Getting Started with IPC

A. Installing IPC

Other components required:

- DSP/BIOS 6.31 (see release notes for precise build version)
- XDCTools 3.20 (see release notes for precise build version)

1. Install IPC using either the Linux (ipc_setuplinux_1_22_xx_xx_eng.bin) or the Windows (ipc_setupwin32_1_22_xx_xx_eng.exe) installation application.

   NOTE: Steps 2-4 are needed if and only if it is necessary to build IPC applications outside CCSv4

2. Add the IPC package path (i.e. <ipc_install>/ipc_1_22_xx_xx/packages) into XDCPATH. XDCTools and BIOS6 must also be present in the XDCPATH. The following template may be used to configure XDCPATH:

   ```
   C:/Program Files/Texas
   Instruments/CCSv4/DebugServer/packages;C:/Program Files/Texas
   Instruments/<IPC_PRODUCT>/packages;C:/Program Files/Texas
   Instruments/<BIOS_PRODUCT>/packages;C:/Program Files/Texas
   Instruments/<XDCTOOLS_PRODUCT>/packages;
   ```

3. Copy and rename the supplied default config.bld as specified below

   ```
   % cp <ipc_install>/ipc_1_22_xx_xx/etc/config.bld.default
   <ipc_install>/ipc_1_22_xx_xx/packages/config.bld
   ```

4. Set/modify the codegen tool paths if necessary. The defaults usually work if CCSv4 was installed in the default path along with codegen tools.
   - Set the rootDir in config.bld to point to your codegen tools.
   - Uncomment the Build.targets as needed.

   NOTE: The config.bld.default supplied with IPC file is very similar to the one supplied with BIOS 6

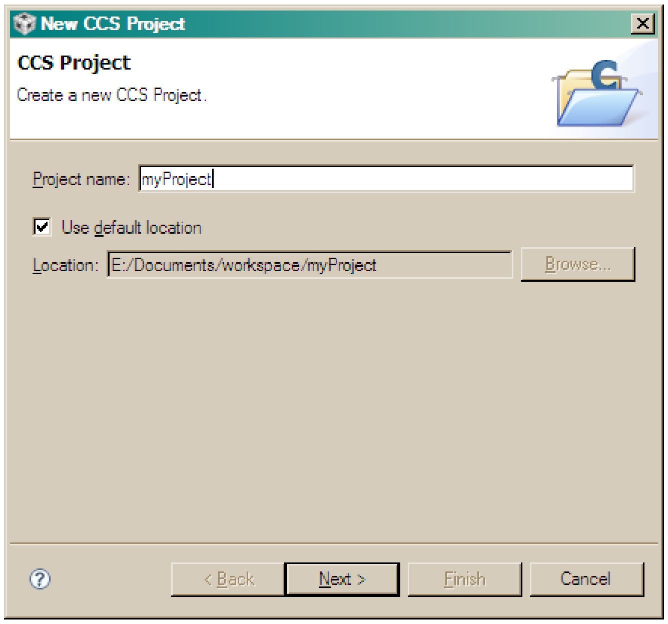
B. Building IPC examples within CCSv4

NOTE: The version numbers displayed in the screenshots in this section might not be equal/compatible with the versions required for this IPC release. Refer to the IPC release notes for IPC requirements.

1. Launch CCS and ensure that the newly installed IPC product is in your RTSC path:
   - You should be prompted to use the new product. Select the new product when prompted.
   - Click on 'Window' → 'Preferences'
   - Navigate to CCS→ RTSC

- Ensure that a BIOS 6.31 release is selected in the "Products and Repositories" list
- Ensure that a compatible XDCtools 3.20 release is selected in the "XDCtools version" dropdown list
- Ensure that your IPC product that was installed is selected in "Products and Repositories"
- Add/select any other repositories you might need for your application in this configuration panel

2. Restart CCS if any changes were made in the previous step and if prompted to do so.
3. Now you should be able to create a new CCSv4 project. Click on File→New Project

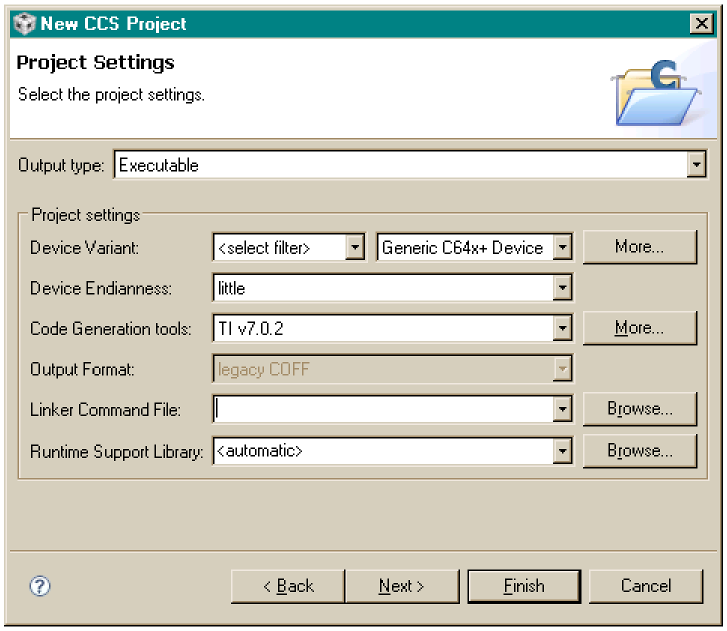


Project name: myProject

☑ Use default location

Location: E:/Documents/workspace/myProject

4. Select the class of processors that you need to build for 'i.e. c6000'
5. Select device-specific attributes as shown below.


