2;
hadc1.Init.Resolution = ADC_RESOLUTION_8B;
hadc1.Init.ScanConvMode = DISABLE;
hadc1.Init.ContinuousConvMode = DISABLE;
hadc1.Init.DiscontinuousConvMode = DISABLE;
hadc1.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
hadc1.Init.ExternalTrigConv = ADC_SOFTWARE_START;
hadc1.Init.DataAlign = ADC_DATAALIGN_RIGHT;
hadc1.Init.NbrOfConversion = 1;
hadc1.Init.DMAContinuousRequests = DISABLE;
hadc1.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
if (HAL_ADC_Init(&hadc1) != HAL_OK)
{
_Error_Handler(__FILE__, __LINE__);
}
/**Configure for the selected ADC regular channel its corresponding rank in the sequencer and its sample time.
*/
sConfig.Channel = ADC_CHANNEL_3;
sConfig.Rank = 1;
sConfig.SamplingTime = ADC_SAMPLETIME_3CYCLES;
if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
{
_Error_Handler(__FILE__, __LINE__);
}
}
ST-Link Debugger
L:177 C:1
CAP NUM SCRL OVR R/W
Type here to search
8:52 AM
17-Jul-18


D:\Correct Code\SpeedController_with_UART_Matlab\MDK-ARM\SpeedController2.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Project: SpeedController2
SpeedController2
Application/MDK-ARM
Application/User
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
ctrl.c
Drivers/STM32F4xx_HAL_Driver
stm32f4xx_hal_pwr.c
stm32f4xx_hal_tim_ex.c
stm32f4xx_hal.c
stm32f4xx_hal_flash.c
stm32f4xx_hal_i2c_ex.c
stm32f4xx_hal_flash_ex.c
stm32f4xx_hal_rcc.c
stm32f4xx_hal_gpio.c
stm32f4xx_hal_i2c.c
stm32f4xx_hal_dma.c
stm32f4xx_hal_dma_ex.c
stm32f4xx_hal_adc_ex.c
stm32f4xx_hal_pwr_ex.c
stm32f4xx_hal_cortex.c
stm32f4xx_hal_adc.c
stm32f4xx_hal_rcc_ex.c
stm32f4xx_hal_flash_ramfunc.c
stm32f4xx_hal_tim.c
stm32f4xx_hal_uart.c
Drivers/CMSIS
CMSIS
ctrl.c
stm32f4xx_it.c
main.c
/* I2C1 init function */
static void MX_I2C1_Init(void)
{
hi2c1.Instance = I2C1;
hi2c1.Init.ClockSpeed = 100000;
hi2c1.Init.DutyCycle = I2C_DUTYCYCLE_2;
hi2c1.Init.OwnAddress1 = 0;
hi2c1.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
hi2c1.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
hi2c1.Init.OwnAddress2 = 0;
hi2c1.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
hi2c1.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
if (HAL_I2C_Init(&hi2c1) != HAL_OK)
{
_Error_Handler(__FILE__, __LINE__);
}
}
/* TIM2 init function */
static void QEI_MX_TIM2_Init(void)
{
TIM_Encoder_InitTypeDef sConfig;
TIM_MasterConfigTypeDef sMasterConfig;
htim2.Instance = TIM2;
htim2.Init.Prescaler = 0;
htim2.Init.CounterMode = TIM_COUNTERMODE_UP;
htim2.Init.Period = 4294967295;
htim2.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
sConfig.EncoderMode = TIM_ENCODERMODE_TI12;
sConfig.IC1Polarity = TIM_ICPOLARITY_RISING;
sConfig.IC1Selection = TIM_ICSELECTION_DIRECTTI;
sConfig.IC1Prescaler = TIM_ICPSC_DIV1;
sConfig.IC1Filter = 0;
sConfig.IC2Polarity = TIM_ICPOLARITY_RISING;
sConfig.IC2Selection = TIM_ICSELECTION_DIRECTTI;
sConfig.IC2Prescaler = TIM_ICPSC_DIV1;
ST-Link Debugger
L:177 C:1
CAP NUM SCRL OVR R/W
Type here to search
8:53 AM
17-Jul-18


D:\Correct Code\SpeedController_with_UART_Matlab\MDK-ARM\SpeedController2.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
SpeedController2
Project
Project: SpeedController2
SpeedController2
Application/MDK-ARM
Application/User
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
ctrl.c
Drivers/STM32F4xx_HAL_Driver
stm32f4xx_hal_pwr.c
stm32f4xx_hal_tim_ex.c
stm32f4xx_hal.c
stm32f4xx_hal_flash.c
stm32f4xx_hal_i2c_ex.c
stm32f4xx_hal_flash_ex.c
stm32f4xx_hal_rcc.c
stm32f4xx_hal_gpio.c
stm32f4xx_hal_i2c.c
stm32f4xx_hal_dma.c
stm32f4xx_hal_dma_ex.c
stm32f4xx_hal_adc_ex.c
stm32f4xx_hal_pwr_ex.c
stm32f4xx_hal_cortex.c
stm32f4xx_hal_adc.c
stm32f4xx_hal_rcc_ex.c
stm32f4xx_hal_flash_ramfunc.c
stm32f4xx_hal_tim.c
stm32f4xx_hal_uart.c
Drivers/CMSIS
CMSIS
ctrl.c
stm32f4xx_it.c
main.c
sConfig.IC1Selection = TIM_ICSELECTION_DIRECTTI;
sConfig.IC1Prescaler = TIM_ICPSC_DIV1;
sConfig.IC1Filter = 0;
sConfig.IC2Polarity = TIM_ICPOLARITY_RISING;
sConfig.IC2Selection = TIM_ICSELECTION_DIRECTTI;
sConfig.IC2Prescaler = TIM_ICPSC_DIV1;
sConfig.IC2Filter = 0;
if (HAL_TIM_Encoder_Init(&htim2, &sConfig) != HAL_OK)
{
_Error_Handler(__FILE__, __LINE__);
}
sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
if (HAL_TIMEx_MasterConfigSynchronization(&htim2, &sMasterConfig) != HAL_OK)
{
_Error_Handler(__FILE__, __LINE__);
}
TIM2->EGR = 1; // Generate an update event
TIM2->CR1 = 1; // Enable the counter
}
/* TIM3 init function */
static void PWM_MX_TIM3_Init(void)
{
TIM_MasterConfigTypeDef sMasterConfig;
TIM_OC_InitTypeDef sConfigOC;
htim3.Instance = TIM3;
htim3.Init.Prescaler = 21;
htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
htim3.Init.Period = 200;
htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
if (HAL_TIM_PWM_Init(&htim3) != HAL_OK)
{
_Error_Handler(__FILE__, __LINE__);
}
Project Books Functions Templates
fatal error: 'main.h' file not found
ST-Link Debugger
L:177 C:1
CAP NUM SCRL OVR R/W
Type here to search
8:56 AM
17-Jul-18


