**Michigan Technological University**

**EE 5750**

**Distributed Embedded Control Systems**

**Lab Experiment 05**

**Remote Electronic Throttle Control (ETC) via CAN**

**Submitted By:** Kirk D’Souza

**Instructor:** Dr. Bo Chen

**T.A.:** Ming Cheng

**Date Performed:** 3/26/2016

**Date Submitted:** 4/1/2016



1. **INTRODUCTION**

The purpose of this lab is to learn and practice CAN Communication using ECUs. The lab gives us a proper overview of setting up the parameters for the CAN communication. The main objective of the lab is to use a PID controller in ECU-1 to control the ETC throttle body opening via CAN communication with ECU-2

1. **SYSTEM INPUT/OUTPUT AND CONTROL LOGIC**

The system setup consists of a Bosch D V-E5 throttle body for ETC systems and 2 Woodward ECM 565-128. The connections for the same are as given below:

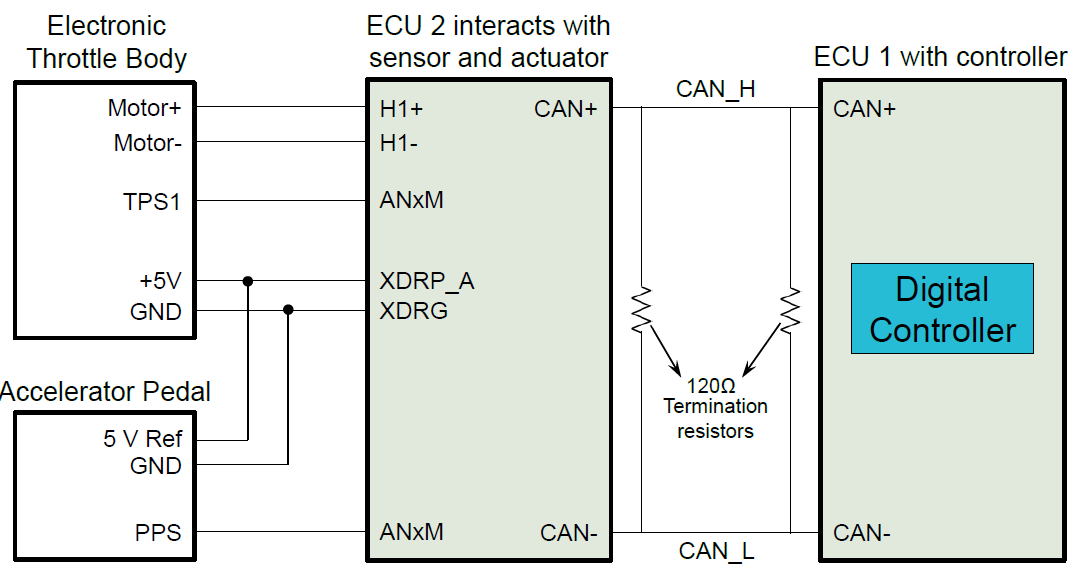


Figure 1: Circuit[1]

Here, one ECU contains the controller model and the other ECU is used to convert the Duty cycle to PWM to send to the DC servo Motor. IT is also used to obtain the feedback for the APP and TPS positions and send the data to the Controller ECU.

ECU-1 is the Controller ECU and ECU-2 is the Sensor and Actuator ECU.

The throttle body has a servo motor which is driven through a H-bridge in the ECM.

The TPS feedback which gives our error is taken from a block in the model.

The inputs for our model are the PPS Offset, PPS Gain, TPS Offset, Kp, Kd, and Ki

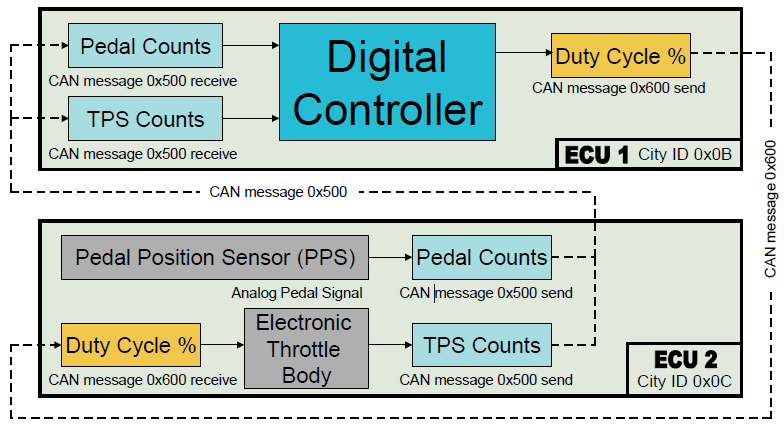


Figure 2:Control Logic[1]

The above control logic diagram can be explained separately as given below:

**Control Logic for ECU-1:**

Receive message from ECU-2, Convert to double format and scale to a range of 0 to 100

The table below shows the data for CAN Communication definition for ECU-1

|  |  |
| --- | --- |
|  | ECU-1 |
| Baud Rate | 250000 |
| Bus Channel | CAN\_1 |
| Message ID | Send-600 |
| Data Length | 4 bytes |
| Repeating Rate | 10ms |
| Message | Duty\_Cycle |

|  |  |
| --- | --- |
|  | ECU-2 |
| Baud Rate | 250000 |
| Bus Channel | CAN\_1 |
| Message ID | Read-500 |
| Data Length | 10 bits |
| Repeating Rate | 10ms |
| Message | Etc\_can\_500 |

The table below shows the data for CAN Communication definition for ECU-1

|  |  |
| --- | --- |
|  | ECU-1 |
| Baud Rate | 250000 |
| Bus Channel | CAN\_1 |
| Message ID | Send-600 |
| Data Length | 4 bytes |
| Repeating Rate | 10ms |
| Message | Duty\_Cycle |

|  |  |
| --- | --- |
|  | ECU-2 |
| Baud Rate | 250000 |
| Bus Channel | CAN\_1 |
| Message ID | Read-500 |
| Data Length | 10 bits |
| Repeating Rate | 10ms |
| Message | Etc\_can\_500 |

The table below shows the data for CAN Communication definition for ECU-1

|  |  |
| --- | --- |
|  | ECU-1 |
| Baud Rate | 250000 |
| Bus Channel | CAN\_1 |
| Message ID | Send-600 |
| Data Length | 4 bytes |
| Repeating Rate | 10ms |
| Message | Duty\_Cycle |

|  |  |
| --- | --- |
|  | ECU-2 |
| Baud Rate | 250000 |
| Bus Channel | CAN\_1 |
| Message ID | Read-500 |
| Data Length | 10 bits |
| Repeating Rate | 10ms |
| Message | Etc\_can\_500 |

to 100

Calibration of APP and TPS gains and offset

PID Controller tuning for setting Duty Cycle

Send Duty cycle to ECU-2

**Control Logic for ECU-2:**

Receive Duty cycle from ECU-1

Convert to PWM signal and send to Servo motor

Obtain feedback from APP and TPS positions of servo motor

Send APP and TPS position values to ECU-1

1. **MODEL DEVELOPMENT**

The model is built in Simulink, Matlab using the Simulink and Motohawk blocks.

