MBSD Lab #4 A.Y. 2022/23

Purposes

- Integrate the one pedal controller into a simulated Arduino Uno[1] microcontroller (μC), resorting to SimulIDE[2].
- Interact with the μC through its digital and analog interfaces.

Instructions

For instruction on how to use SimulIDE and the Platform Support Packages[3] follow the instruction provided by Prof. Violante in the lecture of Thursday 11th May 2023.

The delivery shall contains:

- The controller Simulink model used to generate the firmware binary file (plus all the accompanying files needed to make it possible to generate again the code, like .m files containing initializations)
- The firmware binary file to be loaded into the simulated Arduino in SimulIDE
- The SimulIDE project file.
- The PDF or Microsoft Word version of the report.

It is available an example based on a Tank level controller, in the folder

[1] Arduino Uno Board Anatomy https://docs.arduino.cc/tutorials/uno-rev3/BoardAnatomy, last visited on 16/05/2022.
[2] SimulIDE, https://www.simulide.com/p/home.html, last visited on 16/05/2022.
[3] Simulink Support Package for Arduino Hardware and MATLAB Support Package for Arduino.

Model-Based Software Design, A.Y. 2022/23

Laboratory 4 Report

Components of the working group (max 2 people)

- Alessandro Cavalli, 301494
- Cihan Yurtsever, 296824

I/O interfaces

[Please describe the I/O interfaces to interact with the one pedal controller trough the electrical interfaces of the Arduino]
Since the Arduino Uno does not provide a CAN BUS interface, we assumed that the input and output signals are represented through a suitable voltage level.

