jancumps
February 28, 2017 14:48

Debug IOT2020 C++ Program with Eclipse Neon and Windows10 64-bit

The standard Remote Debug doesn't work for me in Eclipse Neon. It's a known situation (https://support.industry.siemens.com/tf/ww/en/posts/how-to-debug-iot2040-by-eclipse-c-c-code/160821/?page=0&pageSize=10).
I have a process to get it working though. It's not as streamlined, but you have the full debug powers.

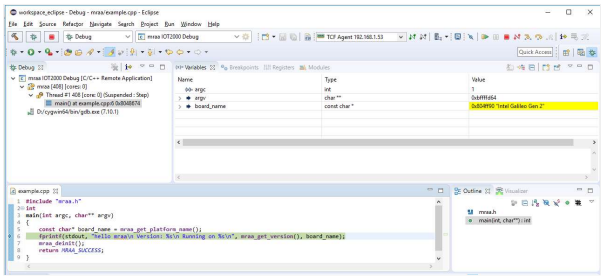
What works:

- stepping through code
- viewing variables & expressions
- the typical things you do in a debugger.

What doesn't work (yet?):

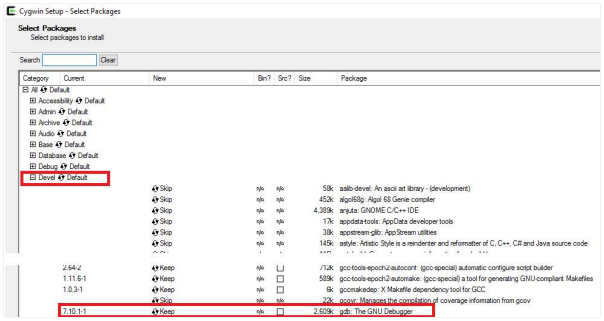
- uploading the binary to the IOT2000 when you start the debug session
- automatic start of the GDB server on the linux side
- redirecting output to the Eclipse console
- automatically find the correct path of every source file when stepping trough code

None of the "doesn't work" topics is a showstopper. There's a replacement technique that covers each of them.



Additional software

You'll need to have the Cygwin version of the GDB executable installed on your Windows PC. The poky version in the Siemens SDK (and the mingw and linaro versions that I tried) don't work well together either with Windows10 64-bits or the GDB server on the IOT2000. This GNU Debugger isn't installed by default when you install Cygwin. Select it during the installation. It's available under the Devel tree:

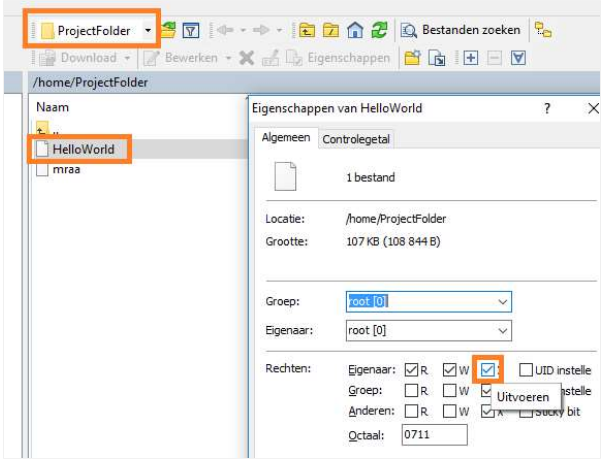


Change Skip to Install to get it as part of Cygwin.

You also need WinSCP or another means to transfer files to the IOT2020.
That's all there is to install. Let's go over the debug cycle now.
I'm assuming that you have the SDK example from Siemens loaded and built. Let's go trough the steps.

Move the binary to IOT2020

The default GDB debugger configuration does that for you. It also sets the executable flag. This fails when using Eclipse Neon on the IOT2020. We'll use WinSCP to do both tasks.



Start WinSCP, enter ip and credentials of your IOT2020 and connect. Navigate to the project folder of your development PC and find your compiled binary (in the Release folder of your project). Even though the project says it's a Release version, the Siemens SDK sets the **-g3** option of the compiler. That means that all symbolic info needed for a debug session is available in the binary. On the right side, navigate to your Linux ProjectFolder directory. Copy the file over (if asked select binary mode).
When copying is finished, right click on the destination file and select properties. Set the executable flag.

Start the Remote Debugger

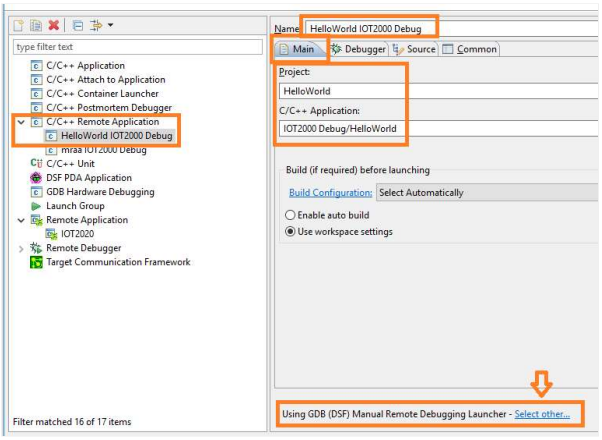
Open an IOT2020 Linux shell (via PuTTY or your personal favourite serial comms program). Navigate to the ProjectFolder directory and execute the following command:

> gdbserver localhost:2000 ./HelloWorld

The debug server will patiently wait until you start your debug session in Eclipse on your PC.

Create a Debug Configuration in Eclipse Neon

The last step. We'll set up the client side and launch the debugger.



You create this via the Run -> Debug Configurations... menu
Create a new Remote Application debug configuration (either by right-clicking on the parent node or using the New button at the top of the dialog.
Give the config a meaningful name, end select the project (use the Browse... button for that).
You can let Eclipse fill in the Application by pressing the Search Project... button. At the bottom of the dialog, **change the GDB Launcher from Automatic to Manual Remote Debugging**.