Cortex-A & Cortex-R On-Chip Emulation

Contents

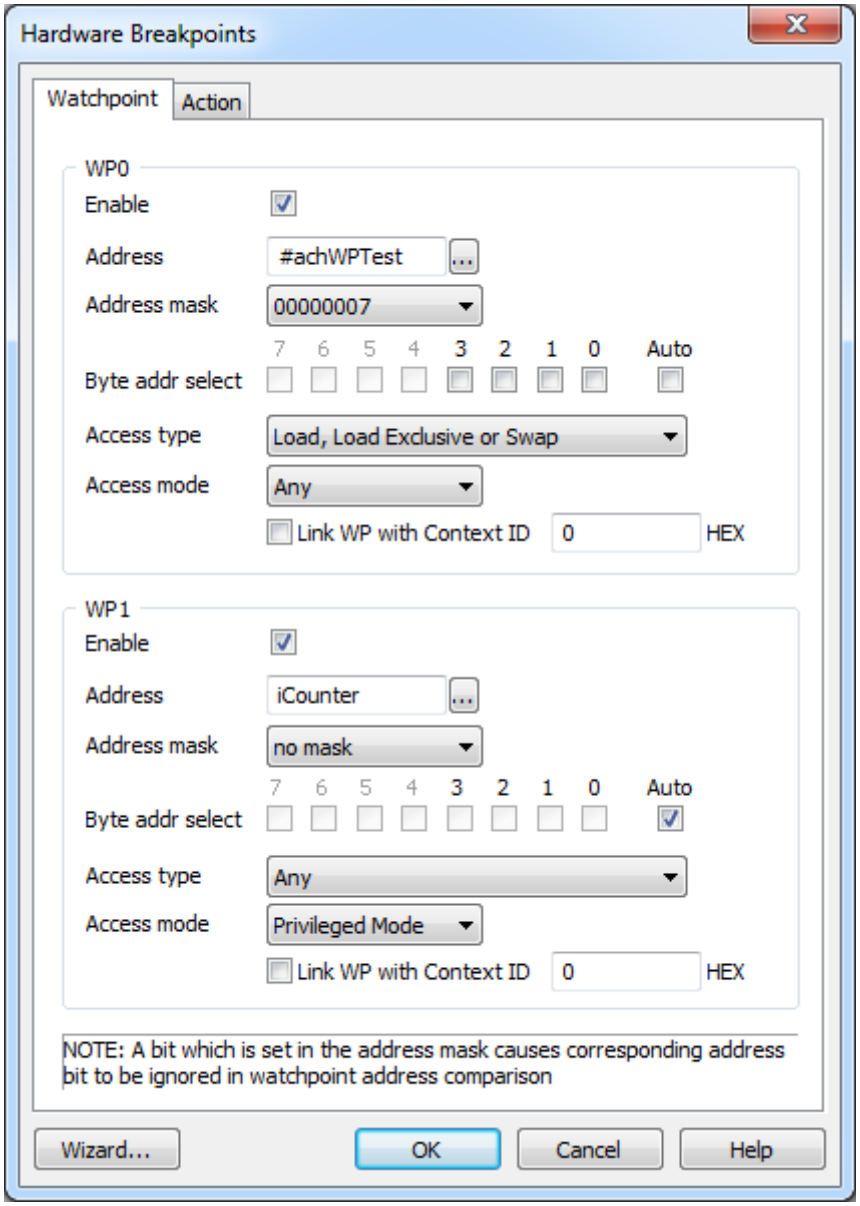
1 Introduction

This document describes only Cortex-A and Cortex-R specific winIDEA configuration and settings. Configuration and settings common to all: Cortex-M, Cortex-A and Cortex-R architectures are described in Cortex On-Chip Emulation section / document.

User is also encouraged to get familiar with documentation from ARM. Following are suggested documents:

- ARM®v7 Architecture Reference Manual
- Cortex™-R4 Technical Reference Manual
- Cortex™-A5 Technical Reference Manual
- Cortex™-A7 Technical Reference Manual
- Cortex™-A8 Technical Reference Manual
- Cortex™-A9 Technical Reference Manual
- Cortex™-A15 Technical Reference Manual
- ETMv3.5 Architecture Specification
- CoreSight™ Program Flow Trace™ Architecture Specification
- CoreSight™ PTM™-A9 Technical Reference Manual

2 Access Breakpoints



Cortex-A/R hardware breakpoints dialog

Address

Specify the address which is to be monitored for accesses by the target code.