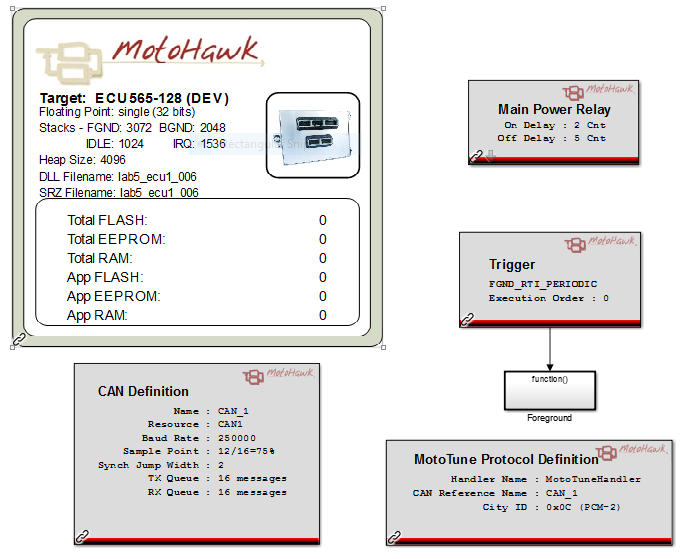
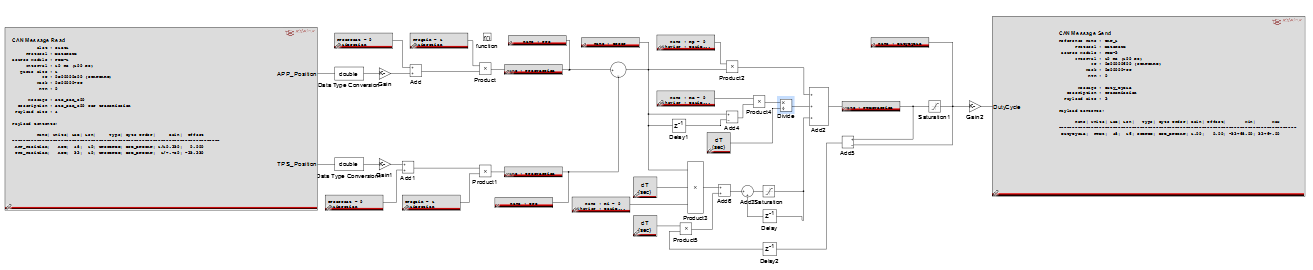
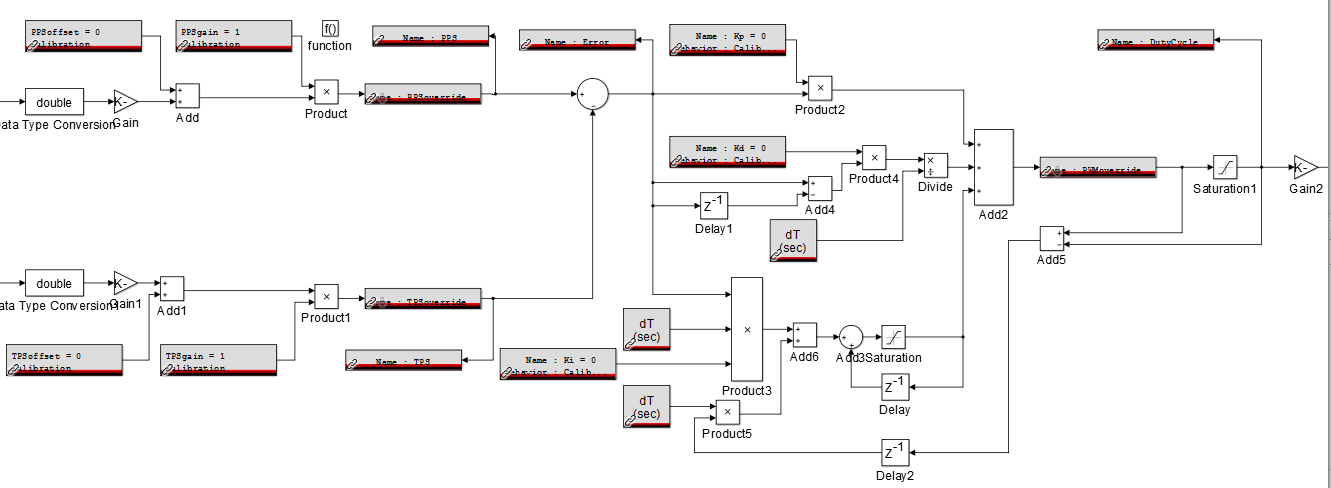


Figure3.1: Main Model

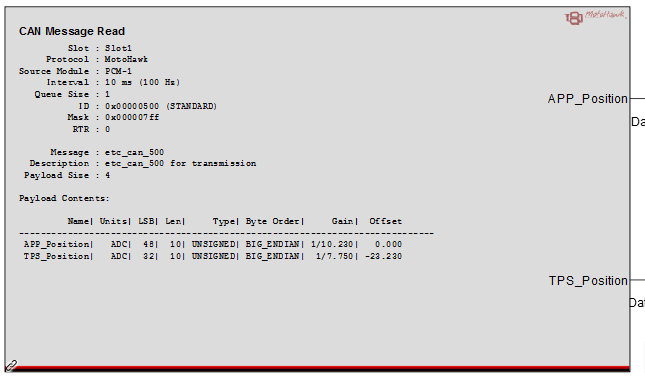
The Foreground model is given below:



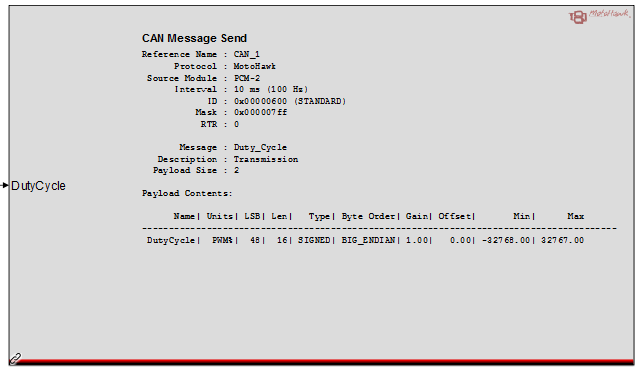


The above blocks is from the model used for the previous lab.

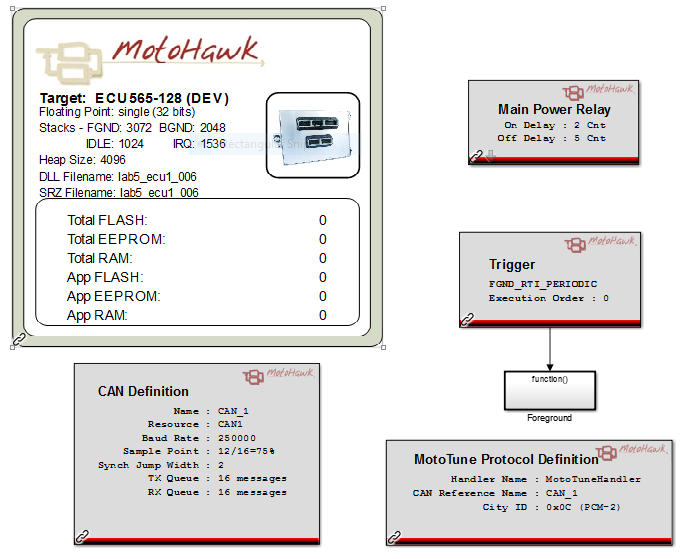
The blocks on the right are used to set the duty cycle



The parameters for the CAN Message Read block are calibrated in the etc\_can\_500 file in the MATLab editor window



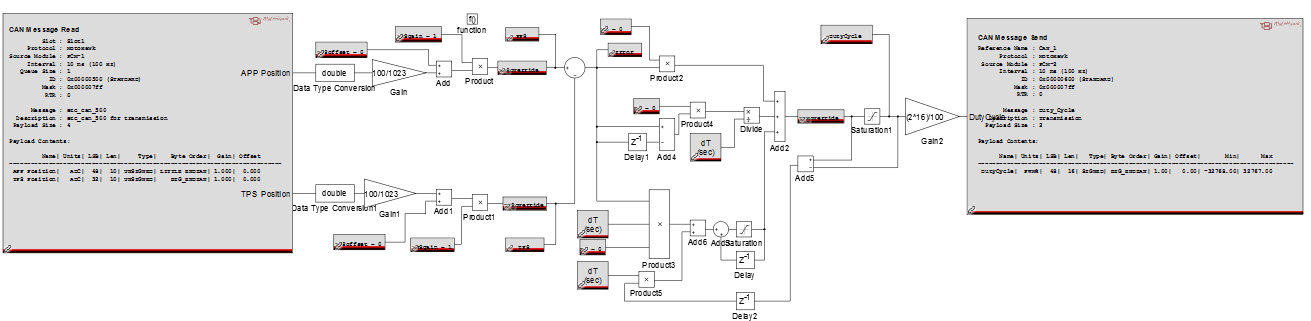
The main model calibrating the foreground model with the ECU is given below:



**ECU1 Method 1:**

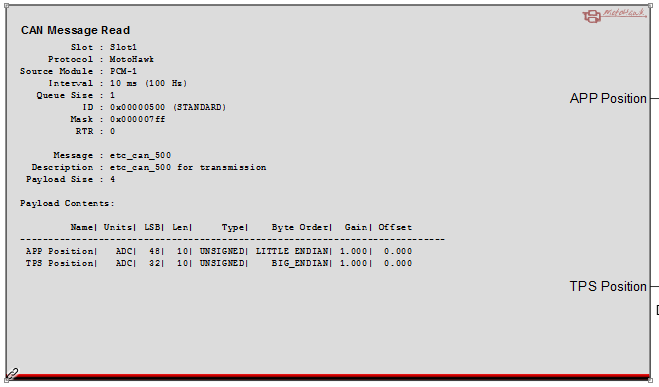
ECU1 Foreground:

The Main model for the Controller/Transmitting ECU is given below:



The above model can be divided into 4 major parts which are explained below:

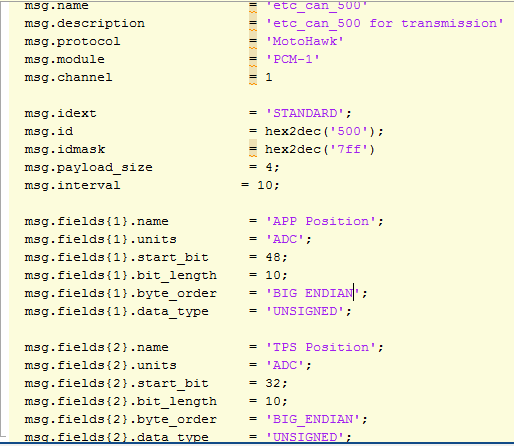
1. ECU 1 CAN Method 1, CAN Message Read:



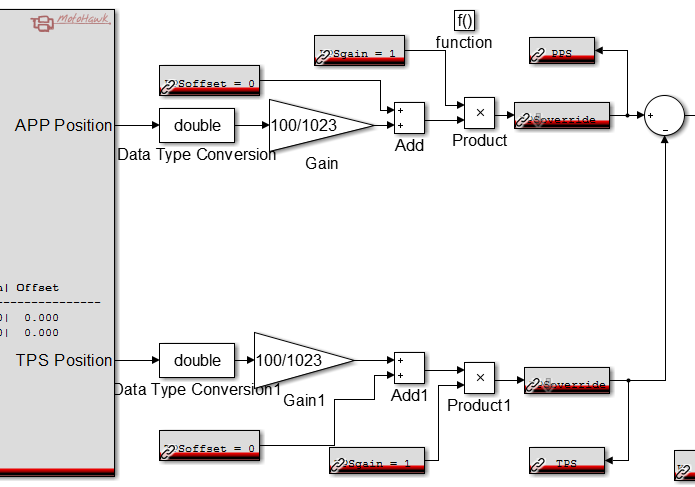
This block gives the configuration of CAN as set in the MATLab file. The Desktop simulator being used to read the transmission is PCM-1, therefore the source module is set to PCM-1. As evident from the above figure the repeating rate is set to 10ms, i.e. the message is transmitted every 10ms, the message mask with 07ff, the payload size is 4 bytes.

Here, the default gain and offset for the APP and TPS positions are set to 1 and 0 respectively. And the byte order is set to “Big Endian”

The configuration for the above block is given below in the MATLab file:



1. ECU 1 Method 1, calibration Blocks:

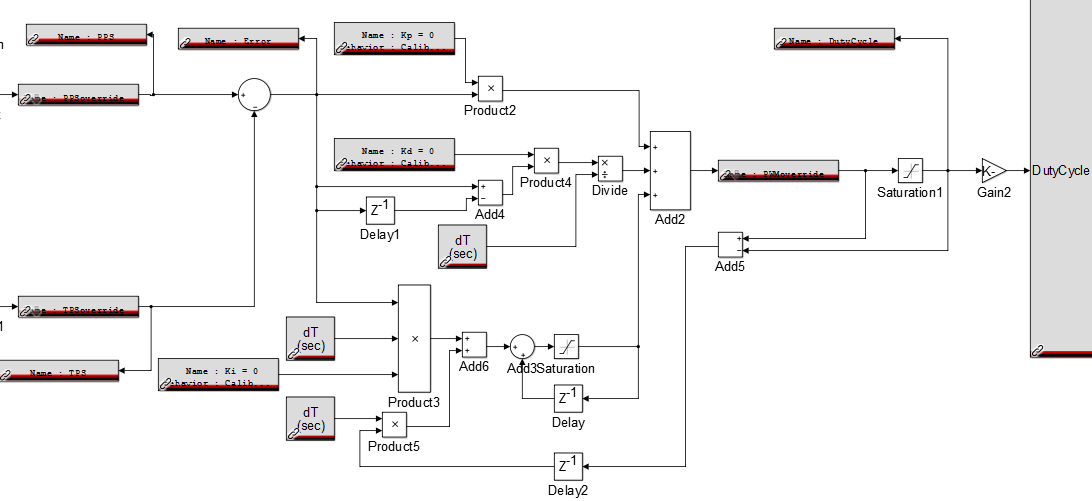


The above blocks are used for calibrating the PPS and TPS values. First, after converting the APP and TPS position values into the double format, they are then scaled to a range of 0 to 100 as that provides better resolution.

The Calibration blocks are used to set the values for APP and TPS gain and offsets in Mototune.

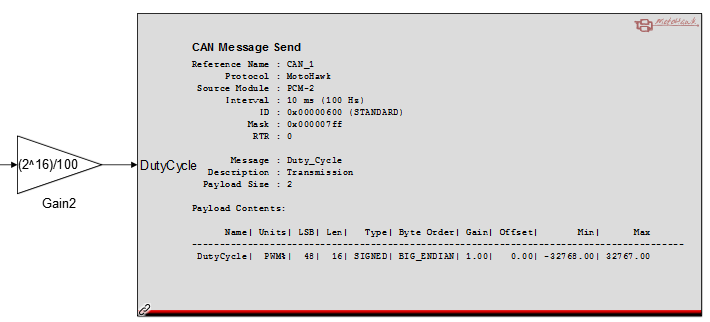
The Override blocks are used to fix a certain value for APP or TPS. The sum of the signals is further tuned with a PID controller.

1. ECU 1 Method 1, PID Tuning Blocks:



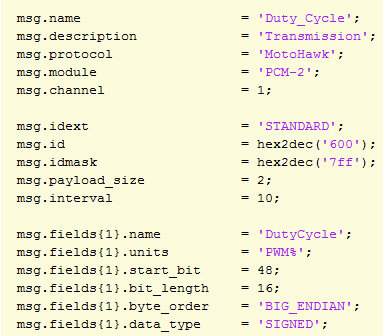
The above model shows the PID controller with an anti-wind-up for tuning the response of the system in Mototune. This is the same model that was developed in Lab 4.

1. ECU 1 Method 1, CAN Message Send:



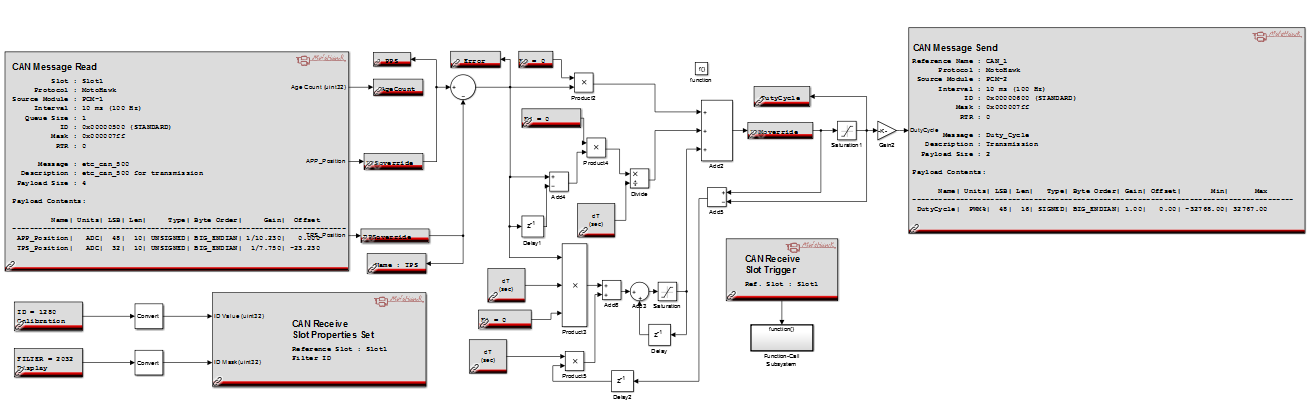
The above block is used to transmit the Duty Cycle to ECU-2. The tuned response is first expanded to 16 bits so that no data is lost in the transmission.