D:\Correct Code\SpeedController_with_UART_Matlab\MDK-ARM\SpeedController2.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
SpeedController2
Project
Project: SpeedController2
SpeedController2
Application/MDK-ARM
Application/User
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
ctrl.c
Drivers/STM32F4xx_HAL_Driver
stm32f4xx_hal_pwr.c
stm32f4xx_hal_tim_ex.c
stm32f4xx_hal.c
stm32f4xx_hal_flash.c
stm32f4xx_hal_i2c_ex.c
stm32f4xx_hal_flash_ex.c
stm32f4xx_hal_rcc.c
stm32f4xx_hal_gpio.c
stm32f4xx_hal_i2c.c
stm32f4xx_hal_dma.c
stm32f4xx_hal_dma_ex.c
stm32f4xx_hal_adc_ex.c
stm32f4xx_hal_pwr_ex.c
stm32f4xx_hal_cortex.c
stm32f4xx_hal_adc.c
stm32f4xx_hal_rcc_ex.c
stm32f4xx_hal_flash_ramfunc.c
stm32f4xx_hal_tim.c
stm32f4xx_hal_uart.c
Drivers/CMSIS
CMSIS
ctrl.c
stm32f4xx_it.c
main.c
sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
if (HAL_TIMEx_MasterConfigSynchronization(&htim7, &sMasterConfig) != HAL_OK)
{
_Error_Handler(__FILE__, __LINE__);
}
HAL_TIM_Base_Start_IT(&htim7);
}
/** Configure pins as
* Analog
* Input
* Output
* EVENT_OUT
* EXTI
*/
static void MX_GPIO_Init(void)
{
GPIO_InitTypeDef GPIO_InitStruct;
/* GPIO Ports Clock Enable */
__HAL_RCC_GPIOH_CLK_ENABLE();
__HAL_RCC_GPIOA_CLK_ENABLE();
__HAL_RCC_GPIOB_CLK_ENABLE();
__HAL_RCC_GPIOD_CLK_ENABLE();
/*Configure GPIO pin Output Level */
HAL_GPIO_WritePin(GPIOD, LED1_Pin|LED2_Pin|LED3_Pin|LED4_Pin, GPIO_PIN_RESET);
/*Configure GPIO pins : LED1_Pin LED2_Pin LED3_Pin LED4_Pin */
GPIO_InitStruct.Pin = LED1_Pin|LED2_Pin|LED3_Pin|LED4_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(GPIOD, &GPIO_InitStruct);
GPIO_InitStruct.Pin = GPIO_PIN_0;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
Project Books Functions Templates
ST-Link Debugger
L:177 C:1
CAP NUM SCRL OVR R/W
Type here to search
OneDrive Paused
8:58 AM
17-Jul-18


D:\Correct Code\SpeedController_with_UART_Matlab\MDK-ARM\SpeedController2.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
SpeedController2
Project
Project: SpeedController2
SpeedController2
Application/MDK-ARM
Application/User
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
ctrl.c
Drivers/STM32F4xx_HAL_Driver
stm32f4xx_hal_pwr.c
stm32f4xx_hal_tim_ex.c
stm32f4xx_hal.c
stm32f4xx_hal_flash.c
stm32f4xx_hal_i2c_ex.c
stm32f4xx_hal_flash_ex.c
stm32f4xx_hal_rcc.c
stm32f4xx_hal_gpio.c
stm32f4xx_hal_i2c.c
stm32f4xx_hal_dma.c
stm32f4xx_hal_dma_ex.c
stm32f4xx_hal_adc_ex.c
stm32f4xx_hal_pwr_ex.c
stm32f4xx_hal_cortex.c
stm32f4xx_hal_adc.c
stm32f4xx_hal_rcc_ex.c
stm32f4xx_hal_flash_ramfunc.c
stm32f4xx_hal_tim.c
stm32f4xx_hal_uart.c
Drivers/CMSIS
CMSIS
ctrl.c
stm32f4xx_it.c
main.c
/* GPIO Ports Clock Enable */
__HAL_RCC_GPIOH_CLK_ENABLE();
__HAL_RCC_GPIOA_CLK_ENABLE();
__HAL_RCC_GPIOB_CLK_ENABLE();
__HAL_RCC_GPIOD_CLK_ENABLE();
/*Configure GPIO pin Output Level */
HAL_GPIO_WritePin(GPIOD, LED1_Pin|LED2_Pin|LED3_Pin|LED4_Pin, GPIO_PIN_RESET);
/*Configure GPIO pins : LED1_Pin LED2_Pin LED3_Pin LED4_Pin */
GPIO_InitStruct.Pin = LED1_Pin|LED2_Pin|LED3_Pin|LED4_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(GPIOD, &GPIO_InitStruct);
GPIO_InitStruct.Pin = GPIO_PIN_0;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_PULLDOWN;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
}
/* USER CODE BEGIN 4 */
/* USER CODE END 4 */
/**
* @brief This function is executed in case of error occurrence.
* @param file: The file name as string.
* @param line: The line in file as a number.
* @retval None
*/
void _Error_Handler(char *file, int line)
{
/* USER CODE BEGIN Error_Handler_Debug */
/* User can add his own implementation to report the HAL error return state */
while(1)
Project Books Functions Templates
fatal error: 'main.h' file not found
ST-Link Debugger
L:177 C:1
CAP NUM SCRL OVR R/W
Type here to search
9:00 AM
17-Jul-18

