Unit Testing with Parasoft C/C++ Test and Keil uVision

Part 1

Introduction, problem statement. What is it that we are trying to achieve, what the current issues or roadblock?

1). Previous examples provided by Parasoft provided support for Keil uVision legacy project files imports (.uv and .uvproj) and there wasn’t enough information regarding the newer versions (.uvprojx).

2). Older, legacy versions of Keil uVision handled the Simulator (ITM) and regarding components manually, similar to how they handled other package components (CMSIS and Device specifics)

The two items above are described in the provided Parasoft with Keil uVision for MDK-ARM guide [old example is available within Parasoft install]

This document applies for Keil uVision and MDK-ARM, Cortex M4, specifically the nrf52 series of processors from Nordic

With the latest versions of Parasoft C/C++ Test, v10.4.1 and Keil uVsion v5.26.2. The previously documented process and flow to execute Unit Tests under Keil uVision Simulator (ITM) do not work properly. This is because of how Keil uVision handles the necessary compiler settings through its package management.

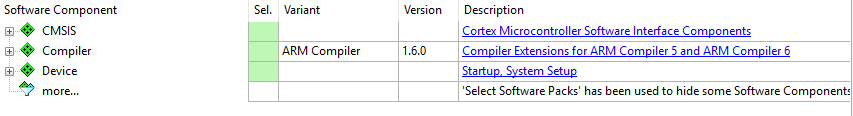
Part 2:

setup for keil uVision

Given that Keil uVision is already installed per instructions and all necessary packages are configured. (not part of the scope for this document).

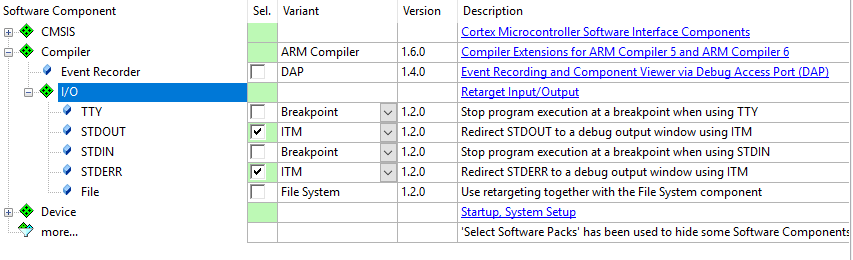
Launch Keil uVision.

In Keil uVsion click on the Green Diamond to open up the Manage Run-Time Environment dialog



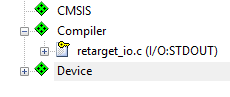
Expand the Compiler component and then the I/O sub-component.

Select STDOUT Variant and change it to ITM. Select the STDERR Variant and change it to ITM.



Press Ok for the Package manager to resolve and update.

Project structure shows the additional packages included in latest version of Keil uVision. The custom compiler setting (ARMCC) is added with retarget\_io.c for STDOUT



Enable retargeting, ITM debugging (optional) and compiler out hardware specifics in system\_nrf52.c (startup)

1). To enable the retargeting, add the RETARGET\_SYS to the Defines in the C/C++ tab under Options for Target…

2). To enable ITM debugging, add the DBG\_ITM to the Defines in the C/C++ tab under Options for Target…

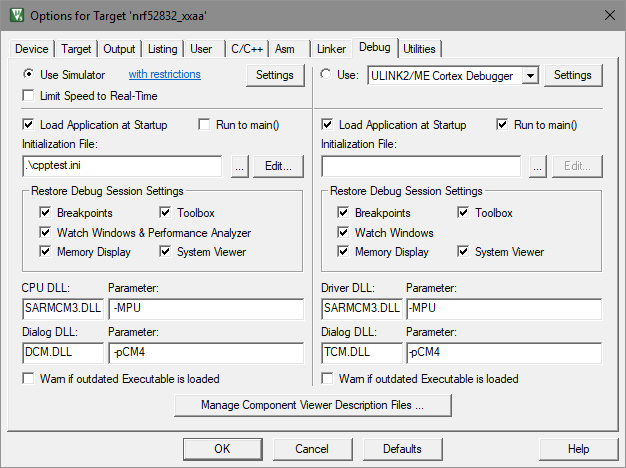
Since Keil uVision is running in simulator mode, there is not need to execute hardware startup routine, especially when they access memory or register locations.

3). Create a new Define, say UNIT\_TEST\_PARASOFT and add it to the Defines in the C/C++ under Options for Target…

Now open the device specific system\_nrf52.c and add the new macro around the errata declarations and definitions, the calls to said functions within the SystemInit function.

Ensure the project builds without any errors or warnings.