The configuration for the above block is given below in the MATLab file:



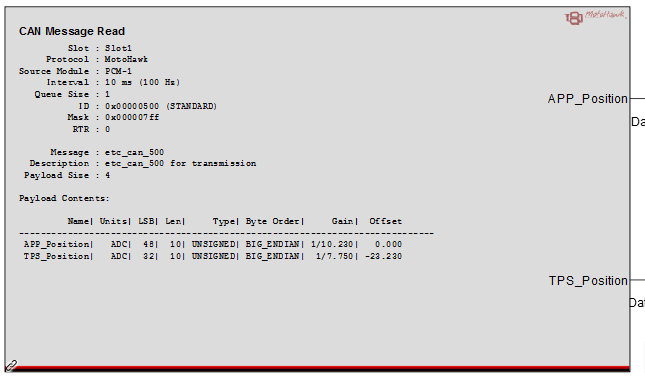
**ECU 1 Method 2:**

Foreground model:

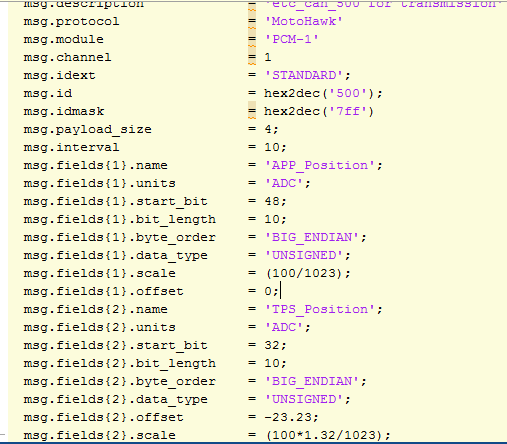


Here, the APP and TPS is calibrated within the CAN Message definition blocks itself, therefore the calibration blocks for the APP and TPS gains and Offsets are removed. The rest of the model is the same

1. CAN Message Read with Scale and Offset for PCM-1:



The configuration for the above block is given below in the MATLab file:

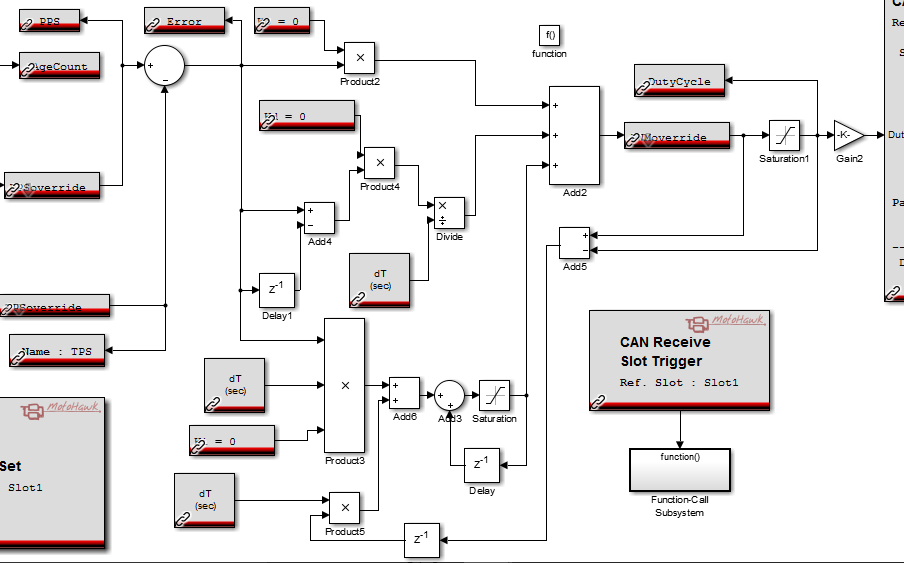


Here, the PPS was first set to 0 using the duty cycle override, and the error obtained for the same is taken as the offset.

For setting the scale the PPS is first taken as maximum (35), and the scale value is taken as (100/max TPS)

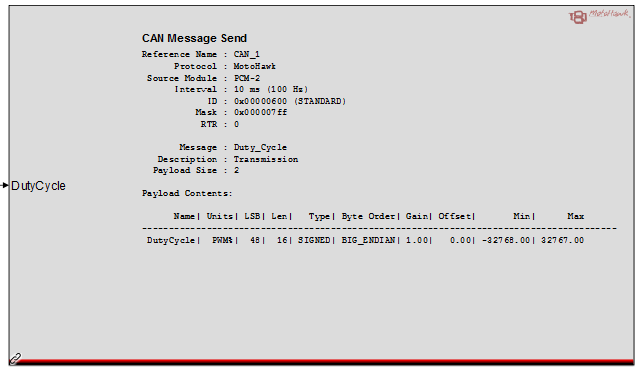
Here, the offset obtained is -23.23 and the scale is (100\*1.32/1023)

1. APP, TPS, and PID Tuning Blocks:

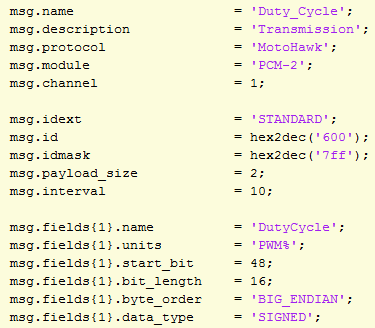


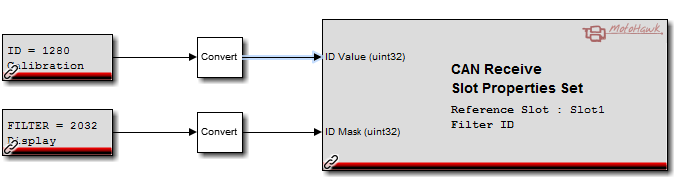
Here, since the PPS and TPS values are calibrated in the definition files itself, the calibration blocks were removed. The rest of the logic remains the same.

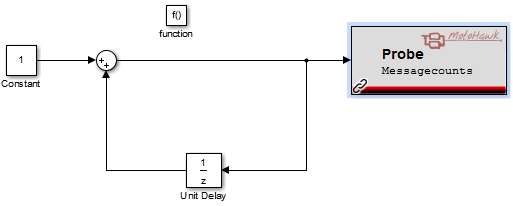
1. CAN Message Send with Scale and Offset for PCM -2:



The configuration for the above block is given below in the MATLab file:



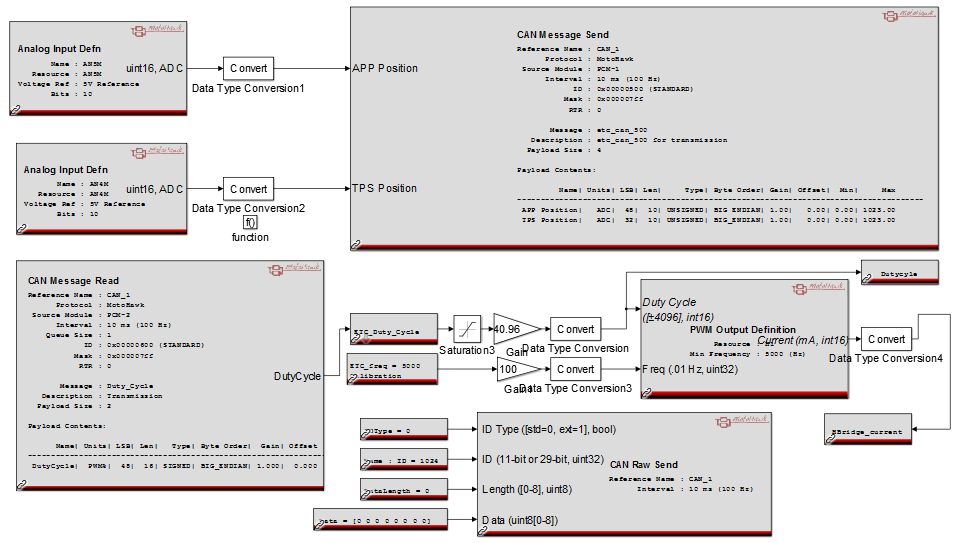




The above blocks are used to check the message counts and ID value for validation

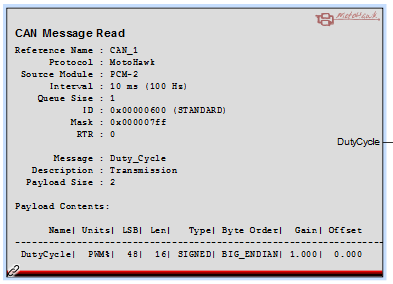
**ECU2-Sensors and Actuators:**

Foreground Model:



The above foreground model is for the sensor and Actuator model on the 2nd desktop simulator. This model operates the ETC body with the duty cycle values that were transmitted from ECU-1.