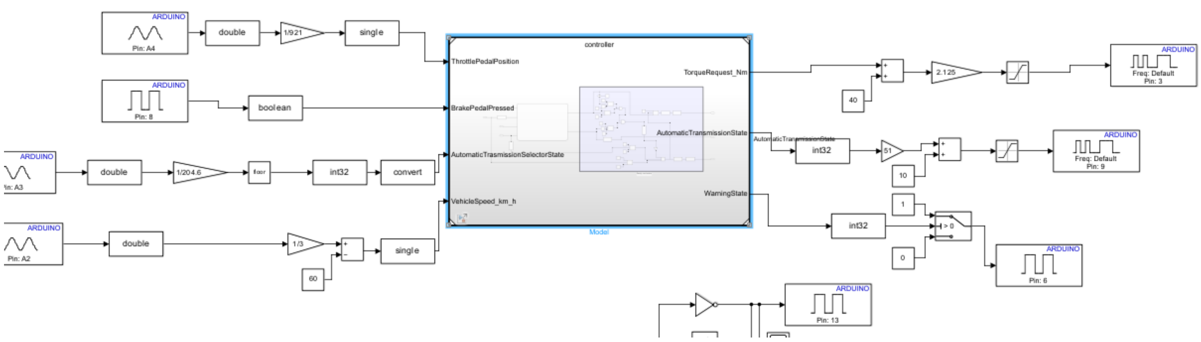

Figure 1. The harness of the One Pedal System

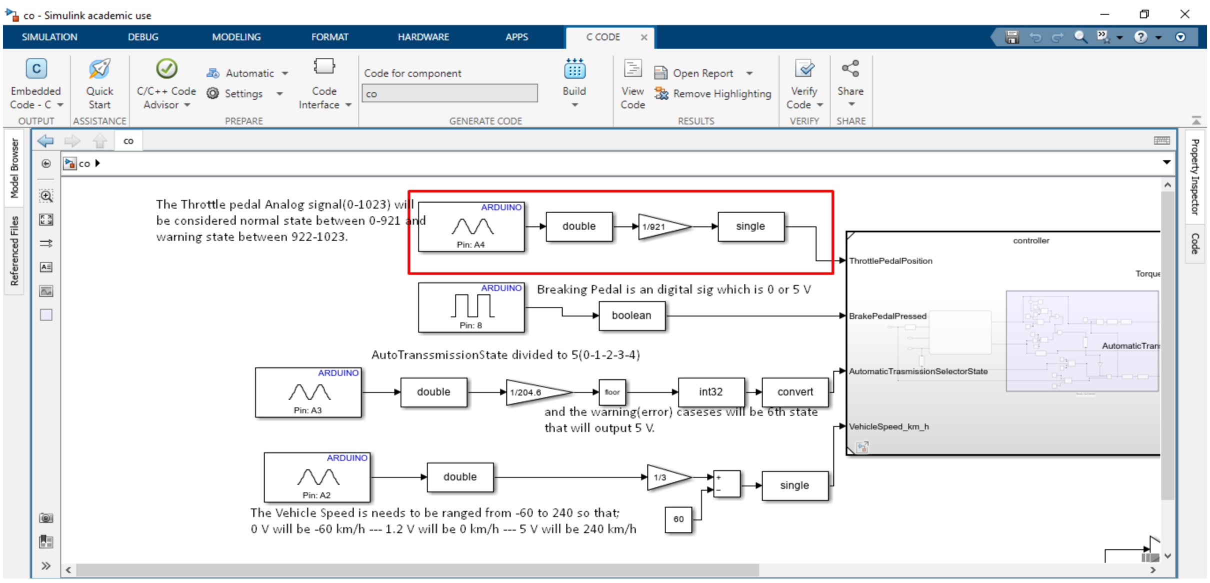

Figure 1.1 The Inputs of the Harness

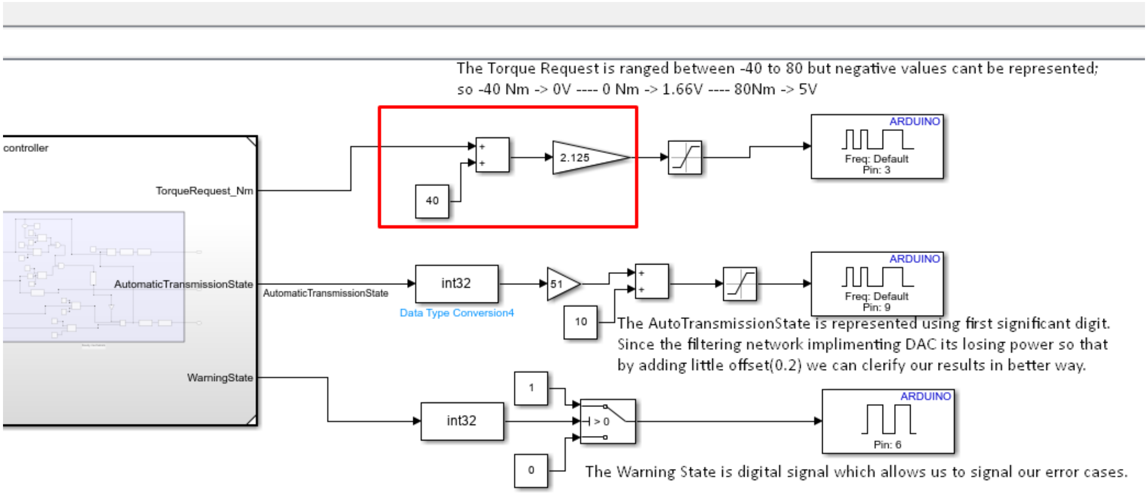

Figure 1.2 The Outputs of the Harness

Name	Unit	Type[4]	Conversion formulas	Min[5]	Max
ThrottlePedalPositionRaw (A4)	Volt/10bit	AI	A4/921	0	1023
BrakePedalPressedRaw (D8)	No	DI	No	0	1
AutomaticTransmissionSelectorStateRaw(A3)	Volt/10bit	AI	floor(A3/204.6)	0	1023
VehicleSpeedRaw(A2)	Volt/10bit	AI	A2/3 - 60	0	1023
TorqueRequestRaw(D3)	Volt/8bit	DO(PWM)	(TorqueRequest+40)*2.125	0	255
AutomaticTransmissionStateRaw(D9)	Volt/8bit	DO(PWM)	(State*51)+10	0	255
WarningStateRaw	No	DO	No	0	1

Notes:
ThrottlePedalPositionRaw is assumed to be an analog measurement acquired from a sensor which operates between 0V and 4.5V. If some error occurs, the output of such sensor will be greater than 4.5V. The conversion formula produces values greater than one in all the error cases so that the controller can detect the malfunction.
AutomaticTransmissionSelectorStateRaw is assumed to be an analog measurement acquired from a sensor wich operates between 0V and 5V. Each Volt corresponds to a state. 5V indicates a selector malfunction.

