Model based software design - Lab 04

Gianluca La Malfa
s290187

Gaia Sabbatini
s291532

1 I/O interfaces

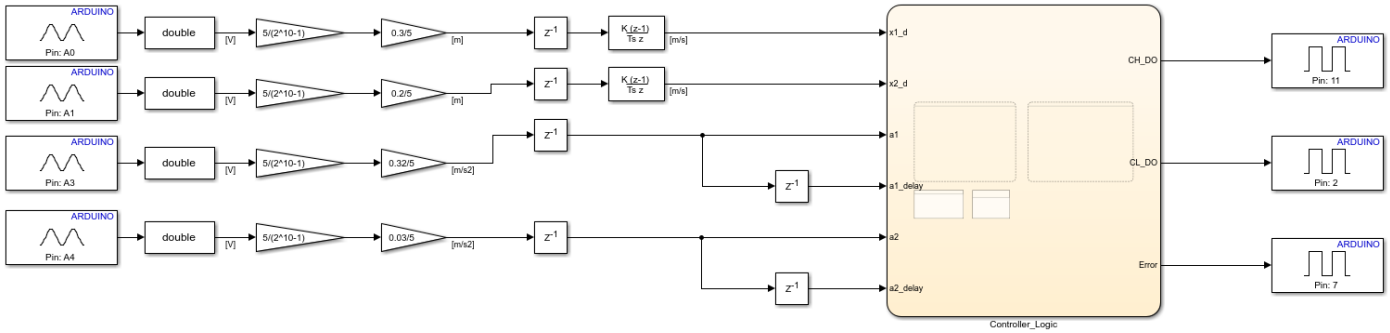

Figure 1: Controller ready for code generation

For what concerns physical interfaces, the controller receives four inputs and returns three outputs to the plant that will be simulated on Arduino Uno board (Tab.1):

- **Inputs**:
  - *Vehicle acceleration*, returns the current acceleration of the entire vehicle.
  - *Wheel acceleration*, returns the current acceleration of the wheel.
  - *Vehicle displacement* , represents the difference of the position of the vehicle with respect to its initial position.
  - *Wheel displacement* , represents the difference of the position of the wheel with respect to its initial position.
- **Outputs**:
  - *Error*, an error flag activating if the input signals are stuck for more than $20ms$
  - *CH_DO*, a boolean signal used to activate the rigid configuration.
  - *CL_DO*, a boolean signal used to activate the soft configuration.

Table 1: Controller Interfaces table

Name	Unit*	Type	Conversion formulas	Min	Max
Vehicle acceleration	m/s^2	Input	Double	-0.015	+0.015
Wheel acceleration	m/s^2	Input	Double	-0.16	+0.16
Vehicle displacement	m	Input	Double	-0.10	+0.10
Wheel displacement	m	Input	Double	-0.15	+0.15
CH_DO	-	Output	Boolean	-	-
CL_DO	-	Output	Boolean	-	-
Error	-	Output	Boolean	-	-

2 Code generation for Arduino

In order to generate the Arduino firmware the code generator App provided by Simulink was employed. First of all, the solver parameters were set up to correctly suit the controller previously developed. The first thing to do was to change the type of solver to `Fixed-step` and the `step size` value to `1/1000`. After this we moved to the `Hardware implementation` section and chose the `Arduino Uno` as hardware board, the code generation system target file automatically changed to `ert.tlc`. Finally, we moved to `Code generation` section and in particular we changed the code interface packaging to `Nonreusable function` in the `interface` subsection. After setting up all the parameters we were able to build the model and generate the `controller_arduino.hex` which represents the firmware that will be used in the following phase of the emulation of the Arduino Uno board.