Address mask

Address mask can be used in combination with the Address field to force some of the lower address bits to be ignored in data access monitoring. When “no mask” is used, all address bits are compared.

Byte addr select

The address value from the Address field is word-aligned before it is used to configure the comparator. The “Byte addr select” checkboxes are used to further select the monitored byte lanes within the word address. If none of the bytes is selected, then the watchpoint will never hit.

Auto

“Byte addr select” will be configured automatically internally when selected.

Access type

Specifies the type of accesses to be monitored by the watchpoint.

Access mode

Specifies the mode of CPU in which the watchpoint will be active.

3 Software breakpoints

Setting SW BPs on devices with advanced memory access system is not straight forward as only writing proper instruction at the address of wanted BP. BKPT instruction might stay in DCACHE and thus not presented to core fetch mechanism. Additionally ICACHE might not detect that BP instruction was written to underlying memory system. That is why SW BP to memory is set with following steps:

- Cores are stopped if running
- BKPT instruction is written
- DCACHE is cleaned
- ICACHE is invalidated for all cores
- Cores are run if were running before.

4 Virtual memory access

Additional memory mechanism is available on Cortex A and R devices. For contrast: physical memory access utilizes debug features to access memory bus directly. But when doing virtual access the core is used to load data from underlying memory system. It is done with writing load or store instruction to debug register prepared for this purpose. The procedure is called instruction stuffing. Advantage of this approach is that DCACHE and MMU configuration is irrelevant. With virtual access developer sees memory as core does. And this can be different from physical access which usually accesses memory below MMU and DCACHE. The amount of debug port messages is roughly the same for both types of access. Drawback of virtual access is that it can’t be used for real time access. This is because instruction stuffing is not allowed while core is running.

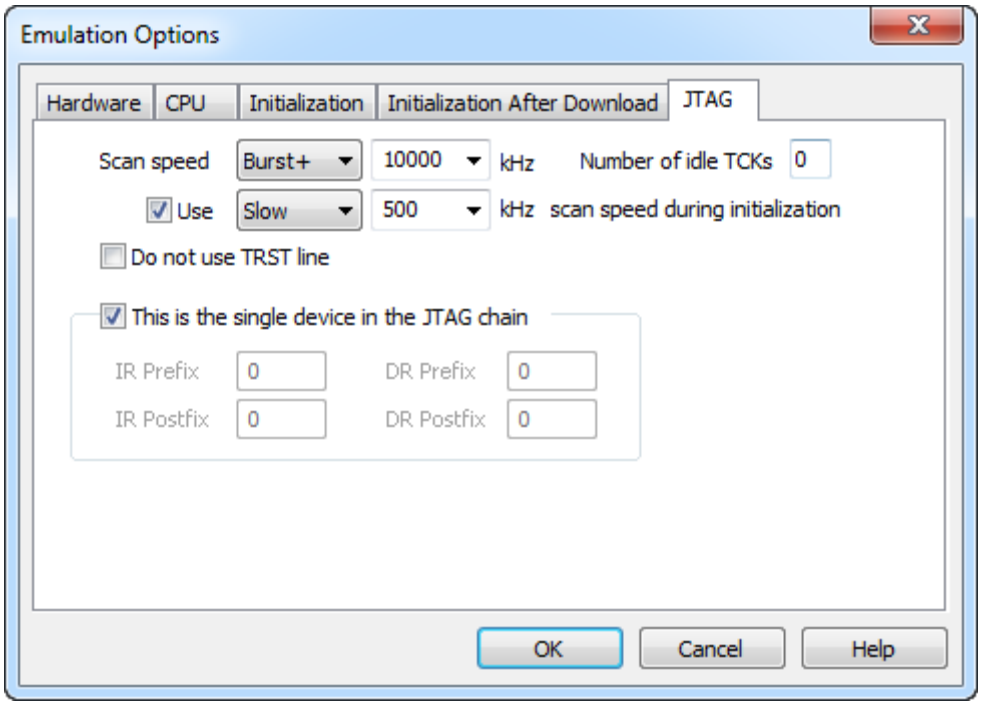
5 Burst+ Scan speed

Provides the ability to set the JTAG clock frequency from 4 MHz to 100 MHz (repetitive JTAG scan cycles are accelerated). Burst+ is used to increase speed of large memory write access.

For Burst+ to work correctly, target CPU must be able to handle the increased JTAG speed. Initialization script can be used to configure device’s PLLs and clocks. Maximum JTAG speed for specific device should be checked in device’s datasheet.

