Tutorial 11 — Building Controllers in PVSio-Web

Overview

This tutorial will help you to:

1. Install PVSio-web
2. Generate a controller using PVSio-Web
3. Generate an FMU from a PVSio-Web project
4. Prepare an ARDUINO Sketch with the FMU code
5. Compile and upload the Sketch into the robot

Requirements

This tutorial requires the following tools:

- Linux OS (or a VM with Linux as OS)
- INTO-CPS Application
- COE (Co-simulation Orchestration Engine) accessible to the Application
- ARDUINO IDE (1.8.5)
- avr-gcc (GCC) $\geq$ 6.3.0
- avr-g++ (GCC) $\geq$ 6.3.0

Downloading the VM (for Windows users only)

Step 1. Go to the following link: `http://releases.ubuntu.com/16.04/`, and select the desktop image that fits your machine.

Step 2. Launch Virtual Box

Step 3. Click New, give a name to the machine, e.g., PVSClass2019, its type is Linux, and its version Ubuntu. Click Continue.

Step 4. Select Create a virtual hard drive file and use default options.

Step 5. Start the new virtual machine

Step 6. Follow the instructions to install Ubuntu.

Installation of PVSio-web

Step 7. Type ”sudo apt-get install npm” and ”sudo apt-get install nodejs-legacy” on terminal to install Nodejs.

Step 8. Type ”git clone https://github.com/mapalmieri/pvsio-web” to download the folder pvsio-web

Step 9. Move into the pvsio-web folder (type ”cd pvsio-web”)

Step 10. Type ”npm install” (it will require few minutes to complete)

Make a PVSio-web Project

Step 11. Launch the script start.sh (type ”./start.sh”)

Step 12. Then open a web browser and go to the page ”http://localhost:8082/”

Step 13. On the web page, click on New Project in the topbar and name it *LFRController*.

Step 14. On the topbar of the project click ”Save Project”

Adding variables to an Emuchart diagram

Step 15. Then on the right panel, click on the on/off button next to ”EmuCharts Editor”.

Step 16. Then scroll down untill you see a new bar and click on *VARIABLES*

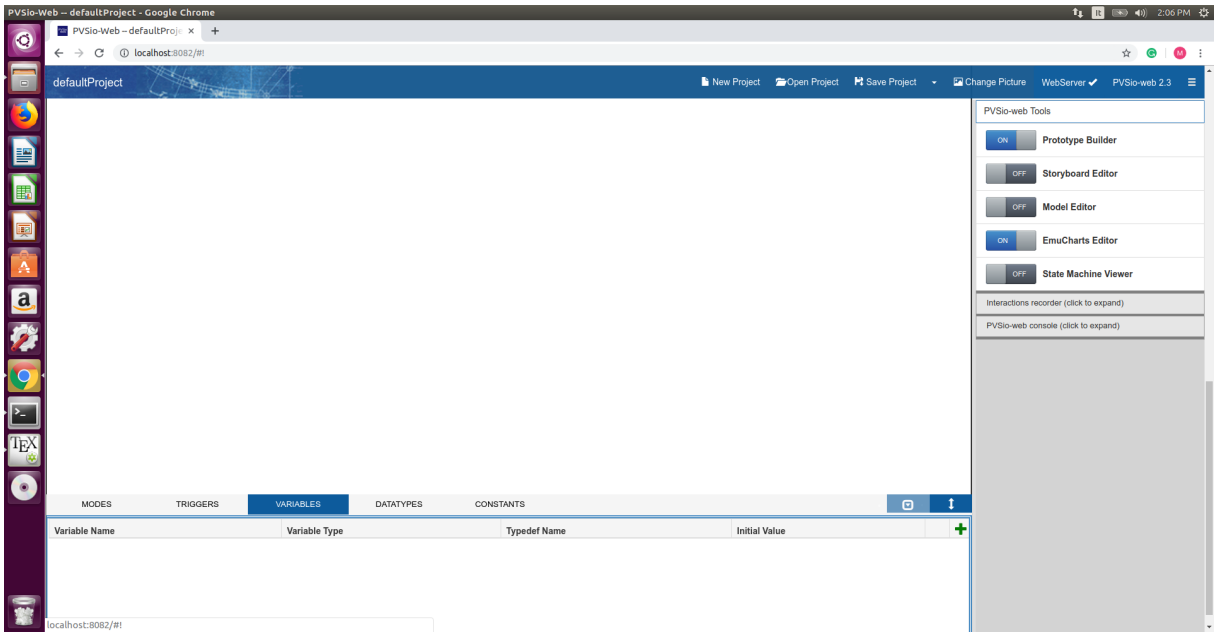

Figure 1: Figure for Step 16.