AutomaticTransmissionStateRaw is assumed to be fed into a Low pass filter network with losses. Therefore, in order to compensate the losses, an initial voltage bias of 10 Volt/8bit (0.2V) is applied.

[4] Digital Input (DI), Digital Output (DO), Analog Input (AI).
For AIs, provide the conversion formula from input voltage to the measurement unit data (indicating also how to perform the conversion from the raw reading of the ADC).
[5] The Min/Max values that can be handled due to the conversion formula shall be expressed in the measurement unit specified in the Column.

Code generation for Arduino

Step 1; Finishing all the charts make it ready to be generated..

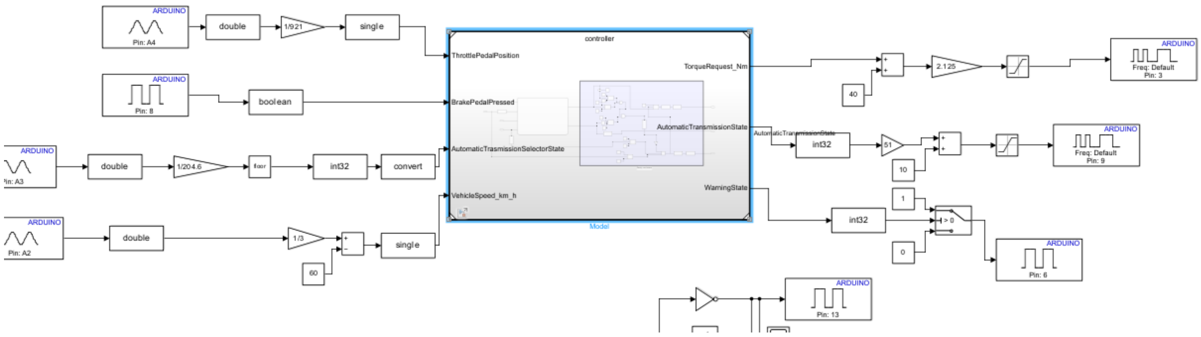

Figure 2. Completed Algorithm of the One Pedal System

Step 2; Entering on the controller and setting the hardware and code generation configuration parameters..

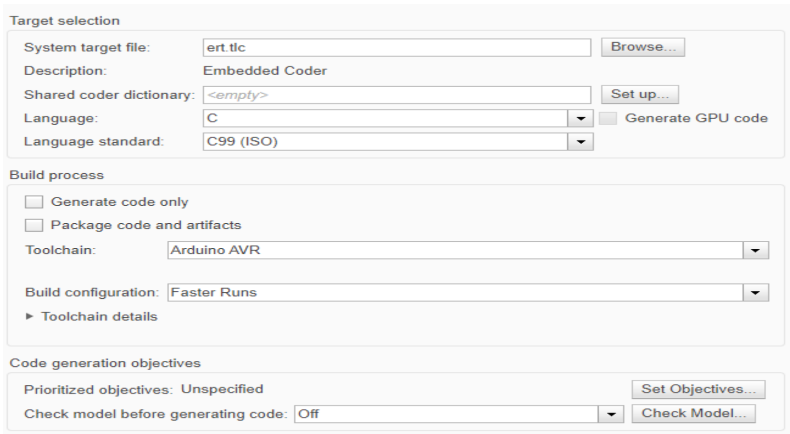

Figure 2.1 Code Generation parameters

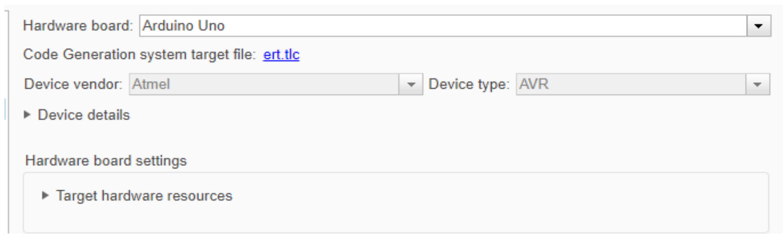

Figure 2.2 Hardware Implementation parameters

Step 3; Generating the code..