New CCS Project

Project Settings

Select the project settings.

Output type: Executable

Project settings

Device Variant: <select filter> Generic C64x+ Device

Device Endianness: little

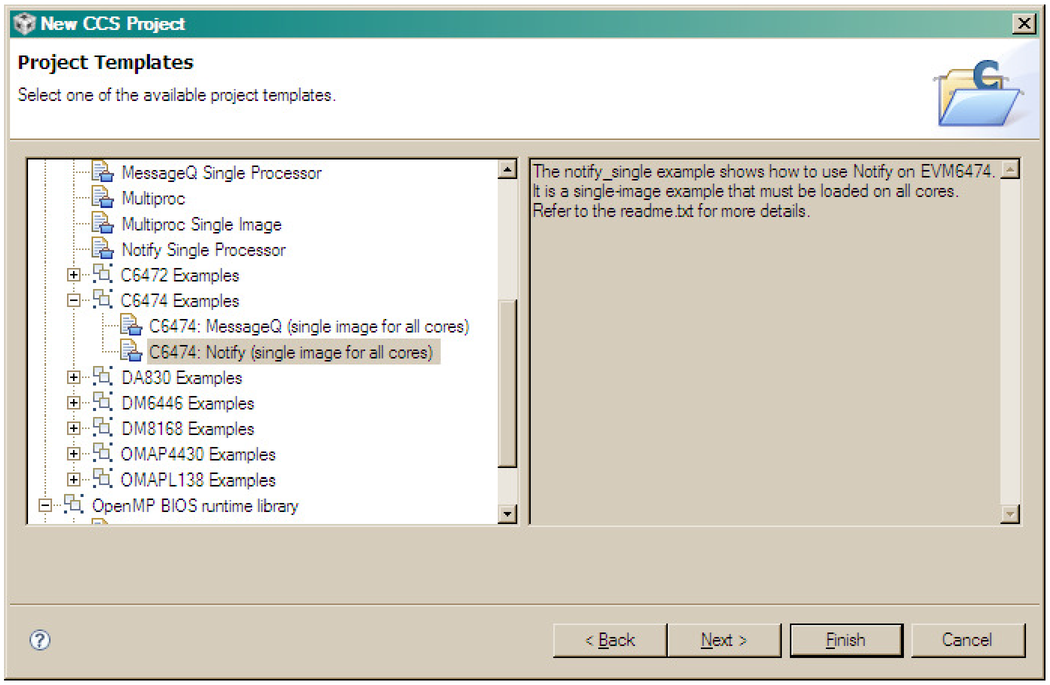
Code Generation tools: TI v7.0.2

Output Format: legacy COFF

Linker Command File:

Runtime Support Library: <automatic>

6. Select an IPC example for your device from the 'Projects Templates' wizard pane.


New CCS Project

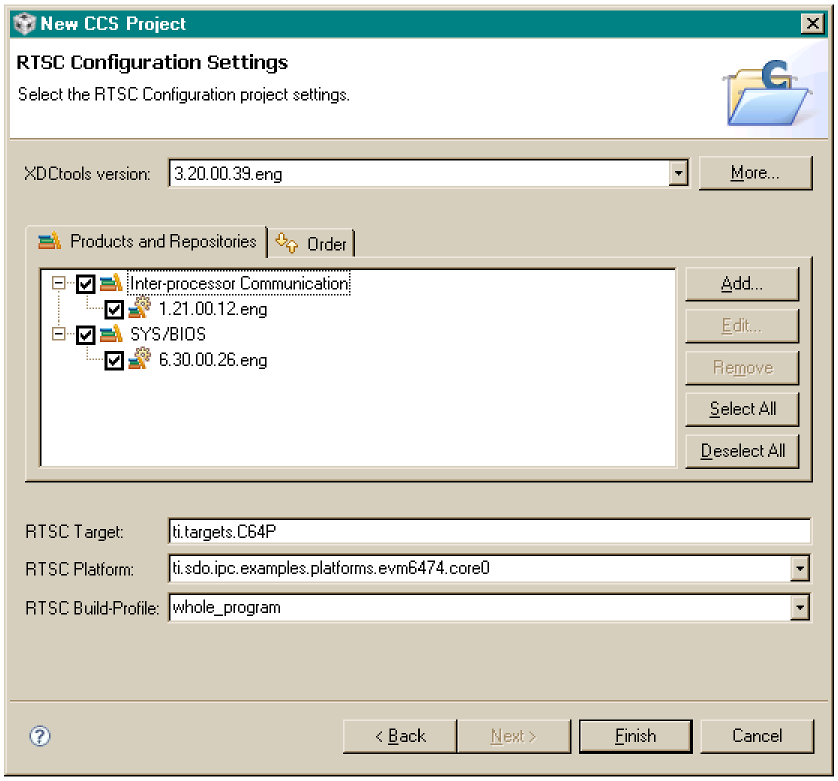
Project Templates

Select one of the available project templates.