The ctrl.c file which was added from the SIMULINK Model

```
/*
 * File: ctrl.c
 *
 * Code generated for Simulink model 'ctrl'.
 *
 * Model version                  : 1.88
 * Simulink Coder version         : 8.13 (R2017b) 24-Jul-2017
 * C/C++ source code generated on : Thu May 31 16:54:13 2018
 *
 * Target selection: ert.tlc
 * Embedded hardware selection: ARM Compatible->ARM Cortex
 * Code generation objectives:
 *    1. Execution efficiency
 *    2. RAM efficiency
 * Validation result: Not run
 */

#include "ctrl.h"
#define NumBitsPerChar                 8U

/* Block signals and states (auto storage) */
DW rtDW;

/* External inputs (root inport signals with auto storage) */
ExtU rtU;

/* External outputs (root outports fed by signals with auto storage) */
ExtY rtY;

/* Real-time model */
RT_MODEL rtM_;
RT_MODEL *const rtM = &rtM_;
extern real_T rtGetNaN(void);
extern real32_T rtGetNaNF(void);
extern real_T rtInf;
extern real_T rtMinusInf;
extern real_T rtNaN;
extern real32_T rtInfF;
extern real32_T rtMinusInfF;
extern real32_T rtNaNF;
extern void rt_InitInfAndNaN(size_t realSize);
extern boolean_T rtIsInf(real_T value);
extern boolean_T rtIsInfF(real32_T value);
extern boolean_T rtIsNaN(real_T value);
extern boolean_T rtIsNaNF(real32_T value);
typedef struct {
  struct {
    uint32_T wordH;
    uint32_T wordL;
  } words;
} BigEndianIEEEDouble;

typedef struct {
  struct {
    uint32_T wordL;
    uint32_T wordH;
  } words;
} LittleEndianIEEEDouble;

typedef struct {
  union {
    real32_T wordLreal;
    uint32_T wordLuint;
  } wordL;
} IEEESingle;

real_T rtInf;
real_T rtMinusInf;
real_T rtNaN;
```

```
real32_T rtInfF;
real32_T rtMinusInfF;
real32_T rtNaNF;
extern real_T rtGetInf(void);
extern real32_T rtGetInfF(void);
extern real_T rtGetMinusInf(void);
extern real32_T rtGetMinusInfF(void);

/*
 * Initialize rtNaN needed by the generated code.
 * NaN is initialized as non-signaling. Assumes IEEE.
 */
real_T rtGetNaN(void)
{
  size_t bitsPerReal = sizeof(real_T) * (NumBitsPerChar);
  real_T nan = 0.0;
  if (bitsPerReal == 32U) {
    nan = rtGetNaNF();
  } else {
    union {
      LittleEndianIEEEDouble bitVal;
      real_T fltVal;
    } tmpVal;

    tmpVal.bitVal.words.wordH = 0xFFF80000U;
    tmpVal.bitVal.words.wordL = 0x00000000U;
    nan = tmpVal.fltVal;
  }

  return nan;
}

/*
 * Initialize rtNaNF needed by the generated code.
 * NaN is initialized as non-signaling. Assumes IEEE.
 */
real32_T rtGetNaNF(void)
{
  IEEESingle nanF = { { 0 } };

  nanF.wordL.wordLuint = 0xFFC00000U;
  return nanF.wordL.wordLreal;
}

/*
 * Initialize the rtInf, rtMinusInf, and rtNaN needed by the
 * generated code. NaN is initialized as non-signaling. Assumes IEEE.
 */
void rt_InitInfAndNaN(size_t realSize)
{
  (void) (realSize);
  rtNaN = rtGetNaN();
  rtNaNF = rtGetNaNF();
  rtInf = rtGetInf();
  rtInfF = rtGetInfF();
  rtMinusInf = rtGetMinusInf();
  rtMinusInfF = rtGetMinusInfF();
}

/* Test if value is infinite */
boolean_T rtIsInf(real_T value)
{
  return (boolean_T)((value==rtInf || value==rtMinusInf) ? 1U : 0U);
}

/* Test if single-precision value is infinite */
boolean_T rtIsInfF(real32_T value)
{
  return (boolean_T)(((value)==rtInfF || (value)==rtMinusInfF) ? 1U : 0U);
}
```