Step 17. Click on the green + on the right

1. Choose a representative name, such as ”LeftSensor”
2. This is a ”real” variable
3. ”Typedef name” is the type used in the MISRA C code, choose float64_t
4. Initial value ”0”
5. Scope Input

Step 18. Repeat for RightSensor, LeftServo(Scope: output) and RightServo (Scope: output)

Step 19. Scroll up to the Emucharts Editor bar

Step 20. On the bar, hover on File , then click on Save Chart

Step 21. On the topbar of the project click ”Save Project”

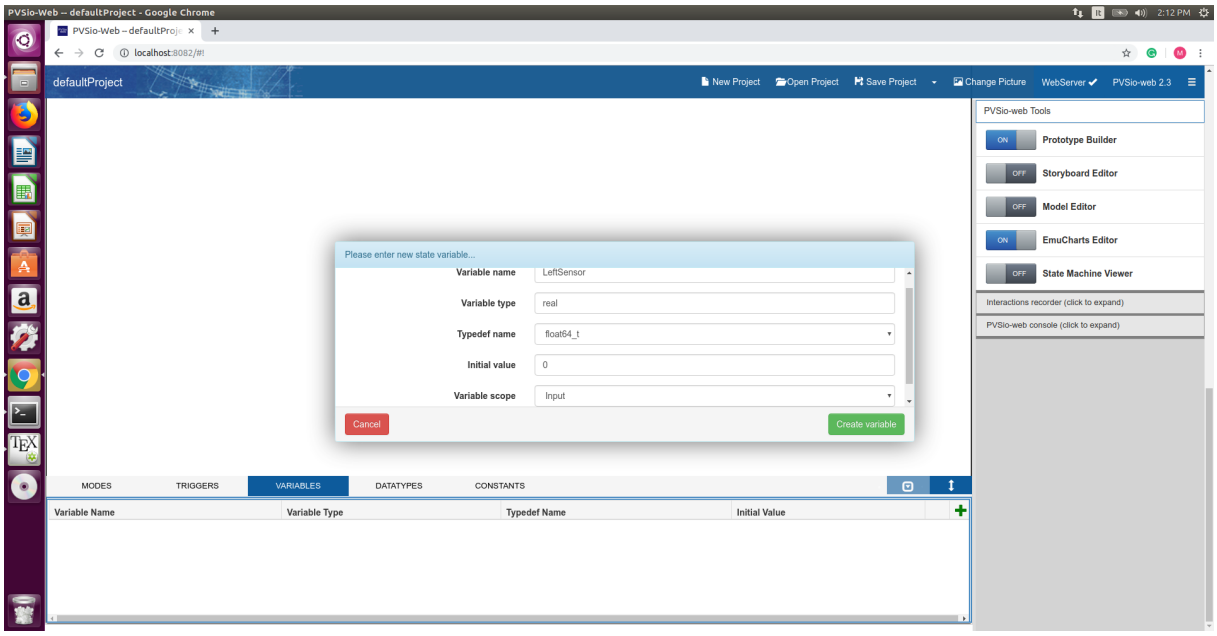

Figure 2: Figure for Step 17.

Creating the Emuchart diagram

Step 22. On the left panel click on ”Add modes” and then click on the middle section to place the mode.

Step 23. On the left panel click on ”Browse diagram” then double-click on the newly created mode and name it ”Auto”

Step 24. On the left panel click on ”Add trigger”, then click on the mode Auto to add a reentrant edge

Step 25. On the left panel click on ”Browse diagram” then double-click on the newly created trigger and copy the following snippet:

```
tick
[LeftSensor <= 150 AND RightSensor <= 150]
{LeftServo := 0.4;RightServo := -0.4}
```

This transition drives the robot forward when both sensors see the painted line (sensor value less than or equal to 150).