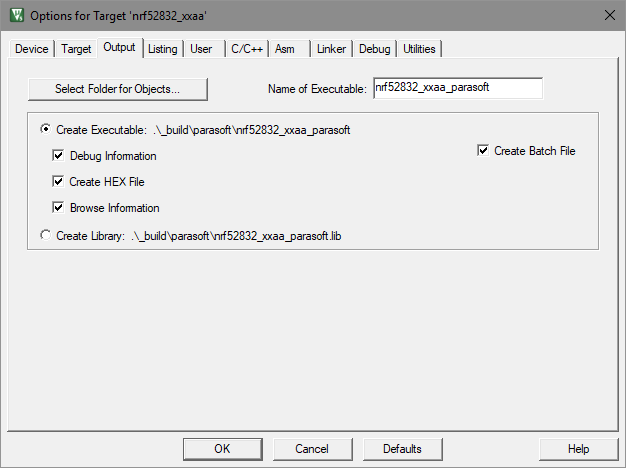
Enable the Simulator (ITM) in Keil uVision. Open the Options for Target… and go to the Debug tab (see below)



The cpptest.ini file is provided by the Parasoft C/C++ Test runtime engine and should be created into the project folder. This file is a script that contains how to execute the built image, collect test data and exit keil uVision upon completion.

Enable the batch file creation for Parasoft. Parasoft still uses the batch output from Keil uVision to understand the project build information and flags during the import process.

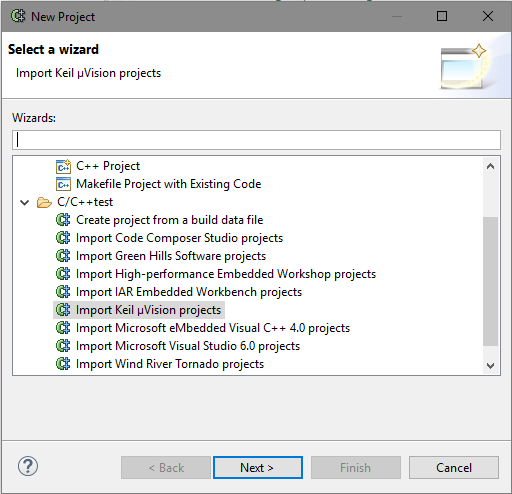
Open the Options for Target… go to Output tab and select (enable) the Create Batch file option



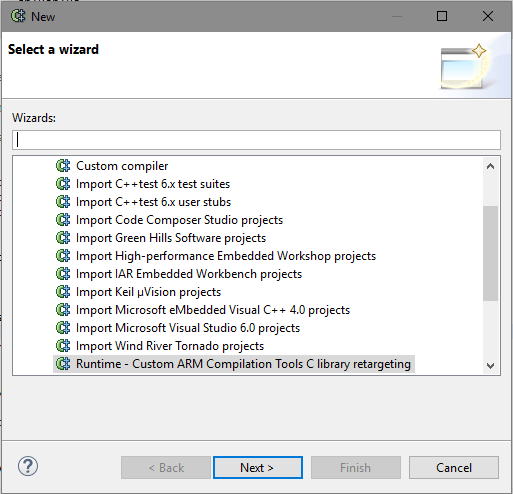
Part 3

Setup for Parasoft

Now we are ready to import the project into Parasoft using the existing method and process.



After creating the project, right-click on the project folder, select New, then Other… and select RunTime – Custom ARM Compilation Tools C library retargeting option.



Enter in the name (cpptest\_retgt.c is the default) and the location ([project]/factory is the default), File Type is C.

click Finish.

In the Project navigator you should see a factory folder with the retargeting source (cpptest\_retgt.c)



Open the file in Parasoft and edit the \_sys\_exit function as below

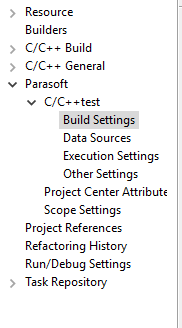
**void** **\_sys\_exit**(**int** return\_code)

{

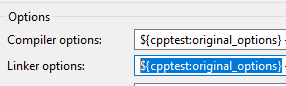
**for**(;;);

}

Go to Project properties, Parasoft C/C++ Test, Build Settings …



Add the following to the Linker Options, after ${cpptest:original\_options}



--Via [Path-to]\system\_nrf52.o" --Via "[Path-to]\retarget\_io.o"

Locate the CppTestTransportITM.c module under the Parasoft installed location (most likely C:\Program Files). Navigate to Parasoft\C++test\10.4\engine\runtime\src\transport location. Open the CppTestTransportITM.c and modify. replace cm3 dependencies with cm4.

#include "core\_cm4.h"

\_\_CM4\_CMSIS\_VERSION

Part 4: putting it together and wrap-up

Provided is a simple, yet complete example, specific to the ARMCC, Cortex M4 using nrf52.