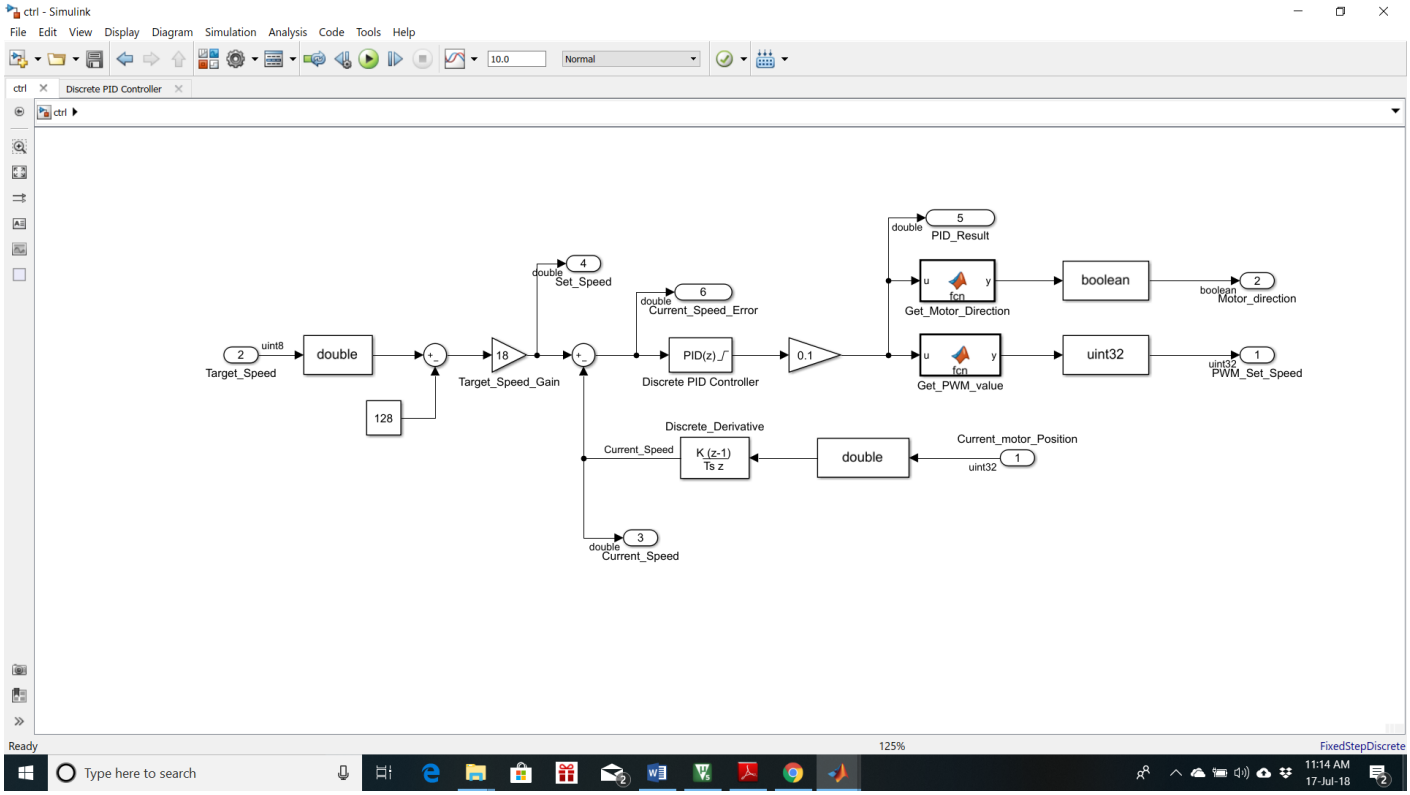
```
/* Test if value is not a number */
boolean_T rtIsNaN(real_T value)
{
  return (boolean_T)((value!=value) ? 1U : 0U);
}

/* Test if single-precision value is not a number */
boolean_T rtIsNaNF(real32_T value)
{
  return (boolean_T)(((value!=value) ? 1U : 0U));
}

/*
 * Initialize rtInf needed by the generated code.
 * Inf is initialized as non-signaling. Assumes IEEE.
 */
real_T rtGetInf(void)
{
  size_t bitsPerReal = sizeof(real_T) * (NumBitsPerChar);
  real_T inf = 0.0;
  if (bitsPerReal == 32U) {
    inf = rtGetInfF();
  } else {
    union {
      LittleEndianIEEEDouble bitVal;
      real_T fltVal;
    } tmpVal;

    tmpVal.bitVal.words.wordH = 0x7FF00000U;
    tmpVal.bitVal.words.wordL = 0x00000000U;
    inf = tmpVal.fltVal;
  }

  return inf;
}

/*
 * Initialize rtInfF needed by the generated code.
 * Inf is initialized as non-signaling. Assumes IEEE.
 */
real32_T rtGetInfF(void)
{
  IEEESingle infF;
  infF.wordL.wordLuint = 0x7F800000U;
  return infF.wordL.wordLreal;
}

/*
 * Initialize rtMinusInf needed by the generated code.
 * Inf is initialized as non-signaling. Assumes IEEE.
 */
real_T rtGetMinusInf(void)
{
  size_t bitsPerReal = sizeof(real_T) * (NumBitsPerChar);
  real_T minf = 0.0;
  if (bitsPerReal == 32U) {
    minf = rtGetMinusInfF();
  } else {
    union {
      LittleEndianIEEEDouble bitVal;
      real_T fltVal;
    } tmpVal;

    tmpVal.bitVal.words.wordH = 0xFFF00000U;
    tmpVal.bitVal.words.wordL = 0x00000000U;
    minf = tmpVal.fltVal;
  }

  return minf;
}
```

```
/*
 * Initialize rtMinusInfF needed by the generated code.
 * Inf is initialized as non-signaling. Assumes IEEE.
 */
real32_T rtGetMinusInfF(void)
{
  IEEESingle minfF;
  minfF.wordL.wordLuint = 0xFF800000U;
  return minfF.wordL.wordLreal;
}

/* Model step function */
void ctrl_step(void)
{
  real_T rtb_Target_Speed_Gain;
  real_T rtb_Diff;
  real_T rtb_TSamp;
  real_T rtb_IntegralGain;
  real_T rtb_FilterCoefficient;
  real_T rtb_SignPreIntegrator;
  real_T rtb_Gain;
  boolean_T rtb_NotEqual;
  int8_T rtb_SignPreIntegrator_0;

  /* Gain: '<Root>/Target_Speed_Gain' incorporates:
   *  Constant: '<Root>/Constant'
   *  DataTypeConversion: '<Root>/Data Type Conversion2'
   *  Inport: '<Root>/Target_Speed'
   *  Sum: '<Root>/Sum1'
   */
  rtb_Target_Speed_Gain = ((real_T)rtU.Target_Speed - 128.0) * 18.0;

  /* SampleTimeMath: '<S2>/TSamp' incorporates:
   *  DataTypeConversion: '<Root>/Data Type Conversion1'
   *  Inport: '<Root>/Current_motor_Position'
   *
   * About '<S2>/TSamp':
   *  y = u * K where K = 1 / ( w * Ts )
   */
  rtb_TSamp = (real_T)rtU.Current_motor_Position * 40.0;

  /* Sum: '<S2>/Diff' incorporates:
   *  UnitDelay: '<S2>/UD'
   *
   * Block description for '<S2>/Diff':
   *
   *  Add in CPU
   *
   * Block description for '<S2>/UD':
   *
   *  Store in Global RAM
   */
  rtb_Diff = rtb_TSamp - rtDW.UD_DSTATE;

  /* Sum: '<Root>/Sum' */
  rtb_IntegralGain = rtb_Target_Speed_Gain - rtb_Diff;

  /* Gain: '<S1>/Filter Coefficient' incorporates:
   *  DiscreteIntegrator: '<S1>/Filter'
   *  Gain: '<S1>/Derivative Gain'
   *  Sum: '<S1>/SumD'
   */
  rtb_FilterCoefficient = (0.005 * rtb_IntegralGain - rtDW.Filter_DSTATE) * 10.0;

  /* Sum: '<S1>/Sum' incorporates:
   *  DiscreteIntegrator: '<S1>/Integrator'
   *  Gain: '<S1>/Proportional Gain'
   */
  rtb_SignPreIntegrator = (0.35 * rtb_IntegralGain + rtDW.Integrator_DSTATE) +
    rtb_FilterCoefficient;
```

```
/* Saturate: '<S1>/Saturate' */
if (rtb_SignPreIntegrator > 1990.0) {
  rtb_Gain = 1990.0;
} else if (rtb_SignPreIntegrator < -1990.0) {
  rtb_Gain = -1990.0;
} else {
  rtb_Gain = rtb_SignPreIntegrator;
}

/* End of Saturate: '<S1>/Saturate' */

/* Gain: '<Root>/Gain' */
rtb_Gain *= 0.1;

/* MATLAB Function: '<Root>/Get_PWM_value' */
if (rtb_Gain < 0.0) {
  /* Outport: '<Root>/PWM_Set_Speed' incorporates:
   *  DataTypeConversion: '<Root>/Data Type Conversion'
   */
  rtY.PWM_Set_Speed = (uint32_T)(0.0 - rtb_Gain);
} else {
  /* Outport: '<Root>/PWM_Set_Speed' incorporates:
   *  DataTypeConversion: '<Root>/Data Type Conversion'
   */
  rtY.PWM_Set_Speed = (uint32_T)rtb_Gain;
}

/* End of MATLAB Function: '<Root>/Get_PWM_value' */

/* Outport: '<Root>/Motor_direction' incorporates:
 *  MATLAB Function: '<Root>/Get_Motor_Direction'
 */
rtY.Motor_direction = !(rtb_Gain < 0.0);

/* Outport: '<Root>/Current_Speed' */
rtY.Current_Speed = rtb_Diff;

/* Outport: '<Root>/Set_Speed ' */
rtY.Set_Speed = rtb_Target_Speed_Gain;

/* Outport: '<Root>/PID_Result' */
rtY.PID_Result = rtb_Gain;

/* Outport: '<Root>/Current_Speed_Error ' */
rtY.Current_Speed_Error = rtb_IntegralGain;

/* Gain: '<S5>/ZeroGain' */
rtb_Target_Speed_Gain = 0.0 * rtb_SignPreIntegrator;

/* DeadZone: '<S5>/DeadZone' */
if (rtb_SignPreIntegrator > 1990.0) {
  rtb_SignPreIntegrator -= 1990.0;
} else if (rtb_SignPreIntegrator >= -1990.0) {
  rtb_SignPreIntegrator = 0.0;
} else {
  rtb_SignPreIntegrator -= -1990.0;
}

/* End of DeadZone: '<S5>/DeadZone' */

/* RelationalOperator: '<S5>/NotEqual' */
rtb_NotEqual = (rtb_Target_Speed_Gain != rtb_SignPreIntegrator);

/* Signum: '<S5>/SignDeltaU' */
if (rtb_SignPreIntegrator < 0.0) {
  rtb_SignPreIntegrator = -1.0;
} else if (rtb_SignPreIntegrator > 0.0) {
  rtb_SignPreIntegrator = 1.0;
} else if (rtb_SignPreIntegrator == 0.0) {
  rtb_SignPreIntegrator = 0.0;
} else {
```