Step 26. Create another reentrant transition named ”tick” for turning right when only the right sensor sees the painted line (**Hint:** to turn right LeftServo should be 0.5 and RightServo should be -0.1)

Step 27. Create another reentrant transition named ”tick” for turning left when only the left sensor sees the painted line (**Hint:** to turn left LeftServo should be 0.1 and RightServo should be -0.5)

Step 28. On the bar, hover on File , then click on Save Chart

Step 29. On the topbar of the project click ”Save Project”

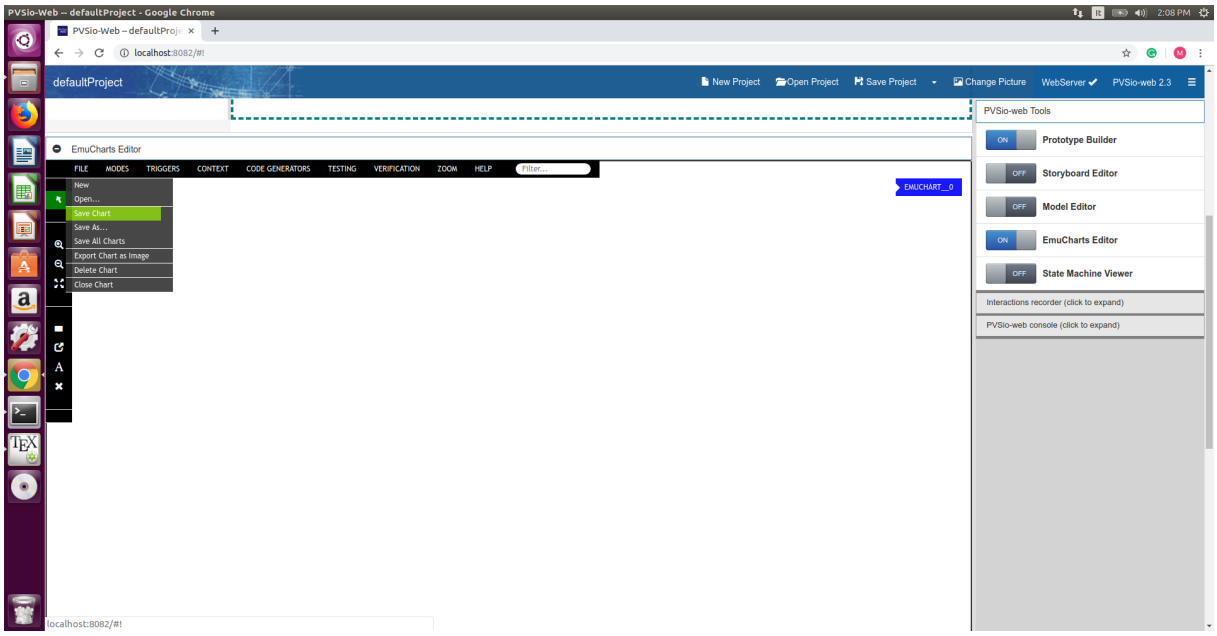

Figure 3: Figure for Step 20. and Step 28.

Generating an FMU from Emuchart

Step 30. Hover on ”CODE GENERATORS” then click on ”FMI Package”

Step 31. The files for the FMU have been generated in the fmu-pvs folder, located in the folder pvsio-web/examples/projects/LFRController along with a makefile. Open a terminal into the fum-pvs folder, and type ”make all” on the terminal.

INTO-CPS co-simulation

Step 32. Open INTO-CPS application

Step 33. File – Import example project; select the line follower robot case study.

Step 34. Move the fmu generated with PVSio-web into the FMU folder of the project.

Step 35. Modify the non-3d multimodel to use the new FMU instead of the one in overture.

Arduino setup

Step 36. Download the ARDUINO IDE version 1.8.5 from:
`https://www.arduino.cc/en/Main/Software`. Choose the zip file version and extract the folder into your `into-cps-projects/install` folder. (Choose the respective files in the case of Linux or macOS installations)

Step 37. The current C/C++ compiler version shipped in the ARDUINO IDE is prior to the required 6.3. So one needs to install an updated version. You can find one from:
`http://blog.zakkemble.co.uk/avr-gcc-builds/`.

