Cross-compilation and testing of OpenModelica-generated FMUs

Martin Sjölund

Programming Environments Laboratory (PELAB)
Department of Computer and Information Science
Linköping University

2016-02-01

OpenModelica – Installers

OpenModelica is compiled for many architectures, which means there are many versions of OpenModelica to test:

- Linux
  - i686-linux-gnu
  - x86_64-linux-gnu
  - arm-linux-gnueabihf
- OSX
  - i386-apple-darwin15
  - x86_64-apple-darwin15
- Windows
  - MinGW 32-bit i686-w64-mingw32

OpenModelica – Installers

OpenModelica is compiled for many architectures, which means there are many versions of OpenModelica to test:

- Linux
  - i686-linux-gnu
  - x86_64-linux-gnu
  - arm-linux-gnueabihf
- OSX
  - i386-apple-darwin15
  - x86_64-apple-darwin15
- Windows
  - i686-w64-mingw32
  - x86_64-w64-mingw32

OpenModelica – Additional Architectures

OpenModelica can also compile code for more platforms:

- Windows, Visual Studio

OpenModelica – Additional Architectures

OpenModelica can also generate different kinds of code:

- C runtime (default)
- C++ runtime
- HPCOM

OpenModelica – FMI Version Choices

OpenModelica can also generate different kinds of FMUs:

- 1.0 ME
- 1.0 CS
- 2.0 ME
- 2.0 CS

Cross-Compilation

Cross-compilation is the ability to compile for a different platform than the current one.

- It is something OpenModelica did not support earlier.
- Imagine using a 64-bit GUI for the compilation process (uses much RAM), and a 32-bit executable for the simulation.
- Generating code for embedded systems.

Ubuntu Installer

The Ubuntu packages are now aware of cross-compilation and install to /usr/lib/x86_64-linux-gnu/, etc. But...

- If you try to install libomc:i386, the system BLAS libraries conflict with 64-bit version (upstream bug).
- Might work in the future.
- Ubuntu does ship with cross-compilers for armhf and mingw-w64 as optional packages.

Windows Installer

Work in progress by Adrian.

- Based on msys2, which includes a package manager (easier to update for new packages)
- Using mingw32 and mingw64 gcc compiler 5.3.0
- Cross compilation just for 32 and 64 bit windows

FMUs Approach

The new FMU approach is different due to a number of reasons:

- We wanted all necessary code to be part of the FMU, with no external dependencies.
  - Previously, the FMU depended on having OpenModelica installed (some shared objects were not part of the FMU).
- We wanted to support embedded platforms, etc that OpenModelica either does not run on or is not powerful enough to run OM:
  - Source-code FMUs can be compiled on other platforms.
  - Source-code FMUs can also be cross-compiled on this or another platform.

Source-code FMUs

Based on autoconf, generating a configure script (same script for all FMUs).

- autoconf supports cross-compilation.
- FMU includes all solvers, etc needed for the basic OM runtime system, including dgesv from LAPACK.
- configure script allows choosing dynamically linked executable (like before, smaller FMU).
- configure script uses a statically linked FMU when cross-compiling.
- The OMC API supports choosing multiple targets when generating the FMU.
- Additional targets can be added after the FMU has been generated.

Source-code FMUs – OMC API

```
buildModelFMU(FmuExportCrossCompile, version="2.0",
    fmuType="me_cs", platforms={
  // Requires osxcross manually installed
  "i386-apple-darwin15",
  "x86_64-apple-darwin15",
  // apt-get install gcc-arm-linux-gnueabihf
  "arm-linux-gnueabihf",
  "x86_64-linux-gnu",
  // apt-get install libc6-dev:i386
  "i686-linux-gnu",
  // apt-get install gcc-mingw-w64-x86-64
  "x86_64-w64-mingw32",
  // apt-get install gcc-mingw-w64-i686
  "i686-w64-mingw32"
});
```

Source-code FMUs – Behind the Scenes

```
$ unzip Test.fmu -d fmutmp
$ cd fmutmp/sources
$ ./configure --host=i686-w64-mingw32 CFLAGS=-Os
$ make -j6

...
i686-w64-mingw32-gcc -shared -o FmuExportCrossCompile.dll
    ...
mkdir -p ../binaries/win32
cp FmuExportCrossCompile.dll ... ../binaries/win32/
rm -f ...
cd .. && rm -f ../Test.fmu && zip -r ../Test.fmu *
```

How to Test the FMU?

- OpenModelica servers mostly run 64-bit Linux (only).
- We would like to test the FMU in a single job, but test multiple platforms...
- OpenModelica does not come with a stand-alone FMU simulator.
- FMUChecker is cross-platform, but only comes with a simple euler solver.

Testing the FMU

- 64-bit Linux: Use FMUChecker (native)
- 32-bit Linux: Use FMUChecker (32-bit Linux version), since we can run the executables natively
- 64-bit Windows: Use FMUChecker (64-bit Windows version), running using wine
- 32-bit Windows: Use FMUChecker (32-bit Windows version), running using wine
- 64-bit OSX: Use darling (64-bit version), which is similar to wine
- 32-bit OSX: Could use darling (32-bit version), but did not compile on a native 64-bit Linux...
- ARM: Could perhaps run in a simulator, but we did not test this

Tested Platforms

- Can test cross-compilation for 7 platforms, 4 FMU versions (1.0/2.0 ME/CS), using C runtime (28 different targets).
- Can test execution of 5 platforms (total 20 targets).
- All using a single computer running a 64-bit Ubuntu Linux.

Future Work

- Windows installer.
- Supporting Modelica libraries using external C-code (embed those sources into the FMU).
- Embed Resources from Modelica libraries into the FMU.