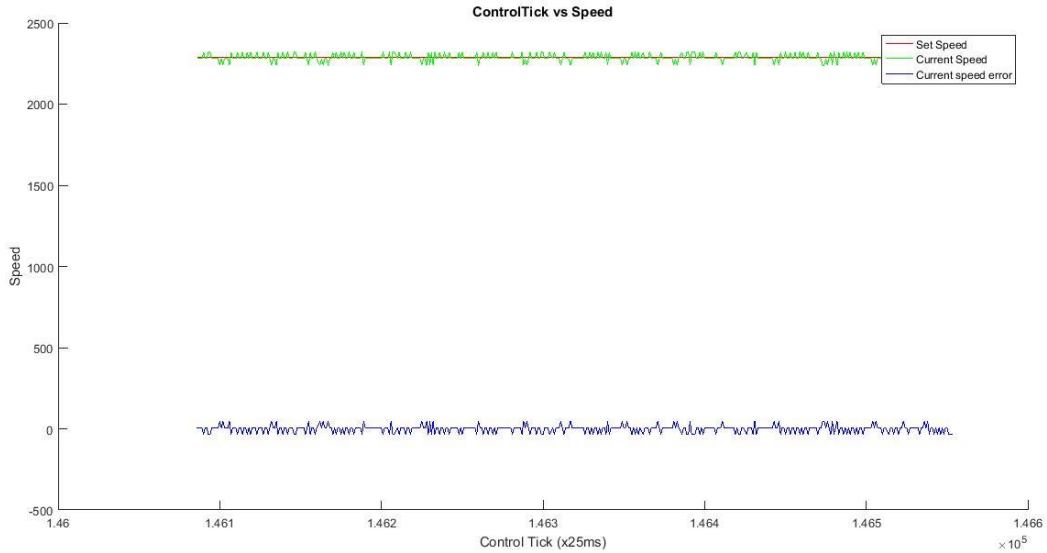
```
  rtb_SignPreIntegrator = (rtNaN);
 }

 /* End of Signum: '<S5>/SignDeltaU' */

 /* Gain: '<S1>/Integral Gain' */
 rtb_IntegralGain *= 1.5;

 /* Update for UnitDelay: '<S2>/UD'
  *
  * Block description for '<S2>/UD':
  *
  *  Store in Global RAM
  */
 rtDW.UD_DSTATE = rtb_TSamp;

 /* DataTypeConversion: '<S5>/DataTypeConv1' */
 if (rtb_SignPreIntegrator < 128.0) {
  rtb_SignPreIntegrator_0 = (int8_T)rtb_SignPreIntegrator;
 } else {
  rtb_SignPreIntegrator_0 = MAX_int8_T;
 }

 /* End of DataTypeConversion: '<S5>/DataTypeConv1' */

 /* Signum: '<S5>/SignPreIntegrator' */
 if (rtb_IntegralGain < 0.0) {
  rtb_Target_Speed_Gain = -1.0;
 } else if (rtb_IntegralGain > 0.0) {
  rtb_Target_Speed_Gain = 1.0;
 } else if (rtb_IntegralGain == 0.0) {
  rtb_Target_Speed_Gain = 0.0;
 } else {
  rtb_Target_Speed_Gain = (rtNaN);
 }

 /* End of Signum: '<S5>/SignPreIntegrator' */

 /* Switch: '<S1>/Switch' incorporates:
  *  Constant: '<S1>/Constant'
  *  DataTypeConversion: '<S5>/DataTypeConv2'
  *  Logic: '<S5>/AND'
  *  RelationalOperator: '<S5>/Equal'
  */
 if (rtb_NotEqual && (rtb_SignPreIntegrator_0 == (int8_T)rtb_Target_Speed_Gain))
 {
  rtb_IntegralGain = 0.0;
 }

 /* End of Switch: '<S1>/Switch' */

 /* Update for DiscreteIntegrator: '<S1>/Integrator' */
 rtDW.Integrator_DSTATE += 0.025 * rtb_IntegralGain;

 /* Update for DiscreteIntegrator: '<S1>/Filter' */
 rtDW.Filter_DSTATE += 0.025 * rtb_FilterCoefficient;
}

/* Model initialize function */
void ctrl_initialize(void)
{
 /* Registration code */

 /* initialize non-finites */
 rt_InitInfAndNaN(sizeof(real_T));

 /* InitializeConditions for UnitDelay: '<S2>/UD'
  *
  * Block description for '<S2>/UD':
  *
  *  Store in Global RAM
```