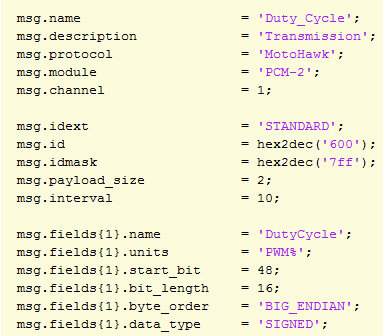
1. CAN Message Read for ECU 2 for PCM-2:



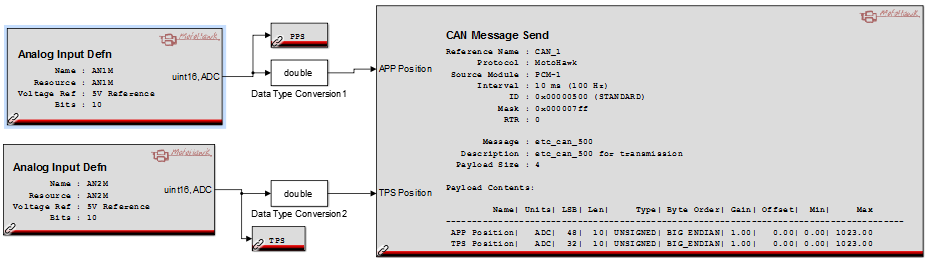
This block gives the configuration of CAN as set in the MATLab file. The Desktop simulator being used to read the transmission is PCM-2, therefore the source module is set to PCM-2. As evident from the above figure the repeating rate is set to 10ms, i.e. the message is transmitted every 10ms, the message mask with 07ff, the payload size is 4 bytes.

Here, the default gain and offset for the APP and TPS positions are set to 1 and 0 respectively. And the byte order is set to “Big Endian”

The configuration for the above block is given below in the MATLab file:

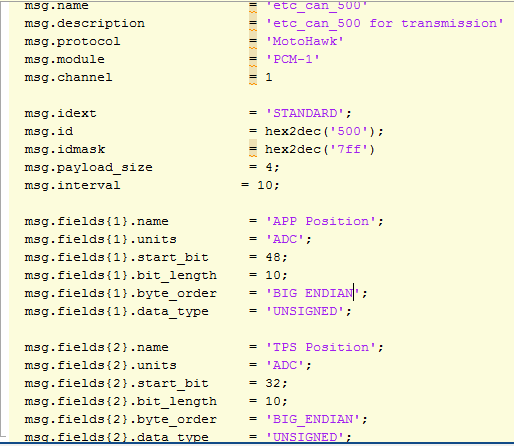


1. CAN Message Send for ECU 2 for PCM-1:



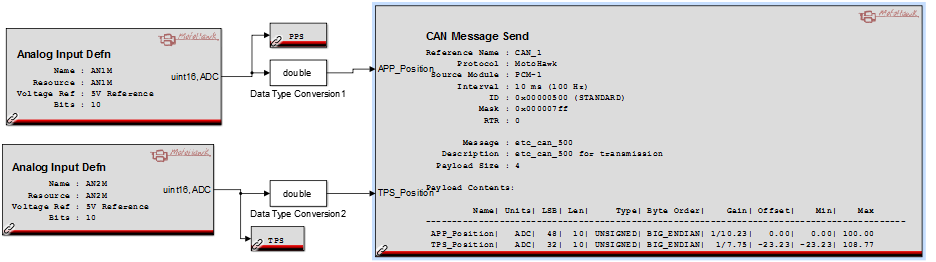
The analog input definition blocks take the APP and TPS feedback values

The configuration for the above CAN Message Send block is given below in the MATLab file:

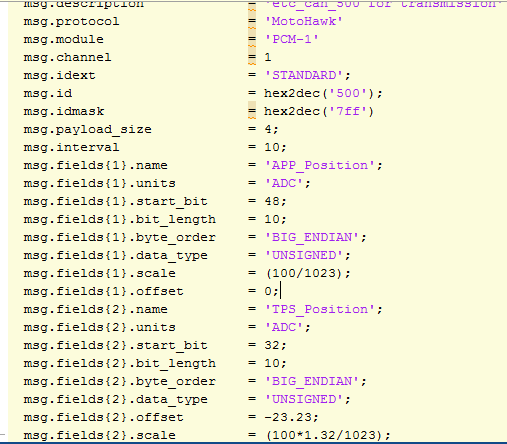


**Method 2:**

1. CAN Message Send:

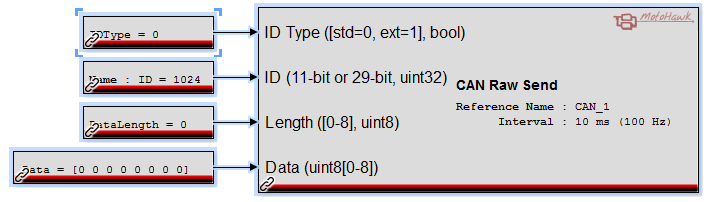


The configuration for the above block is given below in the MATLab file:



The above configuration parameters are set for method 2 where the offset and scale are preset in the MATLab file as discussed above

1. CAN Raw Send:



The above block is used to check basic CAN communication parameters

1. **SIMULATION AND CALIBRATION RESULTS**

**Calibration for PPS and TPS:**

The PPS and TPS have to be in the same range so that the measured error value can be accurately obtained. Here, the DC servo motor in the Throttle body requires a duty cycle of 40% for the throttle valve to be at the maximum open position and 0% to be in a completely closed position.

To set the PPS and TPS values accordingly, a PPS gain of 1 and a TPS gain of 1.32 and TPS Offset of -17.6 is taken

