Tutorial 10 — Deploying the LFRController

Overview

This INTO-CPS tutorial will show you how to:

1. Export the *LFRController* VDM project as an FMU
2. Prepare an ARDUINO Sketch with the FMU code
3. Compile and upload the Sketch into the robot

The instructions in this tutorial were adapted from `https://github.com/INTO-CPS-Association/example-line_follower_robot/tree/arduinoMegaDeploy/deployment`.

Requirements

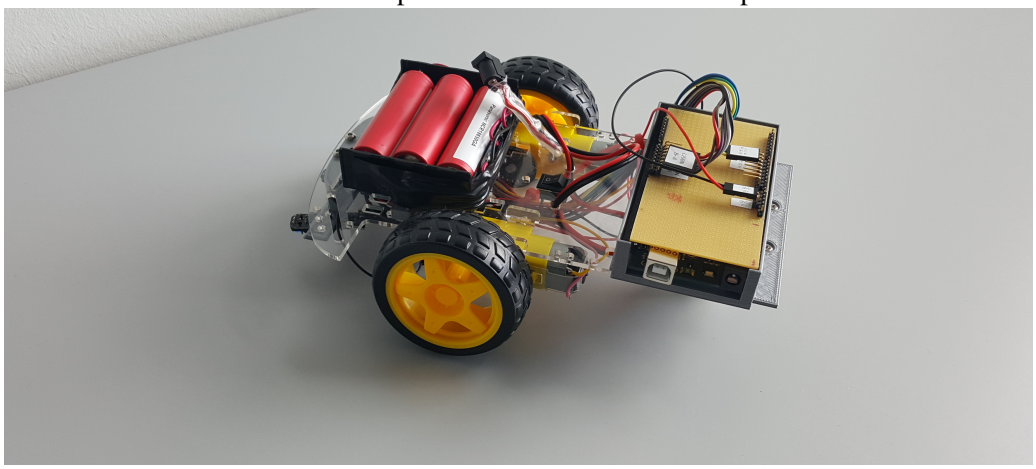
This tutorial requires the following tools from the INTO-CPS tool chain to be installed:

- Overture FMU Import/Exporter CLI

In addition, the following software must be installed without recurring to the Download Manager:

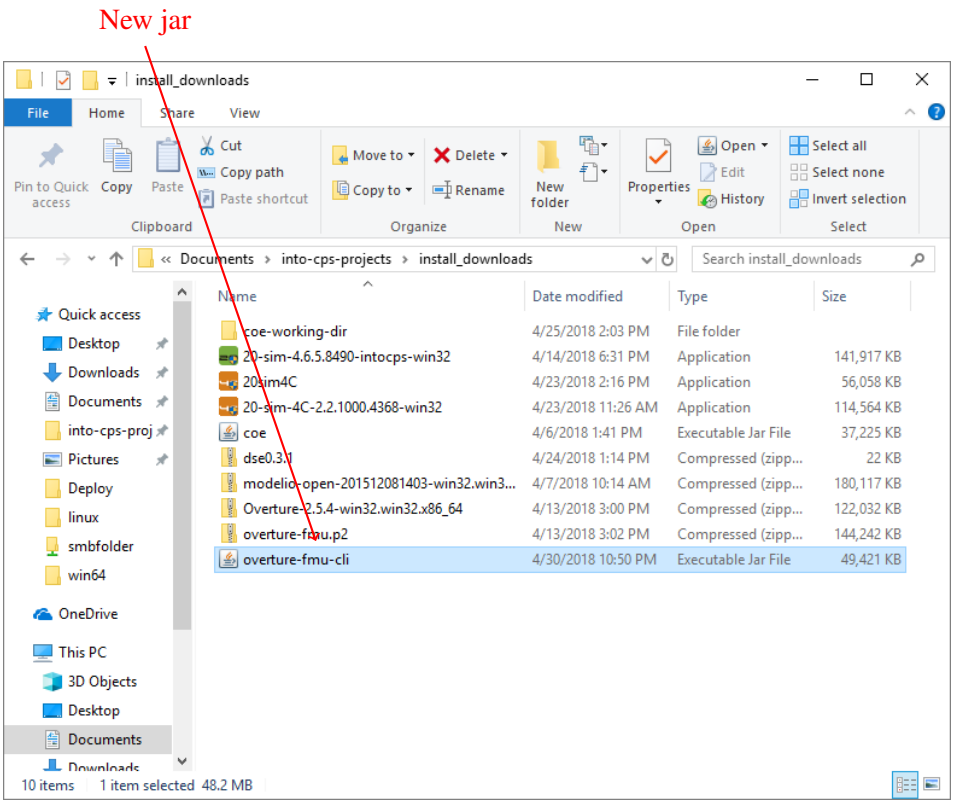
- ARDUINO IDE (1.8.5)
- avr-gcc (GCC) $\geq$ 6.3.0
- avr-g++ (GCC) $\geq$ 6.3.0