Download and unzip it into your `into-cps-projects/install` folder. (Choose the respective files in the case of Linux or macOS installations)

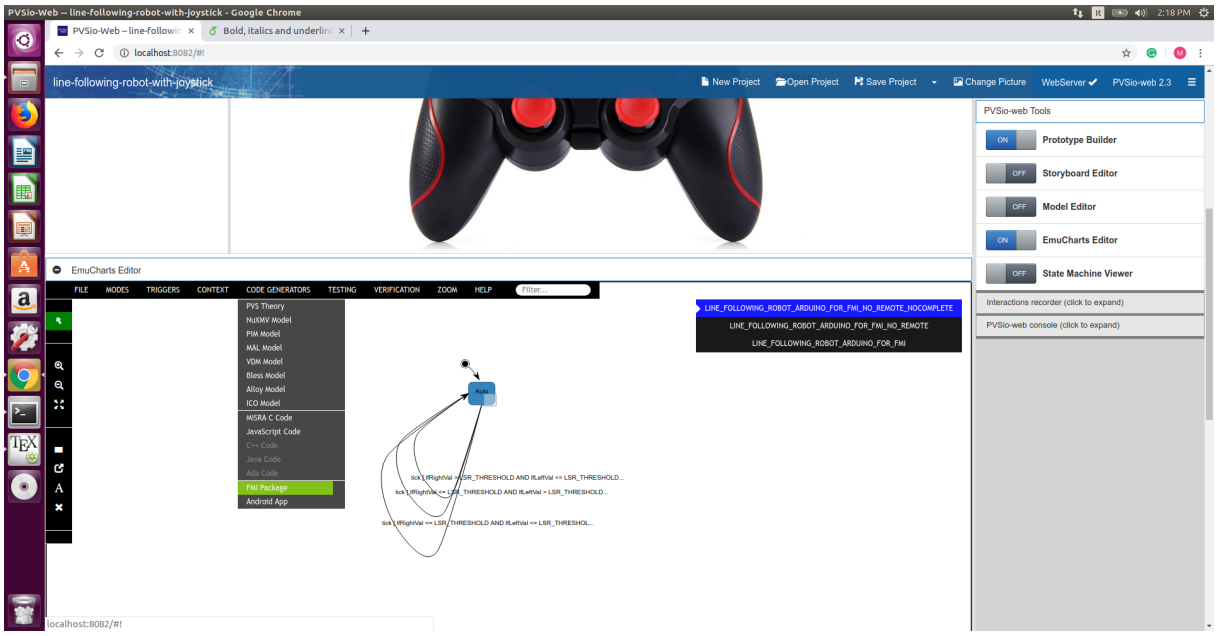

Figure 4: Figure for Step 30.

Create the FMU and upload to Arduino (Linux/macOS Only)

Step 38. Go back to the PVSio-web page and open the project line_ following_ robot: in the topbar click on Open Project, scroll down to the line_ following_ robot (**avoid line_ following_ robot1 !!**) and click open.

Step 39. Open the Emuchart Editor, hover on ”CODE GENERATORS” then click on ”FMI Package”.

Step 40. Before creating the FMU we need to apply two small modifications: in skeleton.c , misrac/line_ following_ robot.c and misrac/line_ following_ robot.h change the name ”init” with ”Init” **the different is the capital letter**.

Step 41. type ”make all” on the terminal

Step 42. Connect robot to USB and find port (for example you may find /dev/ttyACM0 after running `ls /dev/`). **if you don’t have the robot or the cable assume that you can use /dev/tty-ACM0. When you have the robot please confirm this step**

Step 43. In a terminal change to the *tutorial_10/Deploy/linux* folder. In it you should find the files: *main.cpp*, *modeldescription.h* and the *deploy.sh* script.

Step 44. Set the following variables used in the *deploy.sh* script adapting the path to your own choices:

- port - to the previous result
- gcc_path - to into_cps_project/install/avr-gcc-8.3.0-x64-linux/bin
- avr - to into_cps_project/install/arduino-1.8.5/hardware/arduino/avr/
- avrdudeconfig - to into_cps_project/install/arduino-1.8.5/hardware/tools/avr/etc/avrdude.conf

Step 45. Make the following changes to main.cpp

- in line 10, change Fmu.h with fmu.h

- comment line 11 (the other include)
- comment line 121 (LSR_ THRESHOLD)

Step 46. Move the newly created FMU in this folder.

Step 47. Run the *deploy.sh* script with line_ follower_ robot.fmu as a parameter.