**Calibrated Controller**

Taking inputs as

Kp=5.85

Ki=1.3

Kd=0.065

PPS Gain = 1

PPS Offset = 0

TPS Gain = 1.32

TPS Offset = -17.6

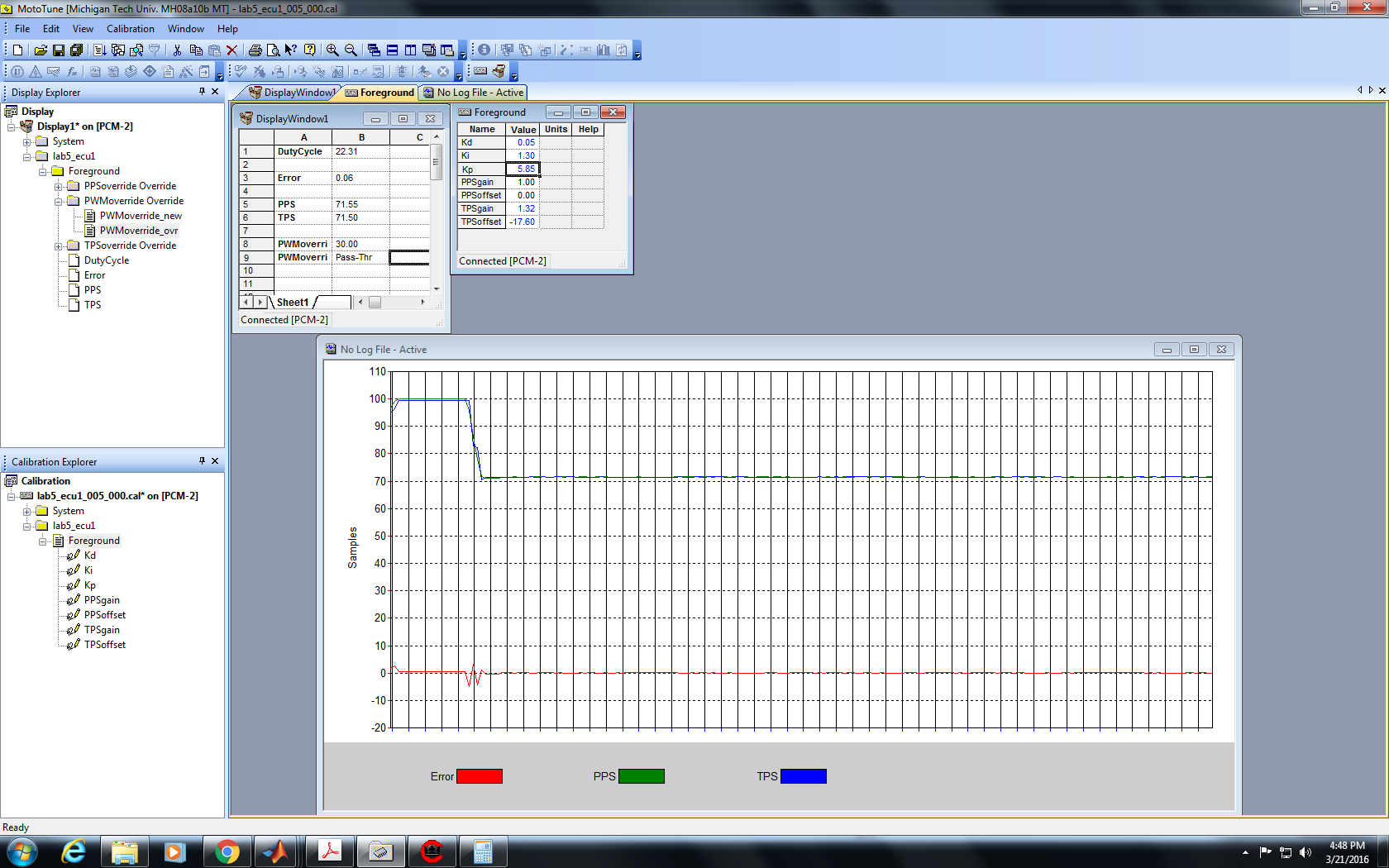


Figure 4:1

Transcient Response Calibration

Kp = 5.85

Ki = 1.3

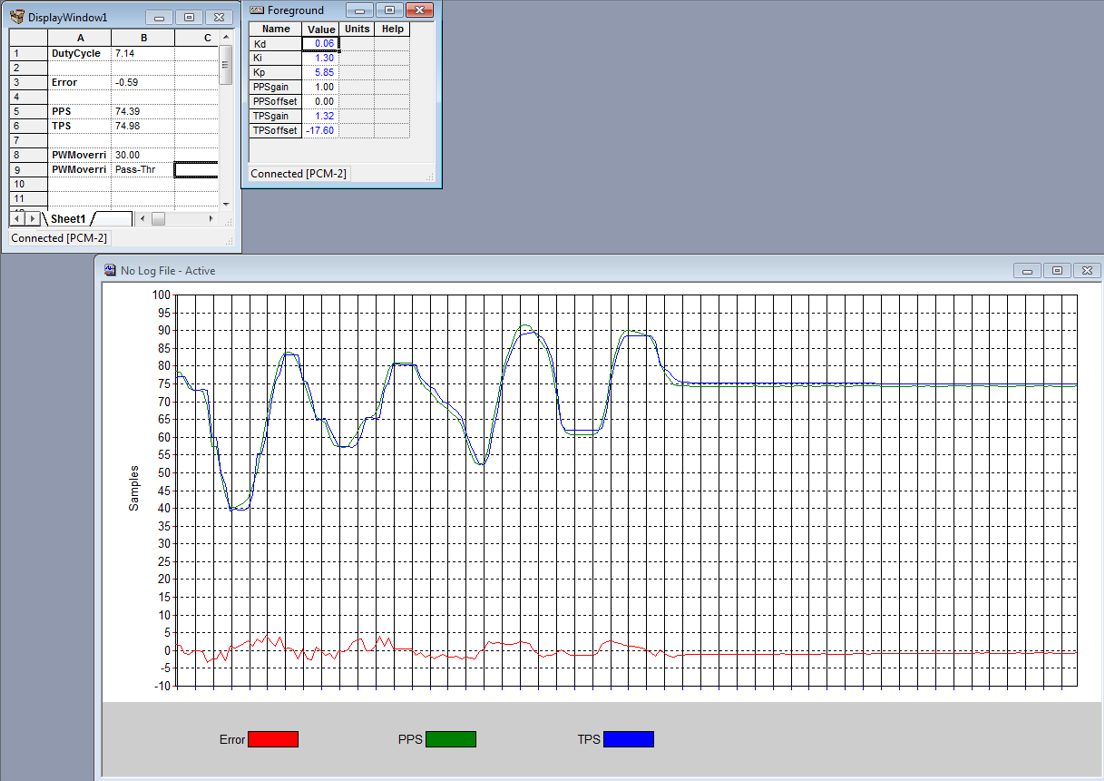
Kd = 0.06

PPS Gain = 1