In terms of hardware one needs a printed version of the line map and the robot:



1 Download and install the dependencies

Step 1. Install the Overture FMU Import/Exporter CLI using the INTO-CPS application *Download Manager*. Please make sure you download the **CLI** version. After the download you will find a new .jar file it in your `into-cps-projects/install_downloads`.



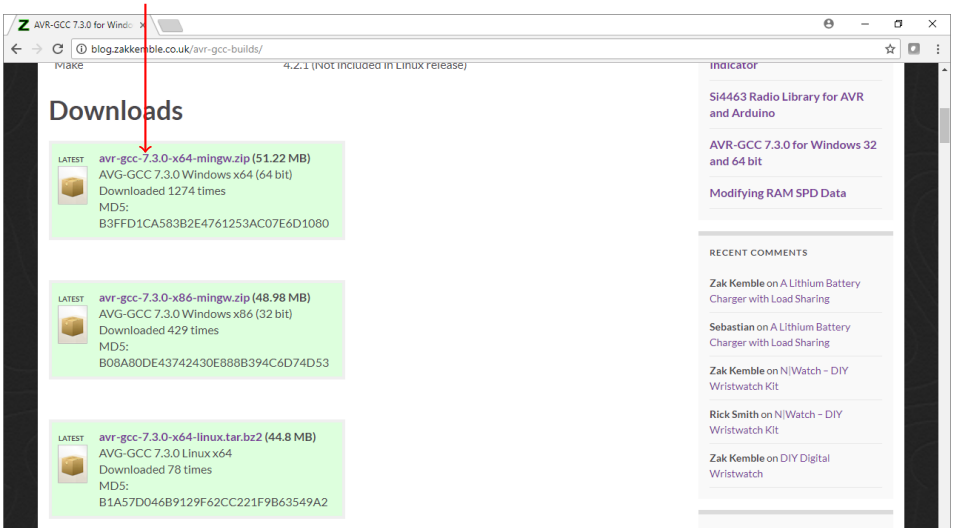
Step 2. Download the ARDUINO IDE version 1.8.5 from:
`https://www.arduino.cc/en/Main/Software`. Choose the zip file version and extract the folder into your `into-cps-projects/install` folder. (Choose the respective files in the case of Linux or macOS installations)

Download the Zip File...

