How to Cross-compile a Simulink Plugin in Windows for Linux

Date: 2022-07-27

Author: IPG Solutions Engineering

Release No.: CM-10.2.2, CM-11.0.1

Introduction

These instructions show how to cross compile the example BodyCtrl_RTW.mdl plugin. The plugin compilation takes place in win64, but the final CM compilation happens in linux64. The examples use R2018a & CM10.2.2, but this can be any combination of CM10.2.2 or CM11.0.1 with a compatible Matlab version.

Make sure to backup any files that are been replaced in the installation directories. This can also be done by renaming the file (e.g. adding a .bak extension).

Please read the steps carefully. You can also skip some steps by downloading and adapting the resulting files.

Requirements

-Windows CM installation
-Windows Matlab installation
-Linux CM installation
-Modified cmplugutil.exe (provided in **Required_files.zip**)

Typical installation path for CM10.2.2

Linux: opt/ipg/carmaker/linux64-10.2.2
Windows: C:/IPG/carmaker/win64-10.2.2

Solution

In Windows

Step 1:
Copy necessary files from the Linux installation to the Windows installation:
- include/MakeDefs.linux64
- Matlab/R2018a/CarMaker_unix-linux64.tmf
- Matlab R2018a /libmatsupp-linux64.a

Step 2:
Add the modified cmplugutil.exe that accepts "-a linux64" to the Windows installation:
- copy cmplugutil.exe.CM10.2.2 from **Required_files.zip**
- paste it in bin/cmplugutil.exe.CM10.2.2
- rename it to cmplugutil.exe

Step 3:
Adapt MakeDefs.linux64:
- Change installation directory and architecture

```
CARMAKER_DIR ?=         C:/IPG/carmaker/win64-10.2.2

ARCHF = win64
```

- Define cross compilation toolchain for win architecture (in section “*### compiler and linker options”*)

```
ifneq ($(filter win32 win64, $(ARCHF)), )
  CC =      linux-gcc
  CCC =     linux-g++
  CXX =     $(CCC)
  LD =      linux-ld
  AR =      linux-ar
else
  CC =      gcc
  CCC =     g++
  CXX =     $(CCC)
  LD =      ld
  AR =      ar
endif
 # Do not change FLAGS below!
```

Step 4:
Adapt CarMaker_unix-linux64.tmf:
- Change installation directory to Windows

```
include C:/IPG/carmaker/win64-10.2.2/include/MakeDefs.linux64
```

```
MAKECMD       = "C:/IPG/carmaker/win64-10.2.2/bin/cmmake"
```

Resulting Files: **Steps_1to4.zip** (note that the folder tree structure is included)

Step 5:
Create a new Project folder (with BodyCtrl_RTW.mdl):
- Open CM-GUI
- set path to Project Folder (e.g. in C:\CM_Projects\CM10)
- Select components:
  - Source / Build Enviroment
  - Simulink Coder (RTW) Examples

Step 6:
Build Plugin from BodyCtrl_RTW.mdl with Simulink Coder
- In Matlab:
  - Navigate to Project Folder\src (e.g. in C:\CM_Projects\CM10\src)
  - Run "cmenv.m"
  - Open "BodyCtrl_RTW.mdl"
  - If a new model is created, select target architecture: linux64

- In Simulink:

- Go to Model Settings > Code Generation > Template makefile:
    - Set to CarMaker_unix-linux64.tmf
- Go to Model Settings > Code Generation > CarMaker:
    - Deselect “Automatic ‘make’ in the project source directory” (this is done in step 10 in linux)
- Build (ctrl +B)

Resulting Files: **Steps_1to6.zip**

In Linux

Step 7:
Create a new Project folder:
- set path to Project Folder (e.g. home/CM_Projects/CM10)
- Select component:

- Source / Build Enviroment

Step 8:
Copy plugin components from Project Folder/src in windows to linux:
- libBodyCtrl_RTW_linux64.a
- BodyCtrl_RTW_CarMaker_rtw/BodyCtrl_RTW.mk
- BodyCtrl_RTW_CarMaker_rtw/BodyCtrl_RTW_wrap.h

Step 9:
Adapt the Plugin make file (BodyCtrl_RTW.mk):
- In BodyCtrl_RTW.mk change the path for Linux

```
include /opt/ipg/carmaker/linux64-10.2.2/include/MakeDefs.linux64
```

- In BodyCtrl_RTW.mk delete everything else in the .mk file except for:

```
# Matlab version in use.
MATSUPP_MATVER  = R2018a
MATSUPP_NUMVER  = 90400


# ------------------------------------------------------------------------------
# No user defined settings below this line!
# ------------------------------------------------------------------------------

# The following macros are read by the Real-Time Workshop build procedure:
MATLAB_ROOT   = $(subst \,/,C:\Program Files\MATLAB\R2018a)
```

```
MODEL           = BodyCtrl_RTW
WRAPPER                 = BodyCtrl_RTW_wrap
CLASS           = SuspExtFrcs
DESCRIPTION     = "BodyCtrl_RTW"
MAINMODEL       = ""
PARAMID                 = 1
SYMBOLS                 = ""
```

```
# Directory where the CarMaker executable is compiled.
CM_SRCDIR         = "$(subst \,/,..)"
CM_PROJHOOKUP            = 1
CM_PROJMAKE   = 1
```

```

ModelMainLib:    ../lib$(MODEL)_$(ARCH).a
              $(QECHO) " MKENTRY  $(CLASS) $(MODEL) $(PARAMID)"
              $Q $(CMPU) mkentry -s $(CM_SRCDIR) -c $(CLASS) -m $(MODEL) -d
$(DESCRIPTION) -i $(PARAMID)
ifeq ($(CM_PROJHOOKUP), 1)
              $(QECHO) " HOOKUP   $(MODEL)"
              $Q $(CMPU) hookup  -s $(CM_SRCDIR) -c $(CLASS) -m $(MODEL) -a $(ARCH) -
mm $(MAINMODEL) -mv $(MATSUPP_MATVER) -mr $(MATLAB_ROOT)
endif
```

Resulting Files: **Steps_1to9.zip**

Step 10:
Perform a rebuild of the plugin:
-Open terminal and run the commands

```
cd home/CM_Projects/CM10/src/BodyCtrl_RTW_CarMaker_rtw
make -f BodyCtrl_RTW.mk
```

Step 11:
Compile the .linux64 executable that includes the plugin:
-In the previous terminal run the commands

```
cd ..
make
```