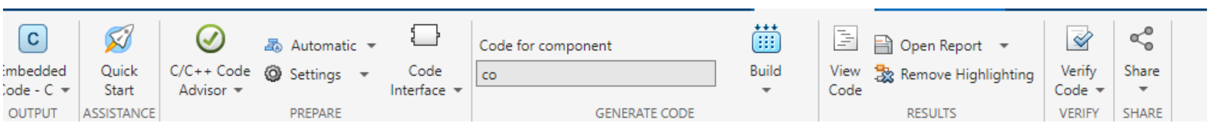

Figure 2.3. Embedded Coder Interface

Building the code from the Embedded Coder app and from the interface selecting the Build.

Harness

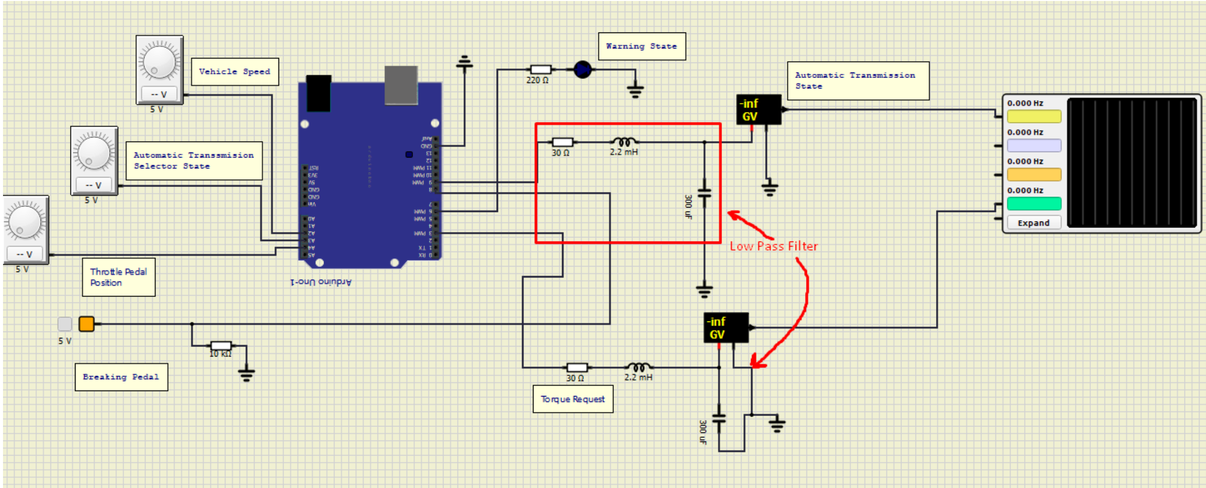

Figure 3. Harness of the Simullde Model of One Pedal Controller

Test stimuli

The test inputs will be formatted in the following way
(BrakePedalPressedRaw, ThrottlePedalPositionRaw,
AutomaticTransmissionSelectorStateRaw, VehicleSpeedRaw).
All the units will be expressed in volts.

Reverse fail for brake(0,0,1,0.9) => no action
Reverse (5,0,1,0.9) => ok
Reverse gas (0,0→4.5,1,0.9) => increasingly backward torque
Error gas (0,5,1,0.9) => warning led and swap to neutral Neutral (0,0,2,0.9) => ok
Drive fail for brake (0,0,3,0.9) => fail
Drive fail for speed (5,0,3,0) => fail
Drive (5,0,3,0.9) => ok
Brake fail for throttle (0,0,4,0.9) => fail
Brake (0,1.6,4,0.9) => ok
Brake gas (0,1.6→4.5,4,0.9) => increasingly forward torque
Brake regenerative (0,1.6→0,4,0.9) => increasingly backward torque
Error selector (0,1.6,5,0.9) => warning led and swap to neutral
Parked fail for speed(5,0,0,2.5) => fail
Parked fail for brake(0,0,0,0.8) => fail
Parked (5,0,0,0.8) => ok

Some screenshots has been added to show part of the testing procedure.

