VECTOR

VectorCAST RSP QNX 7

Version 1.0

2022-03-15

Application Note AN-ACT-1-101

Author Derek Opitz

Restrictions Public Document

Abstract Information on the VectorCAST RSP for QNX 7

Table of Contents

1 Overview

VectorCAST Runtime Support Package (RSP) is an extension of the VectorCAST product. It provides an interface layer that allows you to use the VectorCAST testing techniques and methods on an embedded target processor. The VectorCAST RSP will always run on your host platform (the same platform as the compiler). Only the VectorCAST generated test harness will be downloaded to, and executed on, the embedded target.

A VectorCAST RSP is always customized to a particular Target CPU, Cross Compiler, and Run-Time environment (or kernel). As such, the information presented here contains sections for this specific RSP.

This application note provides details on configuring VectorCAST to run tests on a QNX 7.x target processor. The target can be a live board or a simulated board (such as QEMU).

VectorCAST uses the QNX compiler and linker to build the VectorCAST test harness into an QNX application. The application is then transferred to the QNX target for execution using either the SSH protocol or a GDB debug connection.

Once the application is on the target, VectorCAST uses the SSH protocol or the GDB debug connection to start the application and collect the test results.

VectorCAST provides off the shelf QNX 7.x RSPs for three architectures: ARM, ARM64 and x86-64. If you need support for a different architecture or run into any issues configuring VectorCAST, please contact support@vector.com.

2 Use Cases

2.1 Use Case: Selecting the Proper Configuration

The first thing you need to do when configuring your QNX RSP is to select the architecture (ARM, ARM64 or X86_64) as seen in Figure 1

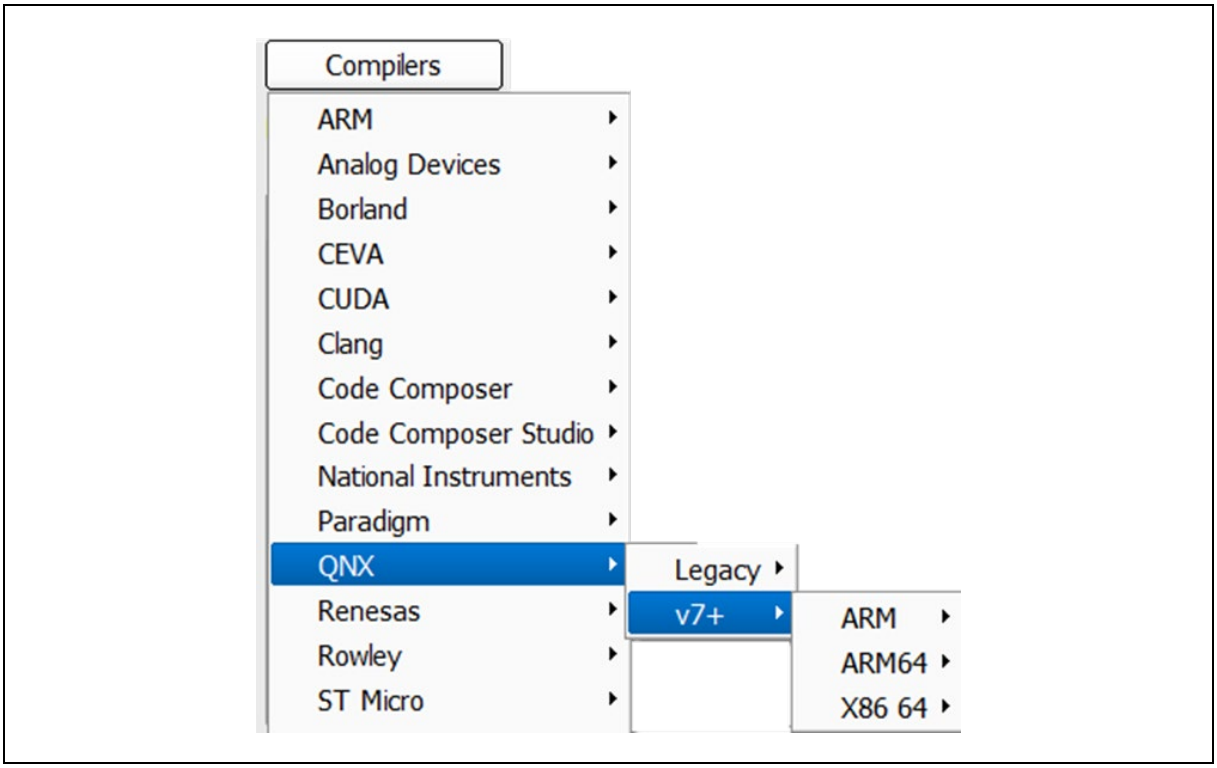

Figure 1 – Selecting the Configuration

The next thing you need select is the type of connection to the QNX target (GDB or SSH) as shown in Figure 2. GDB does not require any configuration and is highly recommended. SSH is supported but will require you to configure your QNX kernel for ssh, setup any logins required, and setup your ssh key.