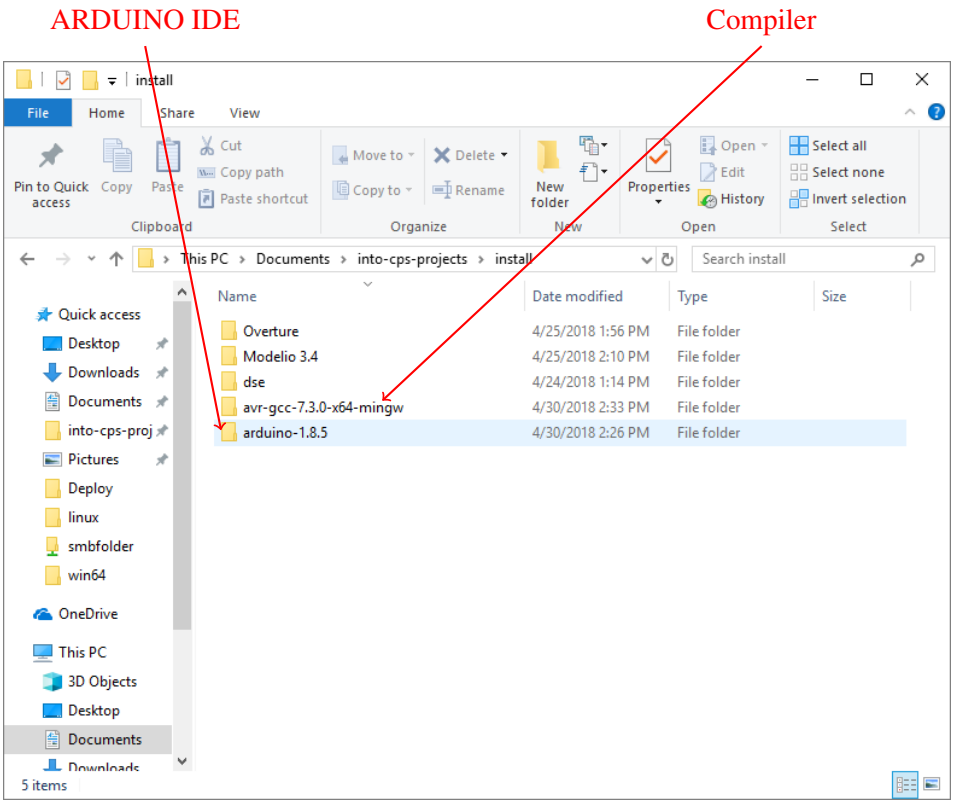
Step 3. The current C/C++ compiler version shipped in the ARDUINO IDE is prior to the required 6.3. So one needs to install an updated version. You can find one from:
`http://blog.zakkemble.co.uk/avr-gcc-builds/`.

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

Download the Zip File...



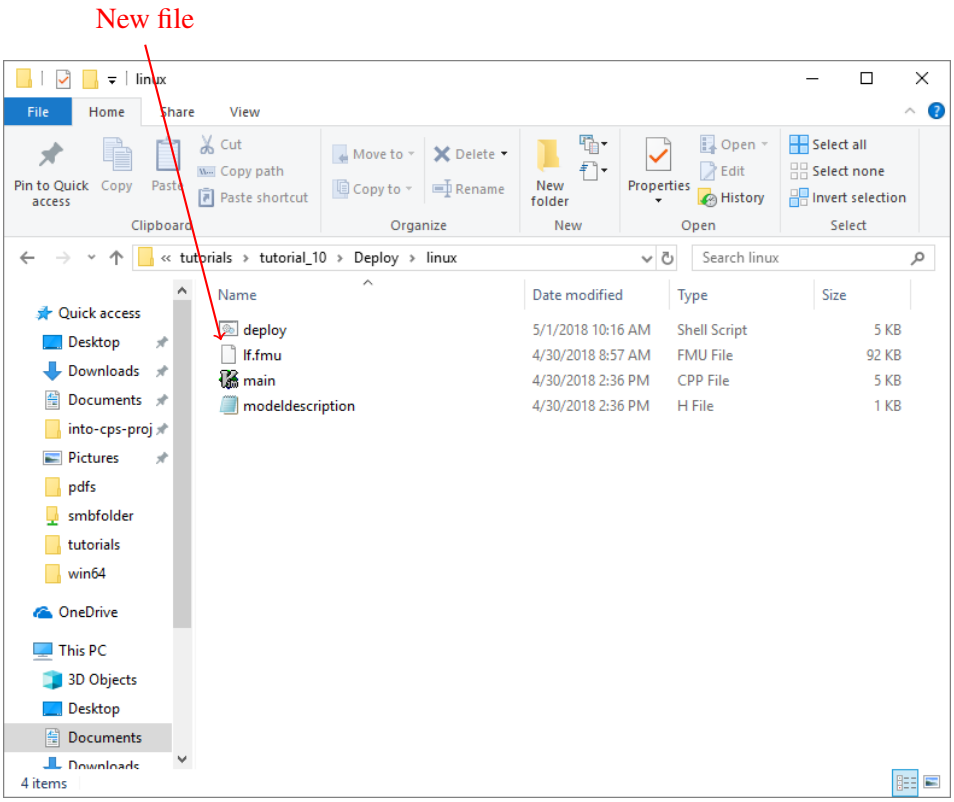
Step 4. At this point your `into-cps-projects/install` folder should contain the two folders corresponding to the ARDUINO IDE and the updated version of the compiler. And you are ready for the next step.