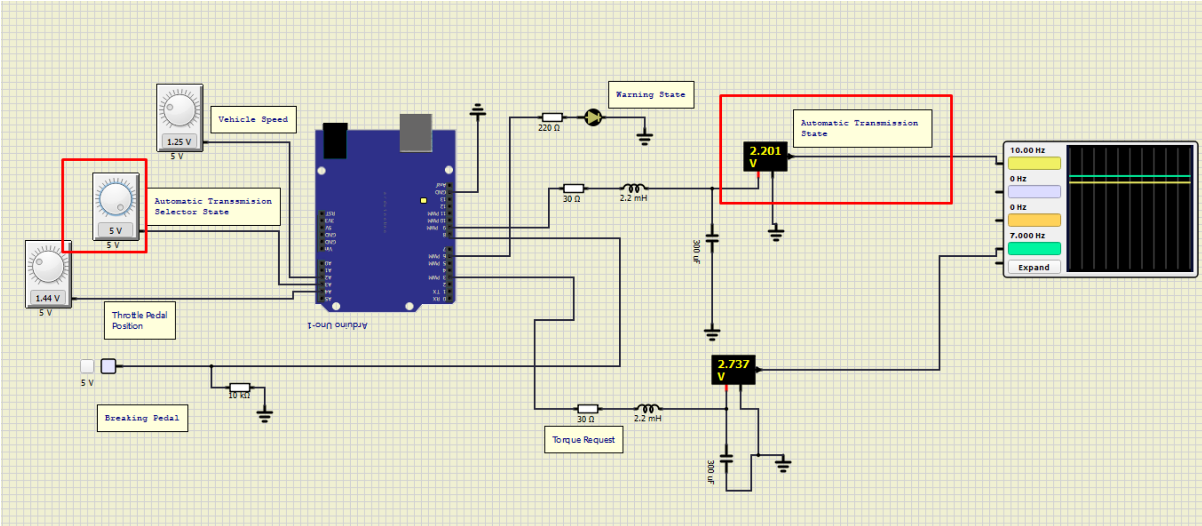

Figure 4.1 Error for malfunction of the selector (which jump to neutral state)
The Warning state is active and the led is ON.