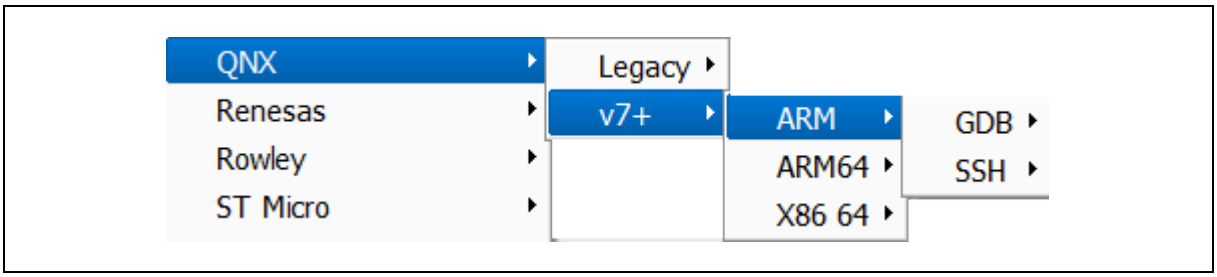

Figure 2 – Selecting the Connection Type

The last thing you need to select is what type of code you want to test as shown in Figure 3. If you are just testing C code then choose C, if you are testing C++ or a combination of C and C++ then choose C++.

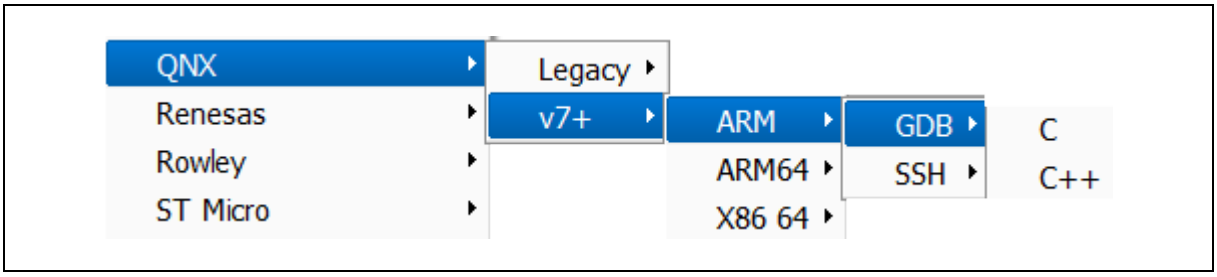

Figure 3 – Selecting the Language

Once you have selected either C or C++, the configuration will show up in your project and the compiler node will be populated with the proper data as shown in Figure 4