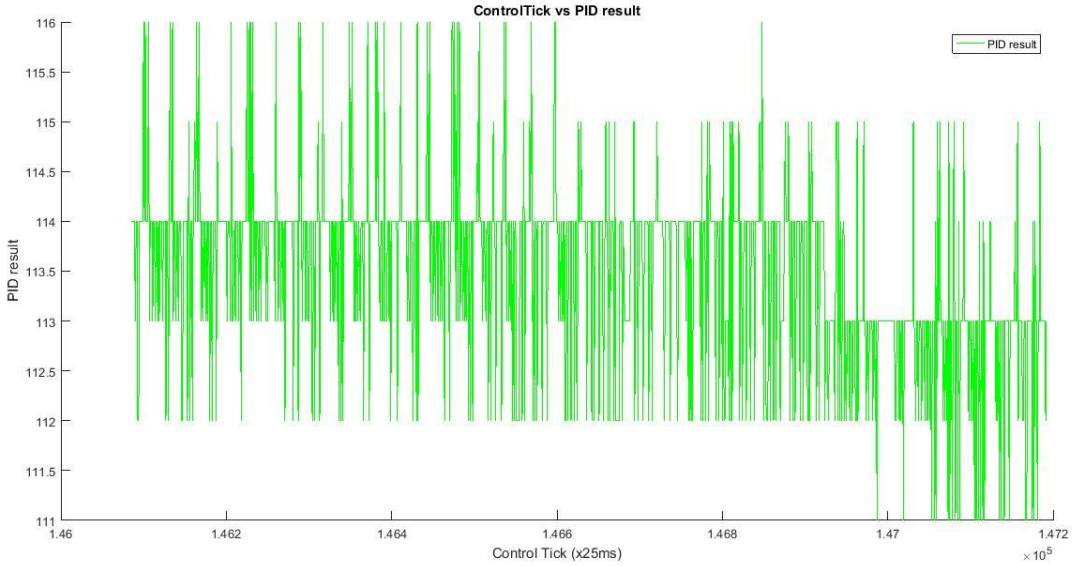
```
  */
 rtDW.UD_DSTATE = 2.0E+9;
}

/*
 * File trailer for generated code.
 *
 * [EOF]
 */
```