PPS Offset = 0

TPS Gain = 1.32

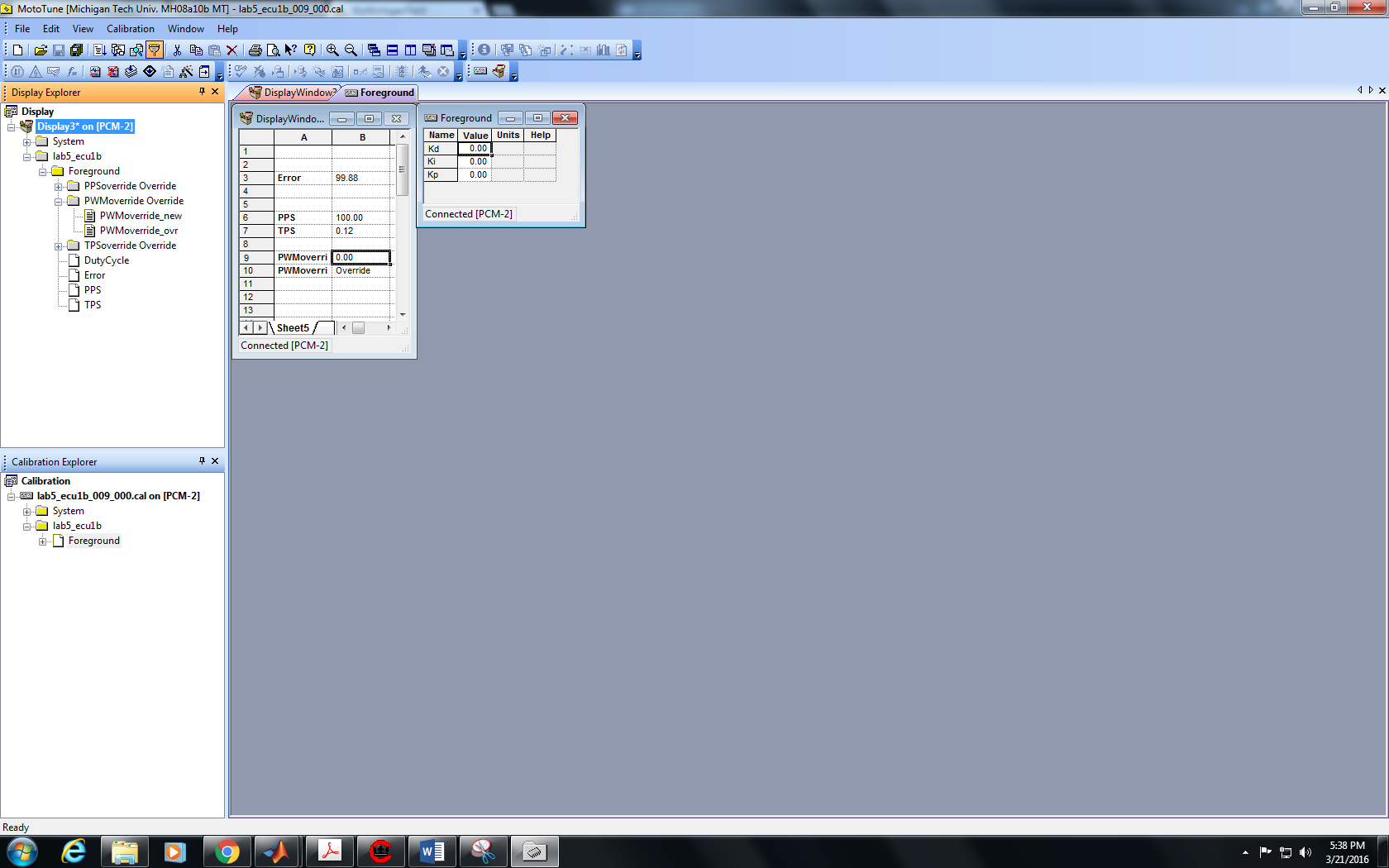
TPS Offset = -17.6



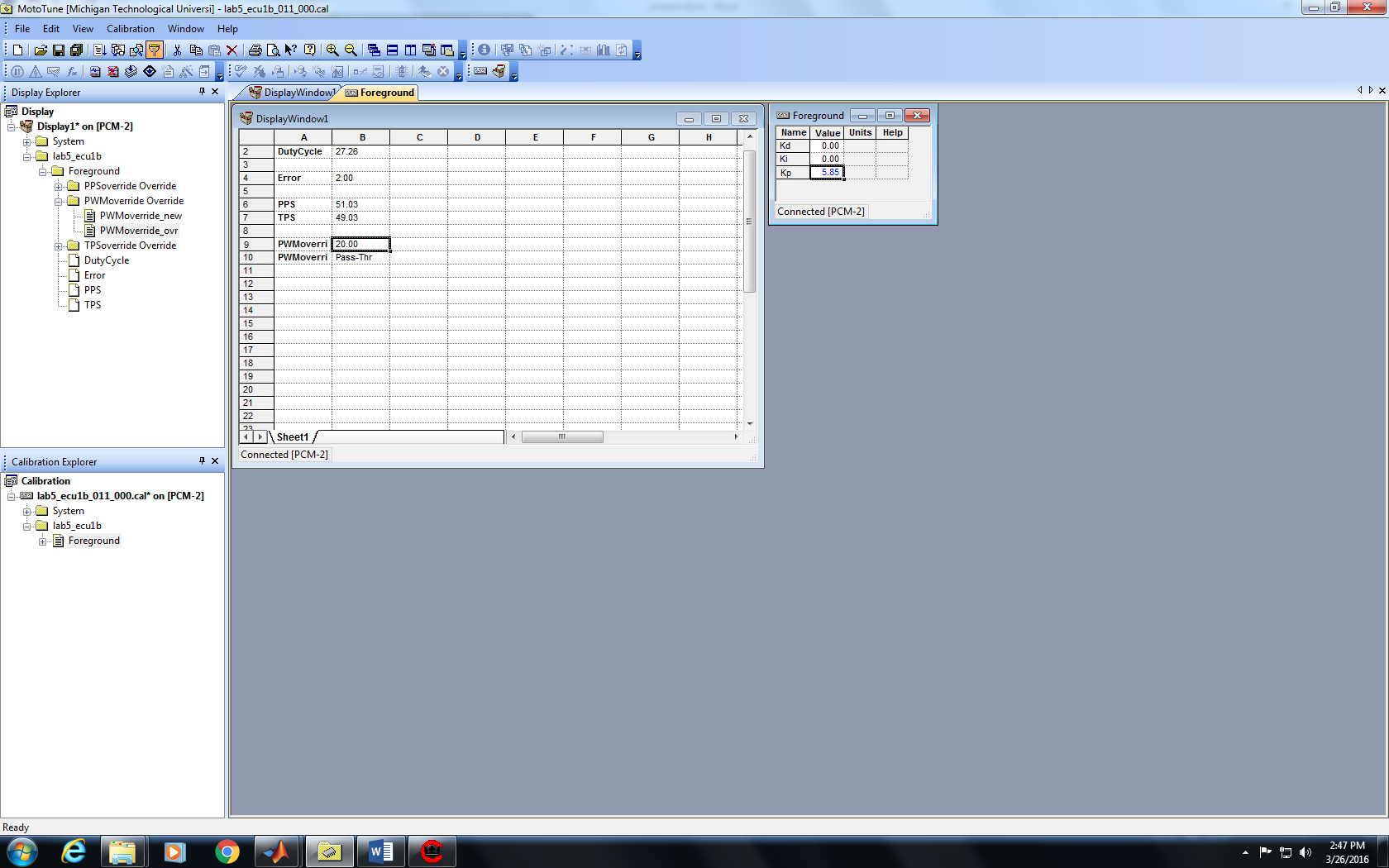
**Method 2:**

With Scale and offset

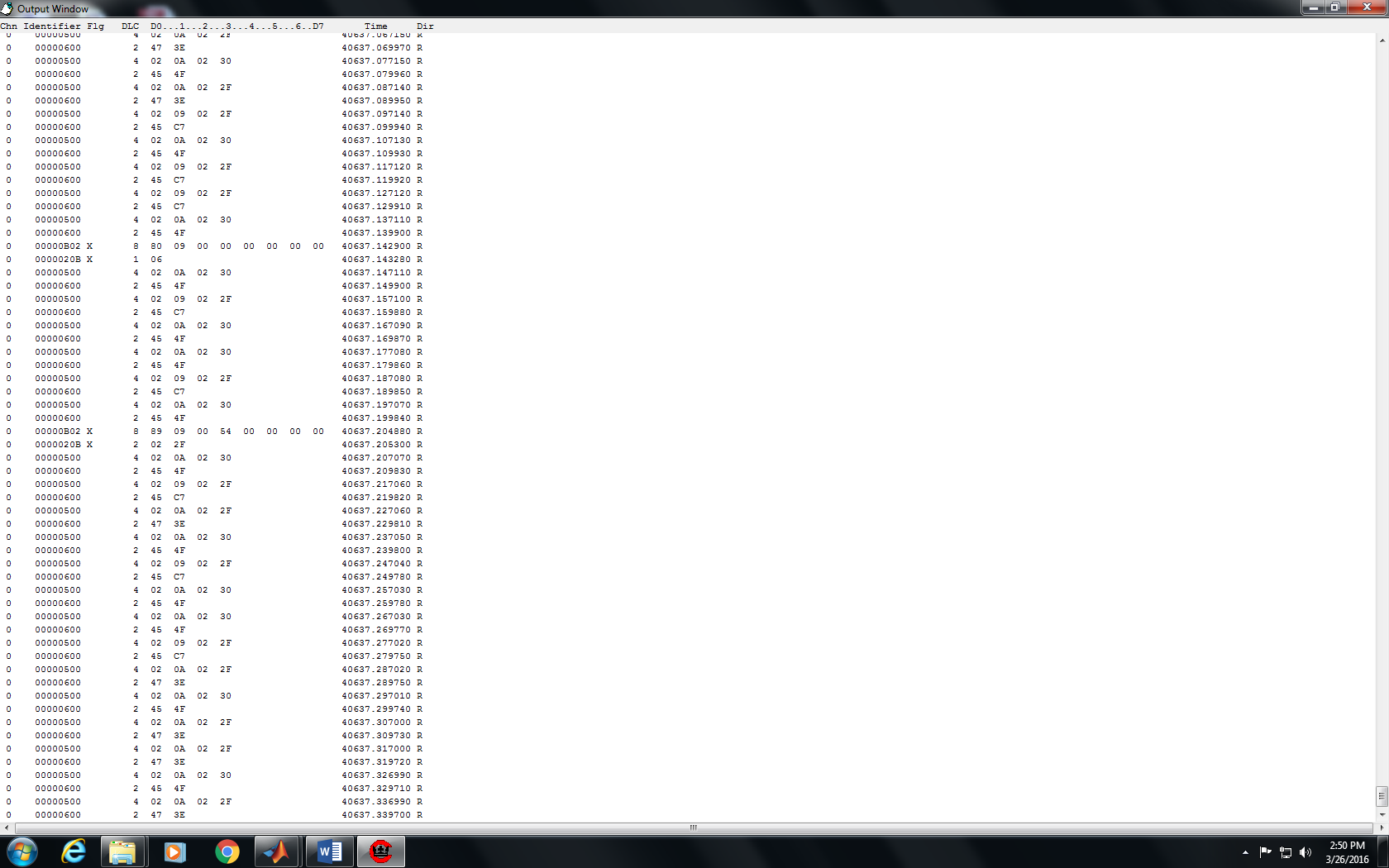
Here, the scale and offset were preset in the m file that is used to configure the CAN Send and Receive blocks