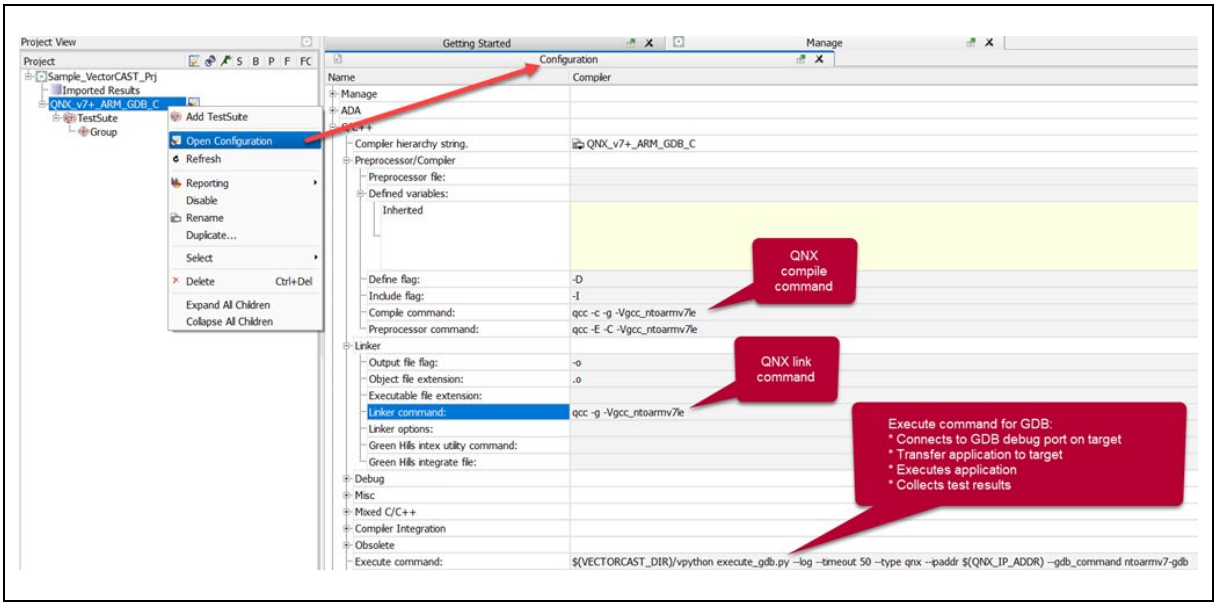

Figure 4 – Configuration Complete

2.2 Setup

The QNX development environment must be setup before launching VectorCAST so that VectorCAST has access to the compiler, linker, debugger, etc. The development environment for QNX is enabled using a shell script provided by QNX called qnxsdp-env.bat for Windows and qnxsdp-env.sh for Linux. The user will need to call this script from the command line before launching VectorCAST.

VectorCAST will also need to know the IP address of the target board you are wanting to test on. This is provided using an environment variable (QNX_IP_ADDR).

VectorCAST uses QNX Momentics to debug the test harness and requires a path to the QNX Momentics IDE to be added to the path environment variable.

If you are using SSH then VectorCAST will also need to know the directory on the QNX target to run the tests. This directory is provided using an environment variable (QNX_WORKING_DIR).

We recommend that you put the calling of the QNX environment script, the path to the Momentics IDE, and the environment variables into a startup script called setup_VectorCAST.bat or setup_VectorCAST.sh depending on if you are on Windows or Linux.

Your setup_VectorCAST.bat script will end up looking something like this:

```
call D:\qnx710_2\qnxsdp-env.bat
set path=D:\QNX\QNX_Momentics_IDE;%path%
set QNX_IP_ADDR=192.168.153.130
```

If you are using SSH, then you will need an additional line for the working directory:

```
set QNX_WORKING_DIR=/tmp
```

The setup_VectorCAST.bat script will need to be called before execution of any VectorCAST CLI commands (such as %VECTORCAST_DIR%\manage or %VECTORCAST_DIR%\clicast).

The setup_VectorCAST.bat script will also need to be called before launching the VectorCAST GUI. We recommend that you create another script called Start_VectorCAST.bat that will call setup_VectorCAST.bat and then launch the VectorCAST GUI.

Your start_VectorCAST.bat will look like this:

```
call %~dp0setup_VectorCAST.bat
start %VECTORCAST_DIR%\vcastqt
```

The %~dp0 is a windows batch syntax that will expand the directory path so that setup_VectorCAST.bat will be called from the same directory that start_VectorCAST.bat exists in. This allows you to create a shortcut to start_VectorCAST.bat and if setup_VectorCAST.bat is in the same directory as start_VectorCAST.bat then the shortcut will work.

You will need to use start_VectorCAST.bat to launch VectorCAST or you will not be able to build or execute unit test environments.

2.3 Debugging

Debugging VectorCAST testcases is done using the QNX Momentics IDE. If you would prefer to use command line gdb for debugging, then change IDE to CLI in the Debugger command and VectorCAST will use the command line instead of QNX Momentics IDE.