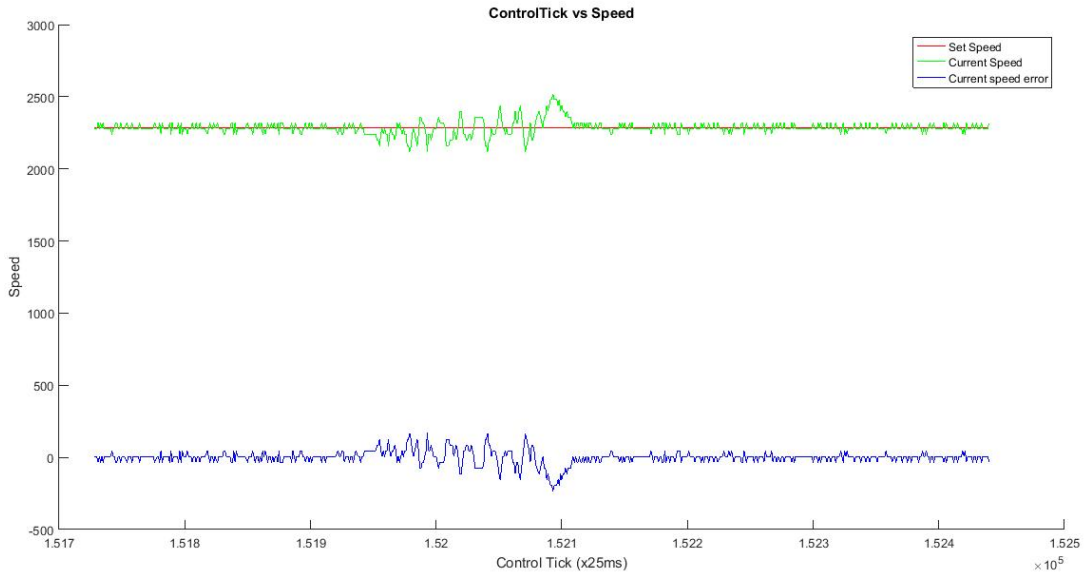
Results

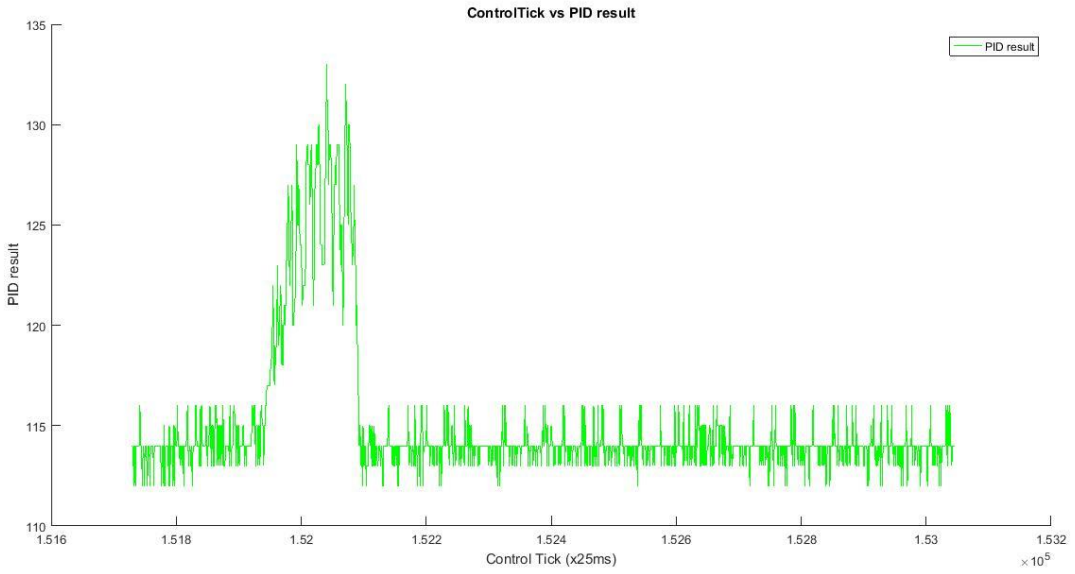
1. *Constant Speed without Load*



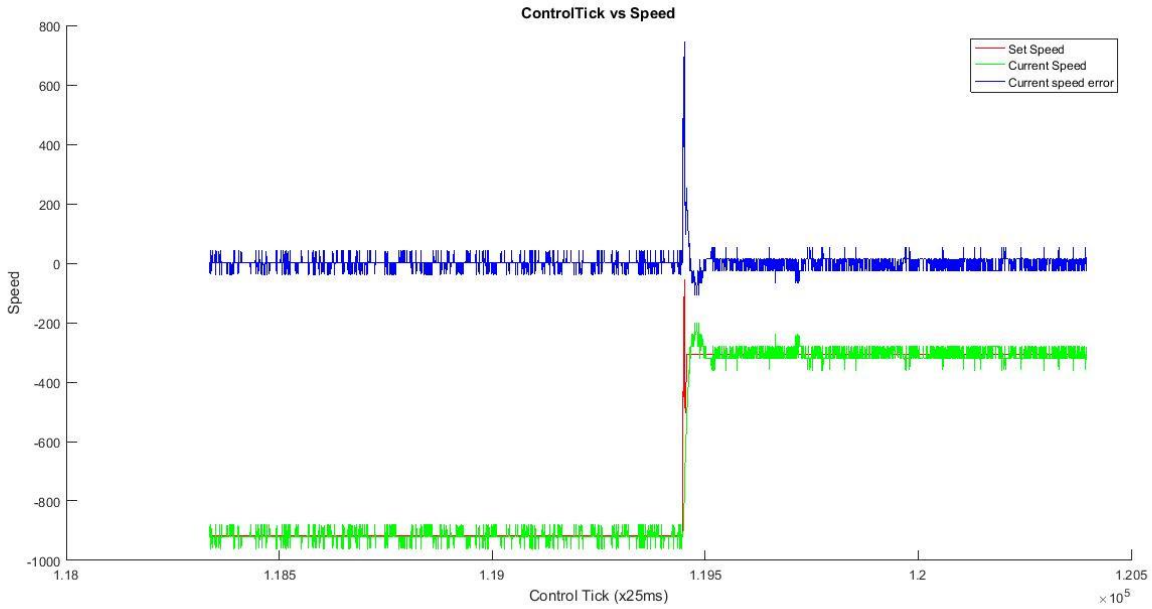


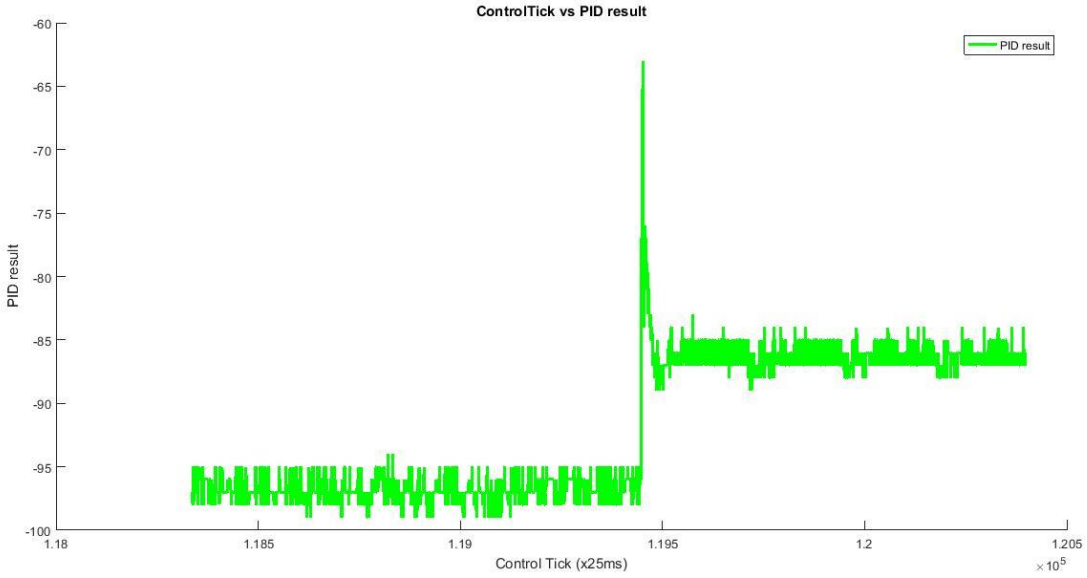
2. *Constant Speed Adding Load for a Little Time*