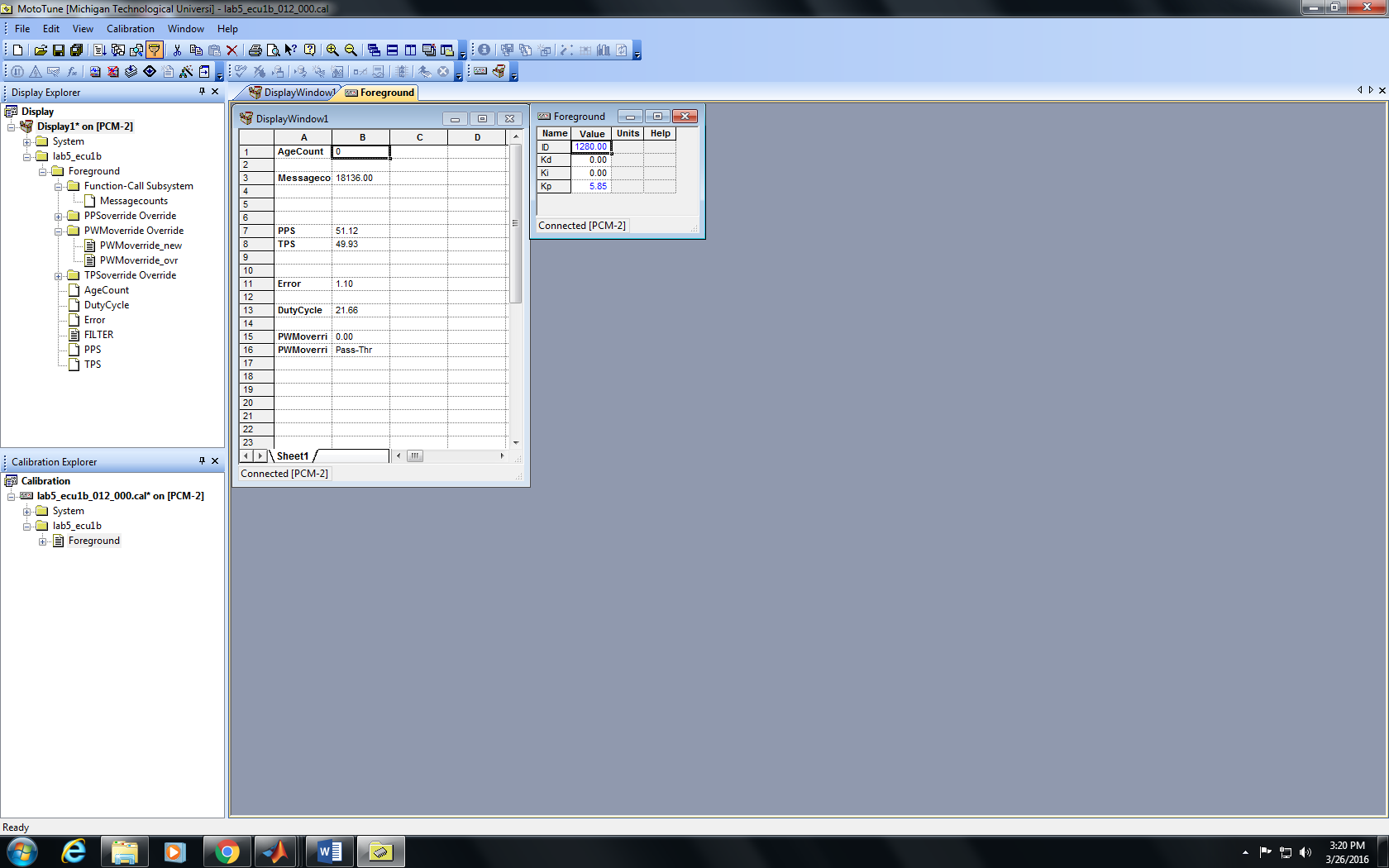
CAN King

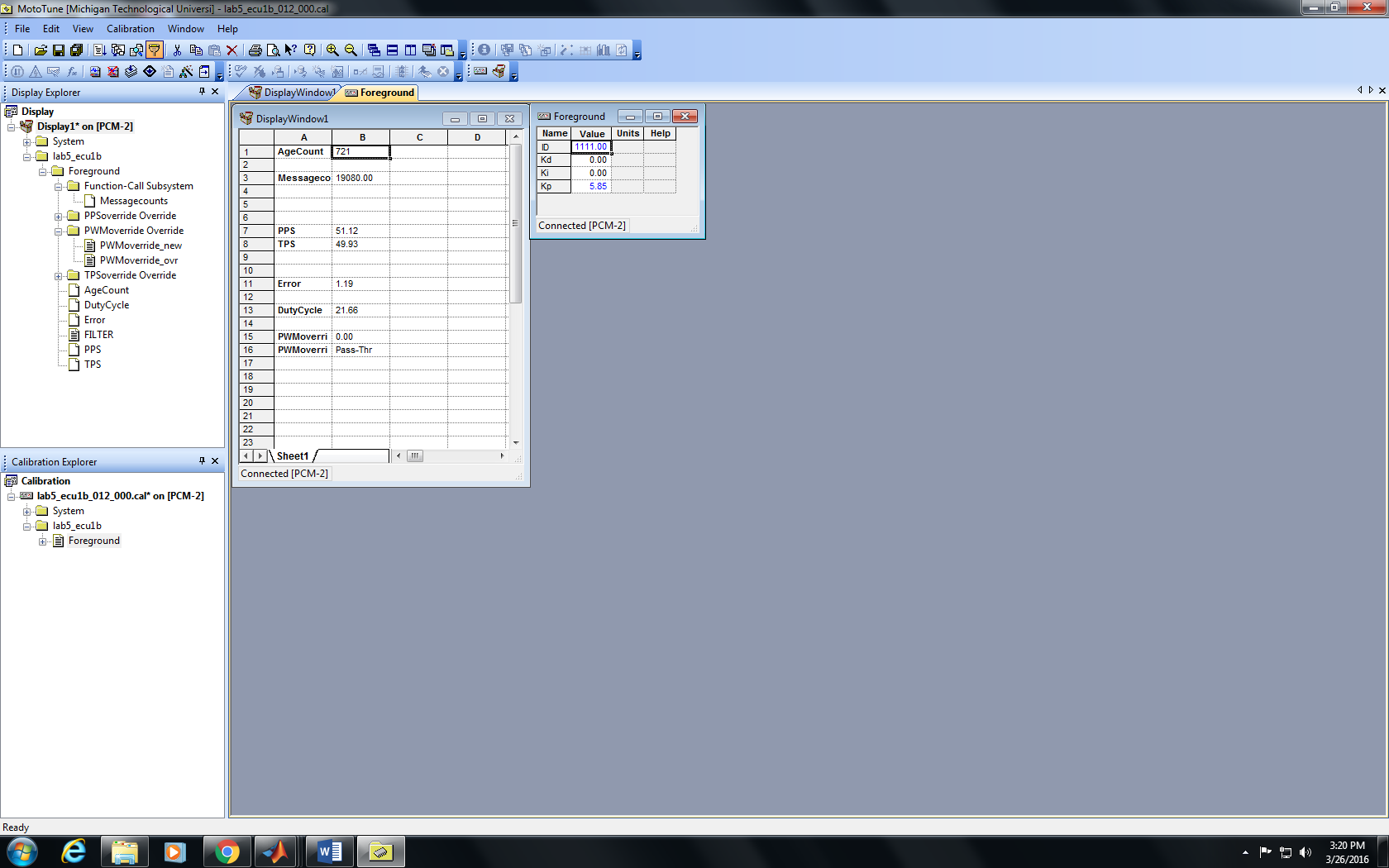


CAN King Output Window:



When the ID value is set to 1280, the Age Count is 1





1. **VALIDATION**

**For Tuned PID Controller:**

It can be seen from figure 4.1 that the TPS values for the PID controller is very close to the PPS values and the steady state error is negligible. Therefore, the values set for the Proportional, Integral and Derivative gains are optimum

Kp = 5.85

Ki = 1.3

Kd = 0.06

PPS Gain = 1

PPS Offset = 0

TPS Gain = 1.32

TPS Offset = -17.6

It can be seen from Figure 4.2 that the feedback response is nearly the same as the Accelerator Pedal Position and the error is less than 5 for most of the changes in the system inputs.

**CANKing**:

From figure 4.5, we can see that the value for PPS is 02 09 which is in hexadecimal,

Converting this to binary we get 0000 0010 0000 1001

Converting it to decimal we get 521

Multiplying it with the scale factor for PPS we get 50.92

This value is close to the PPS value of 51.03 shown in figure 4.4

Hence, we can say that the scaled model works as desired.

**ID Filter:**

From figure 4.6 and 4.7, we can see that when the ID filter is set to 1280, the age count is 0, but when the ID value is changed, the age count increases

The ID filter is set according to the message ID which is 0x500 or 1280(in decimal)

If the ID is set to any other value, then there is no communication between the ECUs and the age count increases.

The graph below is from a log file. When the Age Count starts to increase, the ID was changed from 1280 to 1111 as can be seen from figure 4.6 and 4.7

1. **DISCUSSION**

The table below shows the data for CAN Communication definition for ECU-1

|  |  |
| --- | --- |
|  | ECU-1 |
| Baud Rate | 250000 |
| Bus Channel | CAN\_1 |
| Message ID | 0x600 |
| Data Length | 4 bytes |
| Repeating Rate | 10ms |
| Message | Duty\_Cycle |

The table below show the data for CAN communication definition for ECU-2

|  |  |
| --- | --- |
|  | ECU-2 |
| Baud Rate | 250000 |
| Bus Channel | CAN\_1 |
| Message ID | 0x500 |
| Data Length | 10 bits |
| Repeating Rate | 10ms |
| Message | Etc\_can\_500 |

The PID Controller with Anti –Wind up is the same as the one used in lab 4

In Lab 4, the optimum proportional, integral and derivative gains were

Kp = 13

Ki = 0.1

Kd = 0.6

For Lab 5, the optimum proportional, integral, and derivative gains are:

Kp = 5.85

Ki = 1.3

Kd = 0.06

The proportional gain is less than half because the feedback is slow as the CAN communication interval is 10ms. Therefore, the model takes more time to update the response.

1. **CONCLUSION**

The Lab helped us get a clear understanding of configuring CAN transmission between 2 ECUs and using CANKing to validate the data being transferred. CAN communication is an important

1. **REFERENCE**
2. EE 5750 Lecture and Lab notes by Dr. Bo Chen
3. Bosch D V-E 5 Throttle body for ETC Datasheet
4. Woodward ECM-0565-128-0701-C Engine Control Module (Part No. 237-1238)