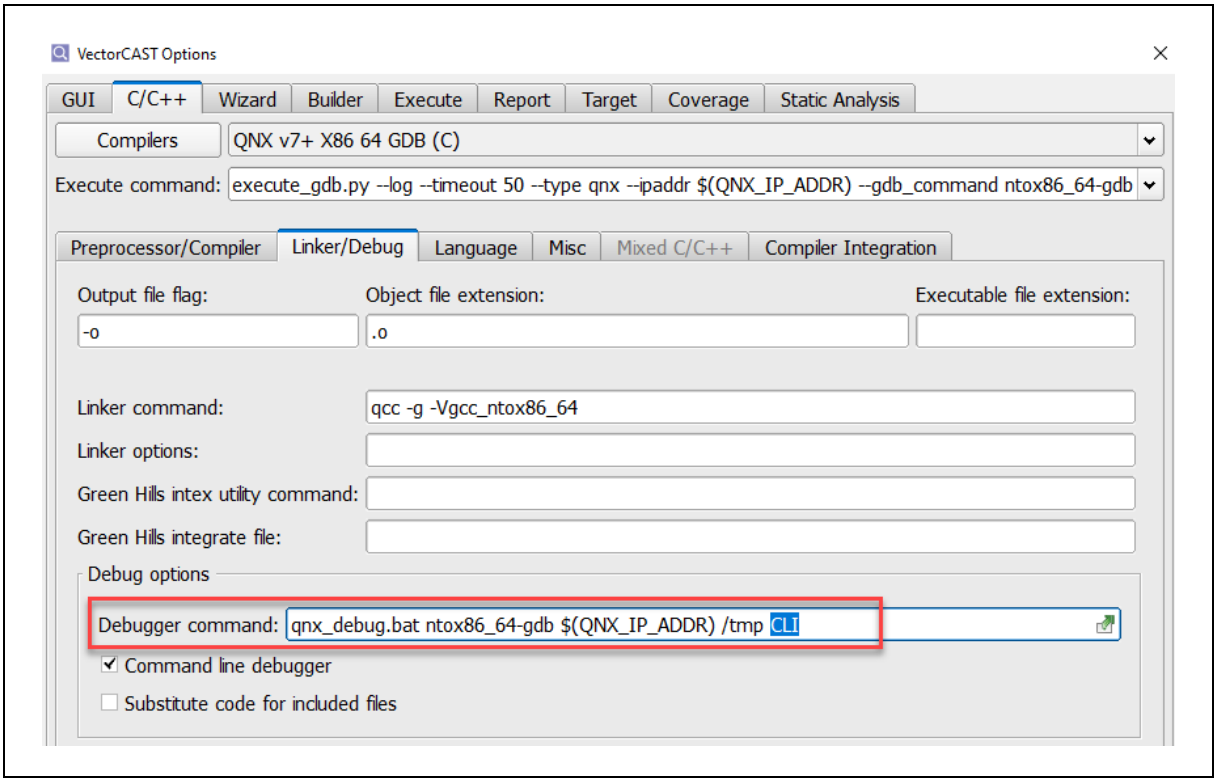

Figure 5 – Setting the Debugger command Argument

The following steps will describe the process for debugging a testcase from QNX Momentics IDE.

When you right click on a testcase and select ‘Execute with Debug’ the first thing VectorCAST will do is create a debug.ini script. This is a gdb script that will be used from Momentics to start the debug instance. After creation of the debug.ini script, VectorCAST will launch Momentics.

Each user of VectorCAST will need to setup a VectorCAST debug configuration in Momentics. This is a one-time activity per user and should not need to be repeated.

We will describe the steps to setup a debug configuration below, but if you would prefer a Vector YouTube video has been created that will walk you through the steps of setting up and using the VectorCAST debug configuration:
https://youtu.be/4BbJFSy7rCo

Once Momentics has been launched go to Run-> Debug Configurations… and create a new configuration called vectorcast_debug under C/C++ QNX Application as shown below.