- MessageQ Single Processor
- Multiproc
- Multiproc Single Image
- Notify Single Processor
- C6472 Examples
- C6474 Examples
  - C6474: MessageQ (single image for all cores)
  - C6474: Notify (single image for all cores)
- DA830 Examples
- DM6446 Examples
- DM8168 Examples
- OMAP4430 Examples
- OMAPL138 Examples
- OpenMP BIOS runtime library

The notify_single example shows how to use Notify on EVM6474. It is a single-image example that must be loaded on all cores. Refer to the readme.txt for more details.

7. Ensure that the correct repositories are selected and then select the IPC platform corresponding to your device/core. I.e. for c6474:


New CCS Project

RTSC Configuration Settings

Select the RTSC Configuration project settings.

XDCtools version: 3.20.00.39.eng

Products and Repositories | Order

- [✓] Inter-processor Communication
  - [✓] 1.21.00.12.eng
- [✓] SYS/BIOS
  - [✓] 6.30.00.26.eng

RTSC Target: ti.targets.C64P

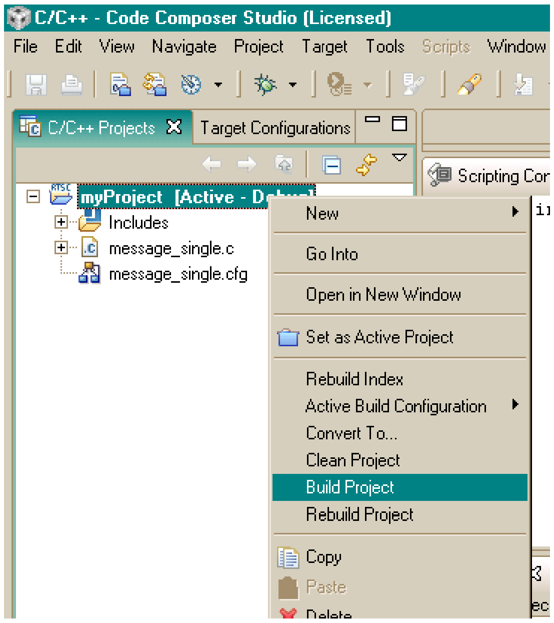
RTSC Platform: ti.sdo.ipc.examples.platforms.evm6474.core0

RTSC Build-Profile: whole_program

*NOTE: The available build profiles are **debug**, **whole_program**, and **whole_program_debug**. The default is whole_program. The "debug" profile allows you to step through the sources if sources are available. The "whole_program" profile has negligible debug symbols. The "whole_program_debug" profile has more debug symbols and improves the debug experience when compared to "whole_program". It is recommended that production code be built with whole_program or whole_program_debug.*

8. Click 'Finish' and you will see your newly created project in the 'C/C++ Projects' pane.

9. You can build your project immediately by right clicking 'myProject' and clicking 'Build'



C. Building IPC examples outside CCSv4

Building Single-core Examples

The single core examples can be built in two different ways. The first is via the "xdc" command. This approach uses the config.bld described in the above section.

1. Change directory to the desired directory
```
% cd <ipc_install>/ipc_1_22_xx_xx/packages/ti/sdo/ipc/
  examples/singlecore
```

2. Run the "xdc" command
```
% xdc
```

The other way is via the CCS templates. Refer to Section C (Building IPC examples within CCSv4) for more details.

The following are some basic single-core examples that ship with IPC:

ti\sdo\io\examples

- stream: Example of the ti.sdo.io.Stream module

ti\sdo\ipc\examples\singlecore

- notify_loopback: Example of the ti.sdo.ipc.Notify module using loopback functionality (which allows it to run on a single core).
- message: Example of the ti.sdo.ipc.MessageQ module on a single core.

Building Multi-core Examples

The multi-core examples can be built in two different ways. The first is via the "xdc" command. This approach uses the config.bld described in the above section.

1. Change directory to the desired directory

```
% cd <ipc_install>/ipc_1_22_xx_xx
/packages/ti/sdo/ipc/examples/multicore/c6474
```

2. Run the “xdc” command

```
% xdc
```

NOTE: Another way to build examples is by using the CCS templates. Refer to Section C (Building IPC examples within CCSv4) for more information.

Each platform example directory contains a multiprocessor Notify and MessageQ example. Refer to the readme.txt files in the respective directories for more details.

Rebuilding IPC modules

The individual modules can be rebuilt also via the “xdc” command. For example, to rebuild the utils modules:

1. Change directory to the desired directory (e.g. utils)

```
% cd <ipc_install>/ipc_1_22_xx_xx/packages/ti/sdo/utils
```

2. Run the “xdc” command

```
% xdc
```