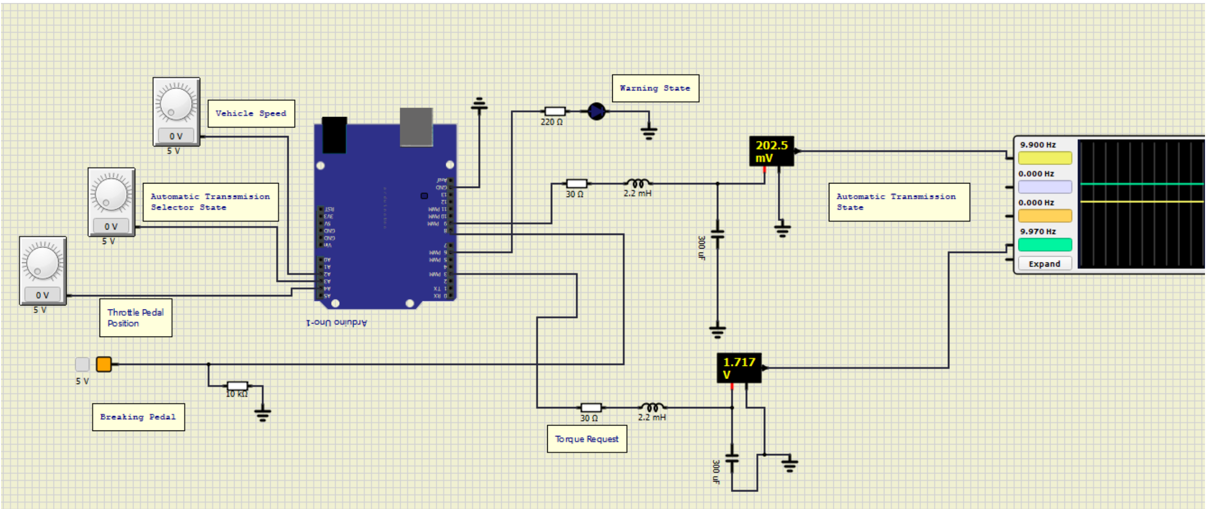

Figure 4.2 Correct parking state

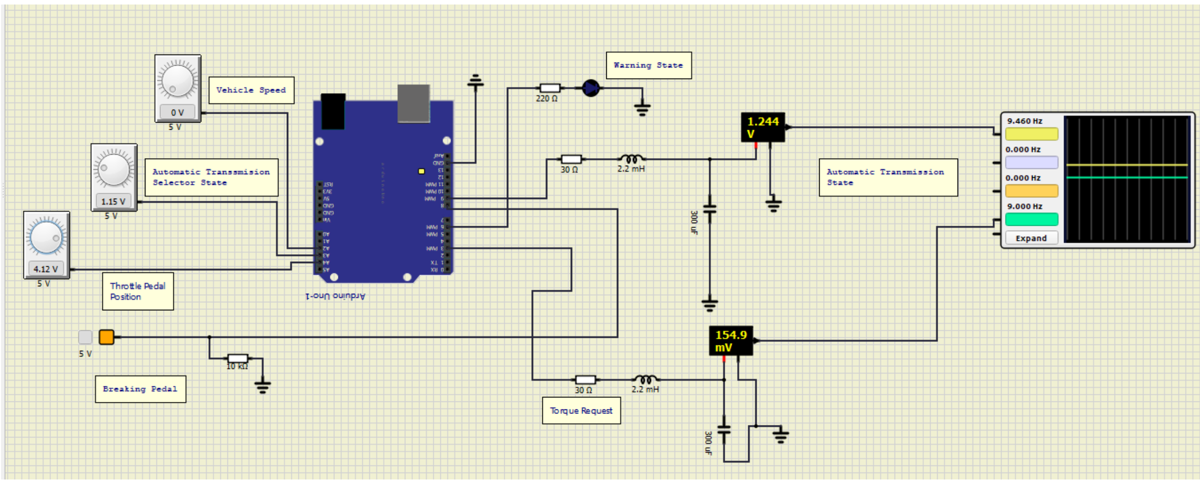

Figure 4.3 Correct reverse state

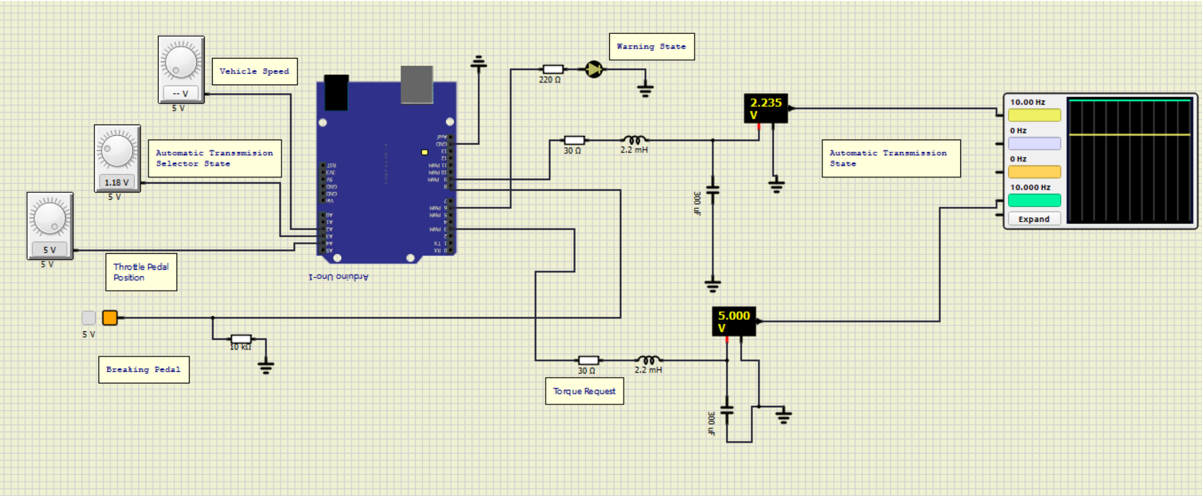

Figure 4.4 Error for pedal malfunction(reverse state but as an error jump to neutral state)

The Warning state is active and the led is ON.