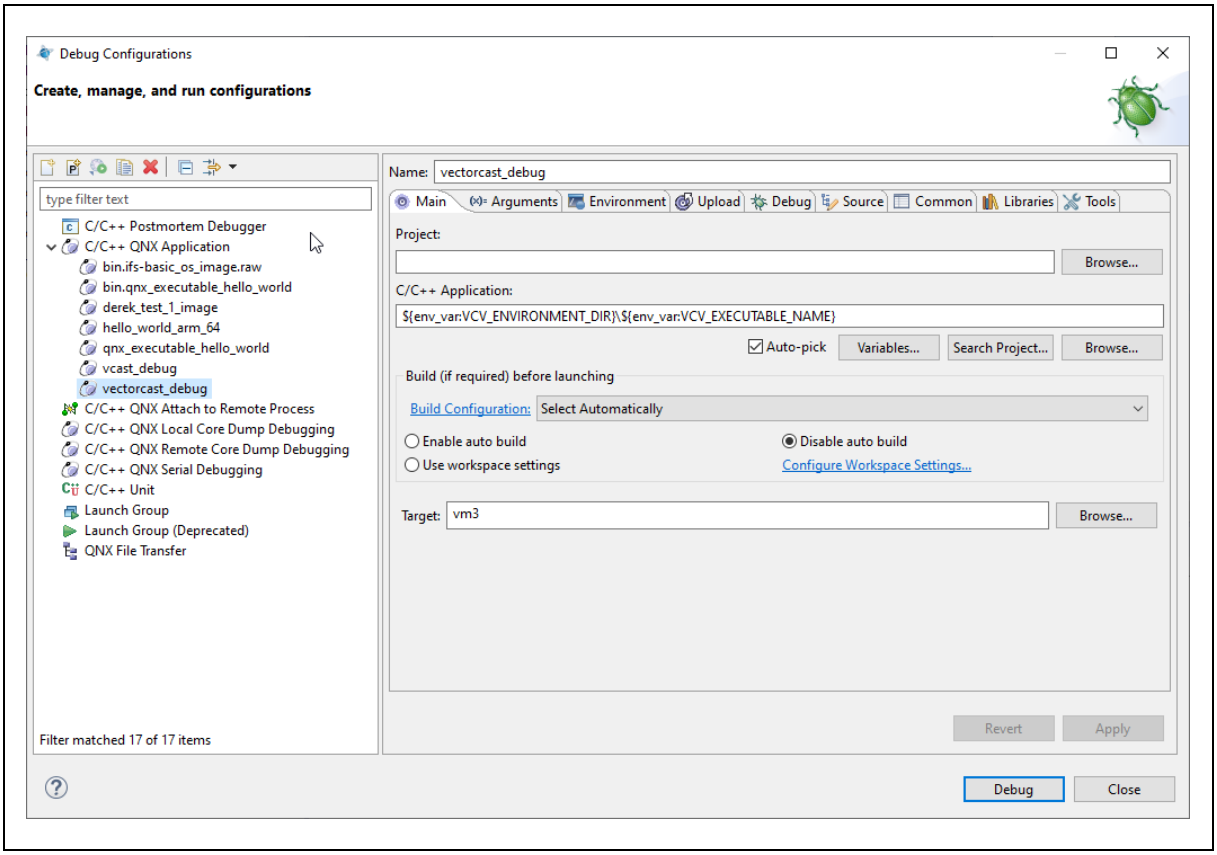

Figure 6 – Creating a New Debug Configuration

As shown in the screenshot above, set the C/C++ Application to:

${env_var:VCV_ENVIRONMENT_DIR}\${env_var:VCV_EXECUTABLE_NAME}

This is Eclipse syntax for using environment variables. These two environment variables are set by VectorCAST so that the debug configuration can be reused without modification for any VectorCAST unit test environment.

Next, select the Debug tab and fill in the fields for GDB debugger and GDB command file.