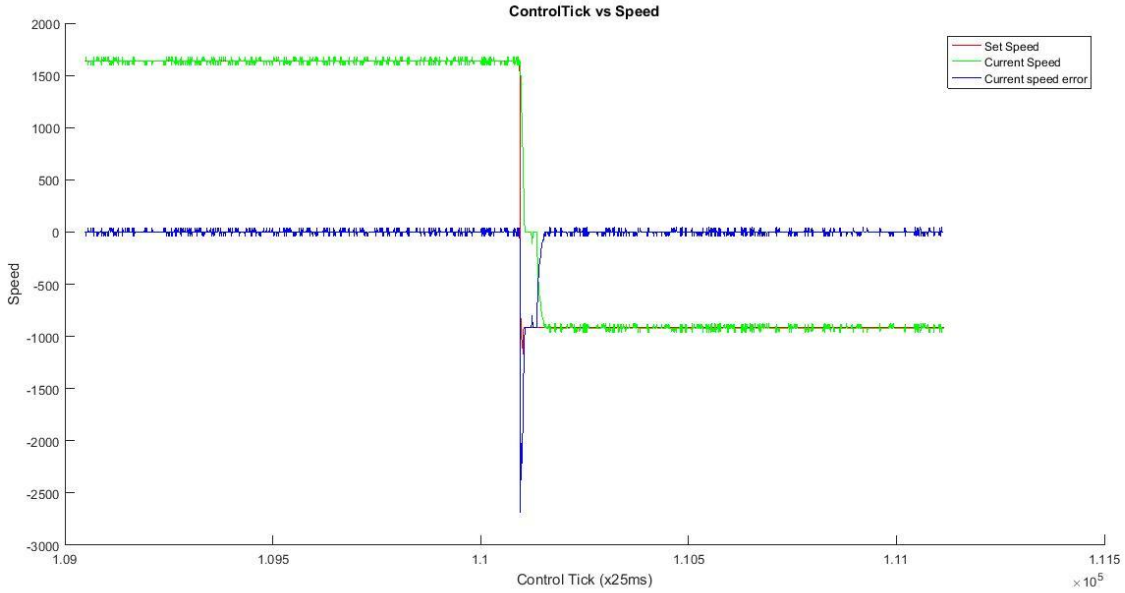


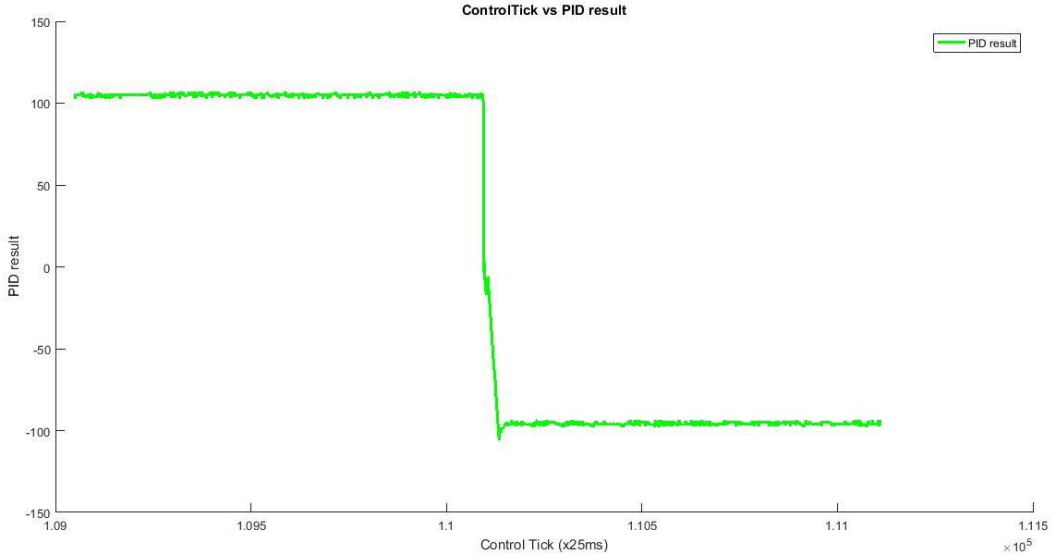
3. *Speed Increase in Same Direction*



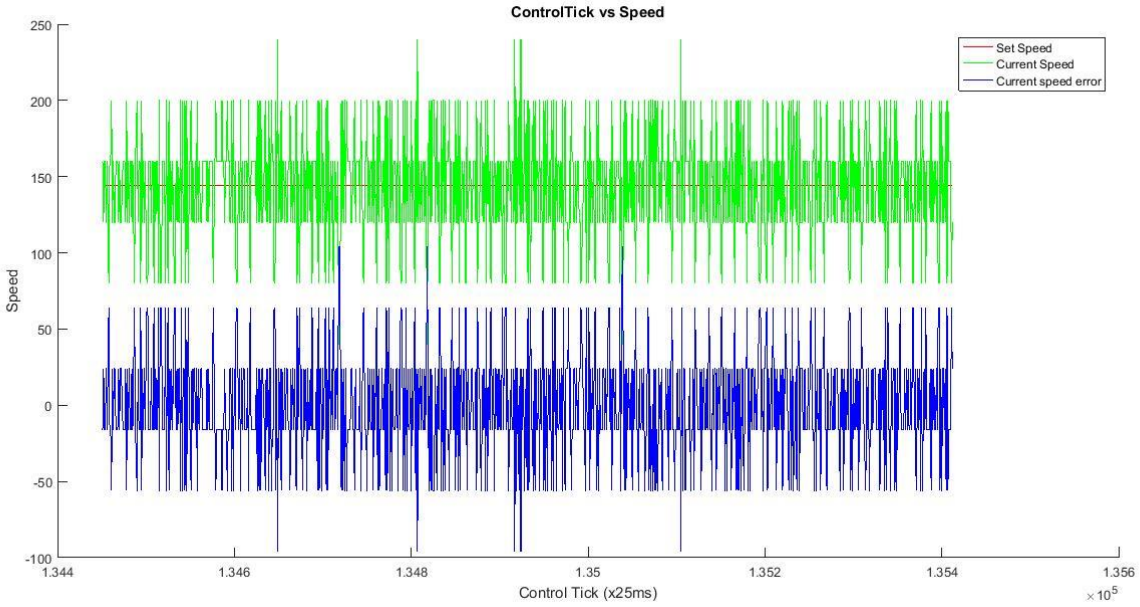


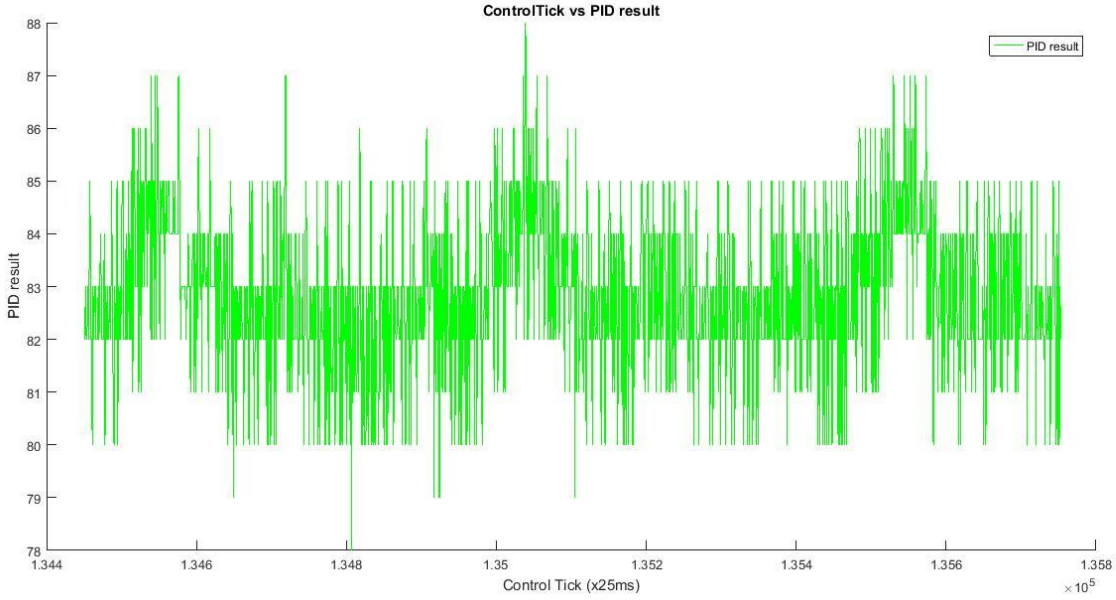
4. *Speed Change with Direction Change*





5. *Constant Speed without Load*





Discussion

Firstly, we used a motor bought from eBay, but the quadrature encoder did not work. Therefore, we had to buy another motor from Sri Lanka and conduct the experimentation again.

Further, as the motor we bought was not up to the required quality standards, we had to find a suitable motor from FAULHABER motors and apply the parameters accordingly.

Hence, the results and modelling done were not precise.

However, the experimentation can be considered as a success as we were able to control the speed of the motor remarkably.

References

Creative Commons Attribution-ShareAlike 4.0 International License. (n.d.). *Control Tutorials for MATLAB and SIMULINK.* Retrieved from DC Motor Speed: System Modelling: http://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=SystemModeling

DC Micromotors Series 2233 Datasheet. (n.d.). Retrieved from FAULHABER: https://www.faulhaber.com/fileadmin/Import/Media/EN_2233_S_DFF.pdf

DC-Micromotors. (n.d.). Retrieved from FAULHABER: https://www.faulhaber.com/en/products/series/2233s/

Sillitoe, P. I. (2017). *Notes: Introduction to Classical Control.*