2 Export the LFRController VDM project as an FMU

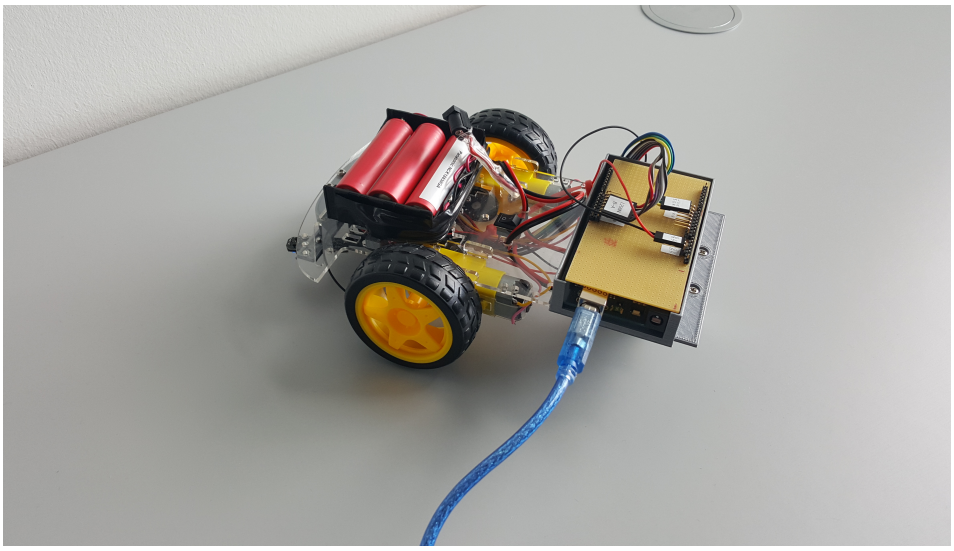
Step 5. Open a Command Prompt. Change directory into *Deploy/linux* or (*Deploy/win64)* inside the *tutorial 10*. It is important you run the command below in this folder or copy the output file produced (*lf.fmu*) into it. If the command succeeds you should see a new file in the folder as depicted.

```
java -jar path_to_install_downloads/overture-fmu-cli.jar
     -export source
     -name lf
     -output .
     -root ../../Models/LFRController
```



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

Step 6. Connect robot to USB and find port (for example you may find /dev/ttyACM0 after running `ls /dev/`).



Step 7. 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 8. Set the following variables used in the *deploy.sh* script adapting the path to your own choices:

- port - to the result of the previous step
- gcc_path - to into_cps_project/install/avr-gcc-7.3.0-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