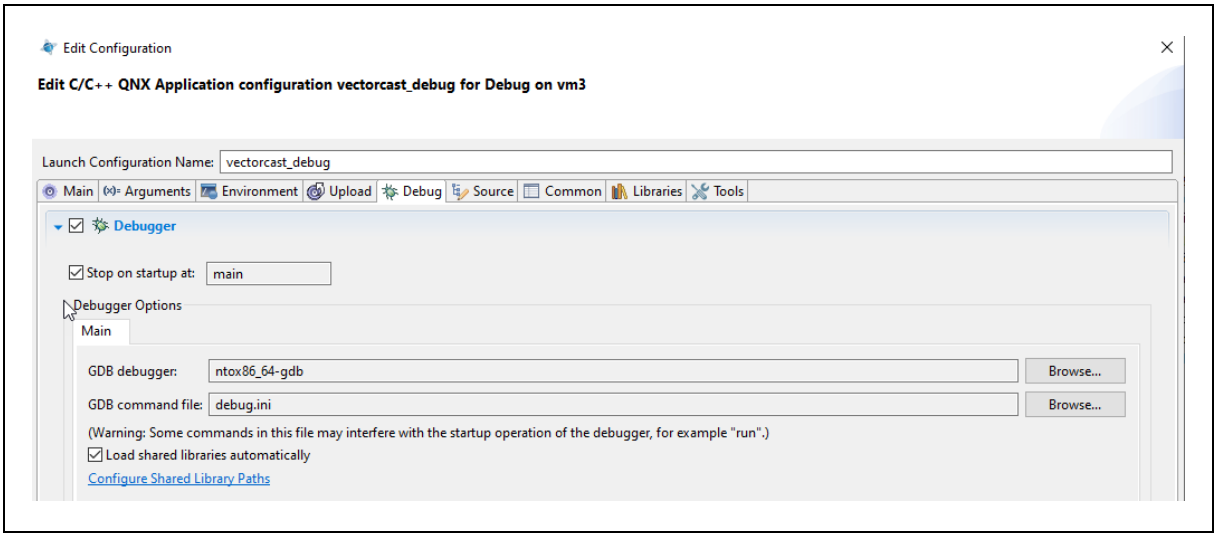

Figure 7 – Update the GDB Debugger Options

The GDB command file will always be debug.ini.

The GDB debugger entry is dependent on the architecture. It will be ntox86_64-gdb for x86_64, ntoarmv7-gdb for ARM, and ntoaarch64-gdb for ARM64. The exact GDB debugger you need to use can be found in the debug line for VectorCAST (Tools->Options…, then select Linker/Debug tab):

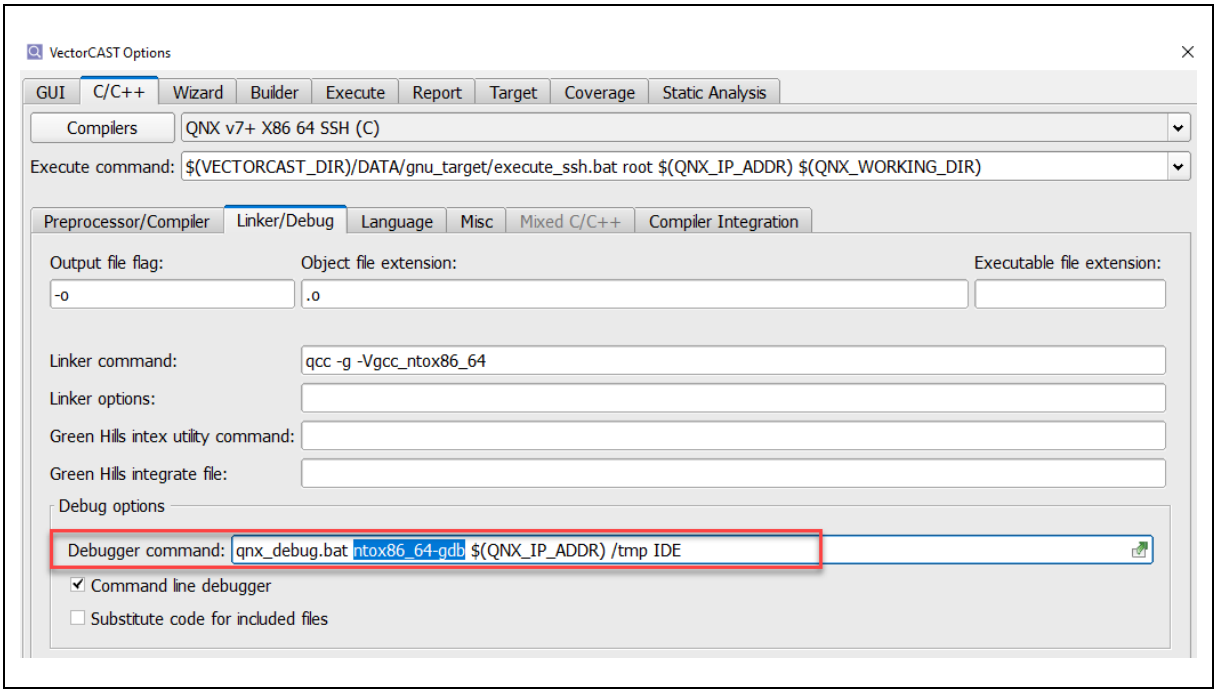

Figure 8 – Update the GDB Debugger Architecture

At this point you can save the configuration and launch the debugger using the selected configuration. When you start debugging, execution will stop on main in the VectorCAST test harness. A breakpoint will be set on the function under test, so if you click go execution should stop on the function under test and be able to debug your code. For more details, see the YouTube video referenced above.

3 Additional Resources

Blackberry QNX

4 Trademarks

All mentioned names are either registered or unregistered trademarks of their respective owners.

5 Contacts

For a full list with all Vector locations and addresses worldwide, please visit http://vector.com/contact/.