3 Harness

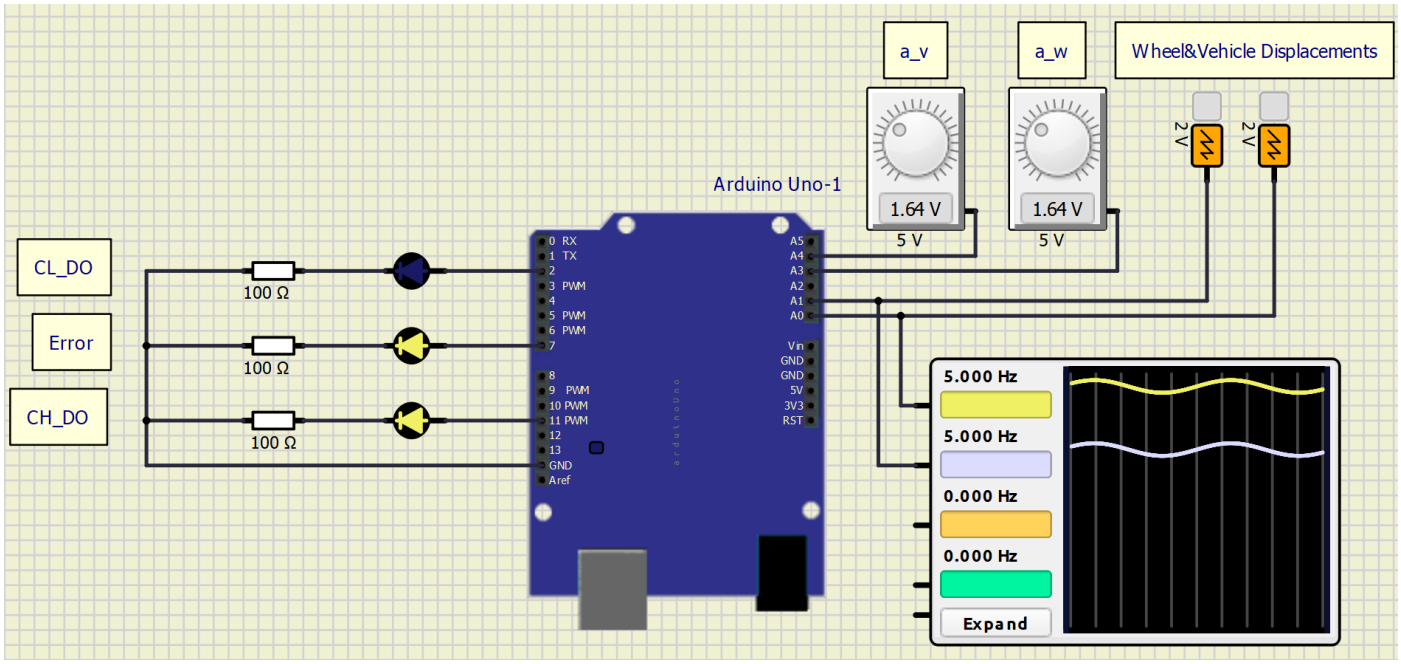

Figure 2: Arduino implementation: Constant voltage source were employed to test aplication SS

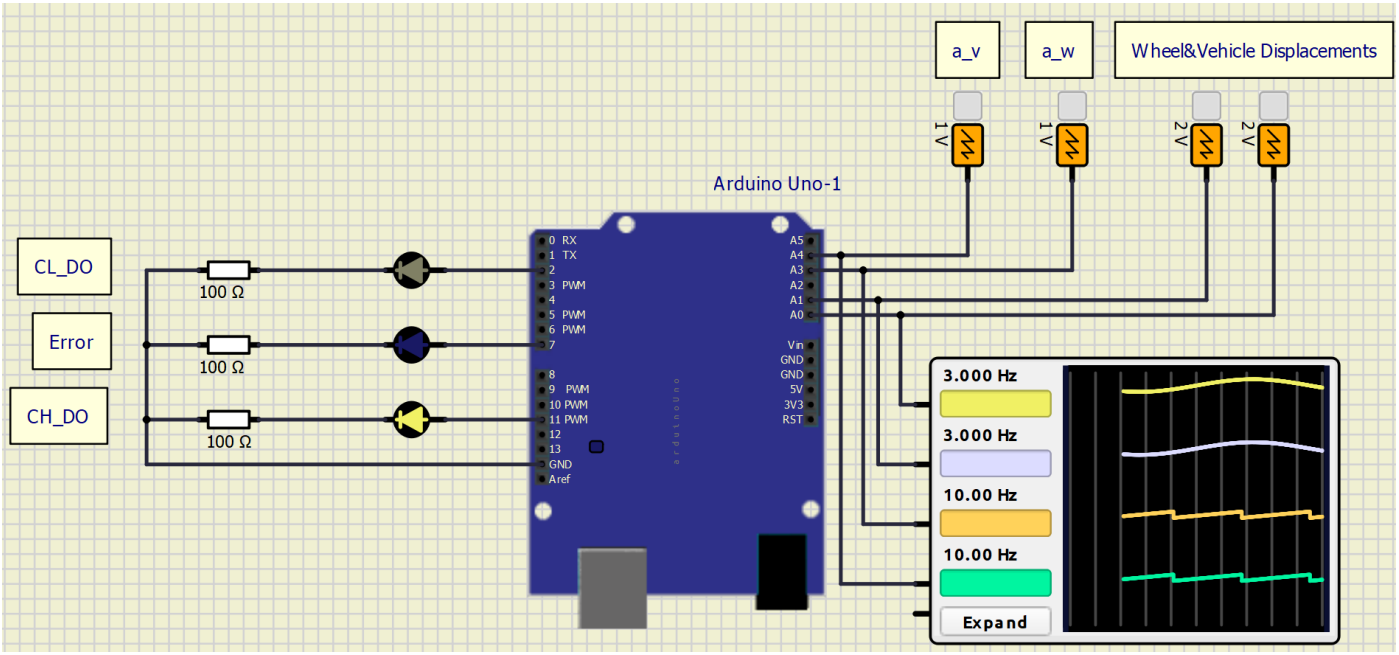

Figure 3: Arduino implementation: Waveform generators were employed to test damping factors switching

3.1 Test stimuli

To test the firmware on the Arduino Uno board four signals were generated through the SimulIDE sources components:

- **Vehicle and Wheel accelerations**, two voltage sources were deployed giving constant outputs. As a matter of fact, the model shall enter the safe state (as shown in Fig. 2) since the two accelerations values are constant through all the simulation. On the other hand, to test the application without rising the error flag two different waveform generated with the Wave gen. block of SimulIDE could be deployed (such as sawtooth or sine waveform). With this scenario the leds will highlight the switching of the damping factor, turning on or off the CL_DO or CH_DO digital signal respectively (Fig. 3).
- **Vehicle and Wheel displacements**, two sine waveforms with 2V amplitude and a voltage base of 2V returning signals between a minimum of 2V and a maximum of 4V with a frequency of 5 Hz.