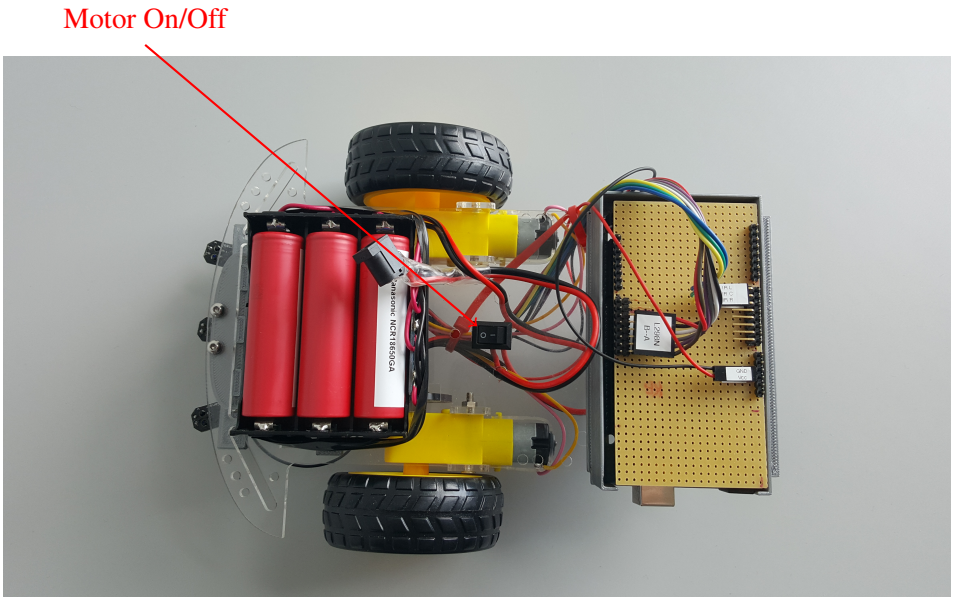
Run the *deploy.sh* script with lf.fmu as a parameter.

```
rojects/tutorials/tutorial_10/Deploy/linux$ port=... gcc_path=... avr=... avrdudeconfig=... ./deploy.sh lf.fmu
```

Step 9. You should see the following message in case everything works as expected.

```
avrdude: safemode: efuse reads as FD
avrdude: reading input file "AURobot.hex"
avrdude: input file AURobot.hex auto detected as Intel Hex
avrdude: writing flash (58026 bytes):

Writing | ################################################## | 100% 9.30s

avrdude: 58026 bytes of flash written
avrdude: verifying flash memory against AURobot.hex:
avrdude: load data flash data from input file AURobot.hex:
avrdude: input file AURobot.hex auto detected as Intel Hex
avrdude: input file AURobot.hex contains 58026 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 7.42s

avrdude: verifying ...
avrdude: 58026 bytes of flash verified

avrdude: safemode: hfuse reads as D8
avrdude: safemode: efuse reads as FD
avrdude: safemode: Fuses OK (E:FD, H:D8, L:FF)

avrdude done.  Thank you.

Done.
hdm@drcagliari:~/into-cps-projects/tutorials/tutorial_10/Deploy/linux$
```

Step 10. Unplug the USB cable, place the robot in the line, turn the motor switch on!



4 Compile the FMU and upload to Arduino (Windows Only)

Step 11. Connect robot to USB and find port (for example you may find COM3 by typing `mode`) in a Command Prompt or check if the ARDUINO IDE detects the port automatically

Step 12. Set the following variables used in the *deploy.sh* script adapting the path to your own choices:

- gcc_path - to into_cps_project/install/avr-gcc-7.3.0-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 13. Fill in the missing details in *deploy.sh* script and run with the port as a parameter.

5 Additional Exercises

Step 14. Browse the different components descriptions you can find inside the *platform* folder in the repository:
`https://github.com/INTO-CPS-Association/example-line_follower_robot/tree/arduinoMegaDeploy/deployment`

Step 15. Workout the details of the hardware platform.

Step 16. Find the relation between the VDM LFRController and the contents of the *main.cpp* file.

Step 17. Currently the robot has an extra mounted and unused sensor... How could you use it to improve the robot behaviour?

Step 18. Find the STL file for the sensor mount:`https://github.com/INTO-CPS-Association/example-line_follower_robot/blob/arduinoMegaDeploy/deployment/platform/LF_sensor_holder.stl` How could you use it in future iterations/improvements of the robot?

Congratulations!
You have reached the end of this tutorial session!