Number of Idle TCKs

Setting can be used if using Burst+ downloading works unreliably. If it doesn’t help, reduce selected Burst+ scan speed.



Emulation options, JTAG panel

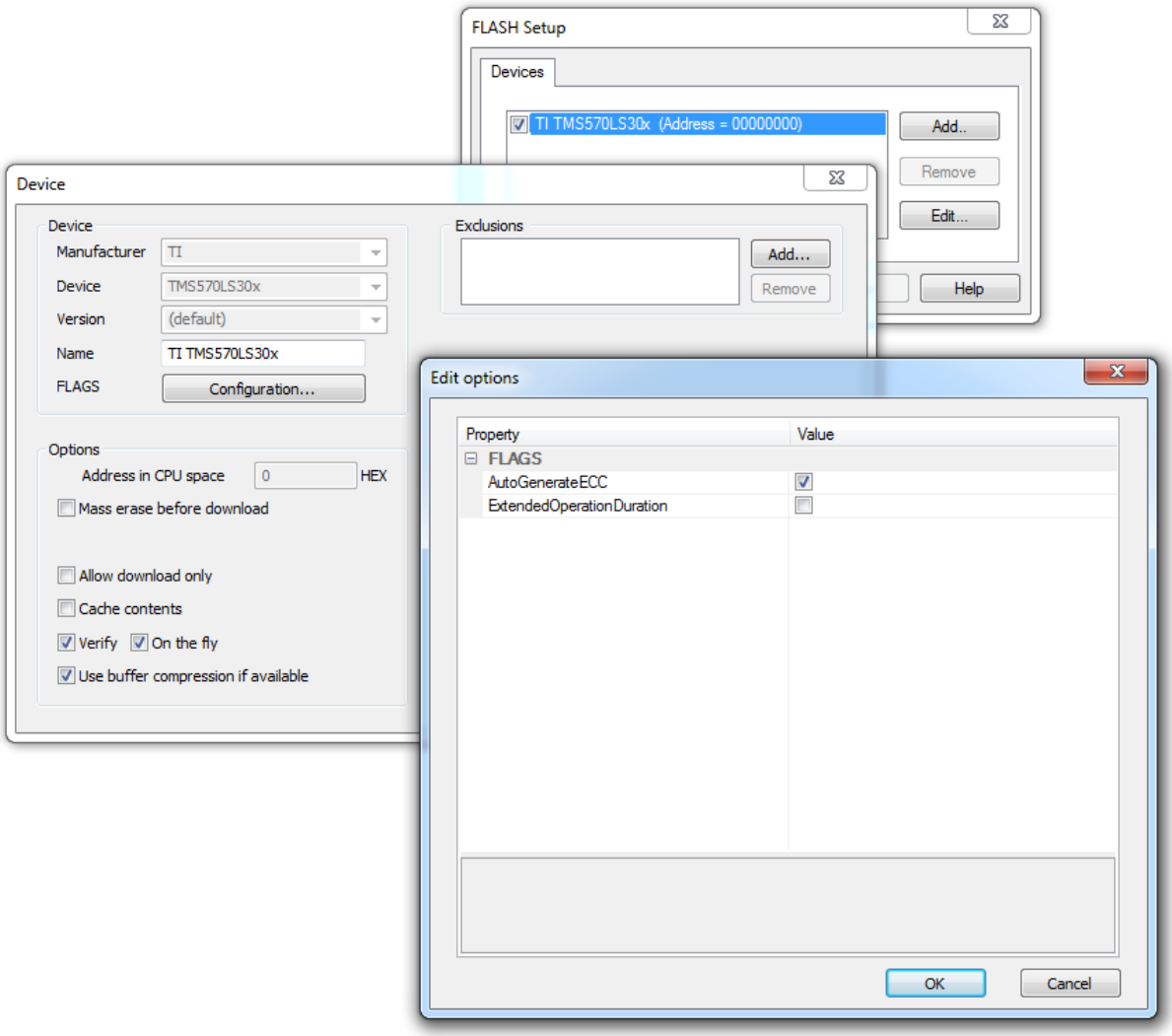
Note: This tab is disabled when debug interface other than JTAG is selected.

6 Texas Instruments TMS570

6.1 Internal Flash Programming

Flash on TMS570 devices features ECC functionality. ECC area can be programmed automatically with values corresponding to downloaded data. Or user can provide own ECC download data. Option for automatic ECC generation is located in Hardware\Flash Setup…\Edit…\Configuration…

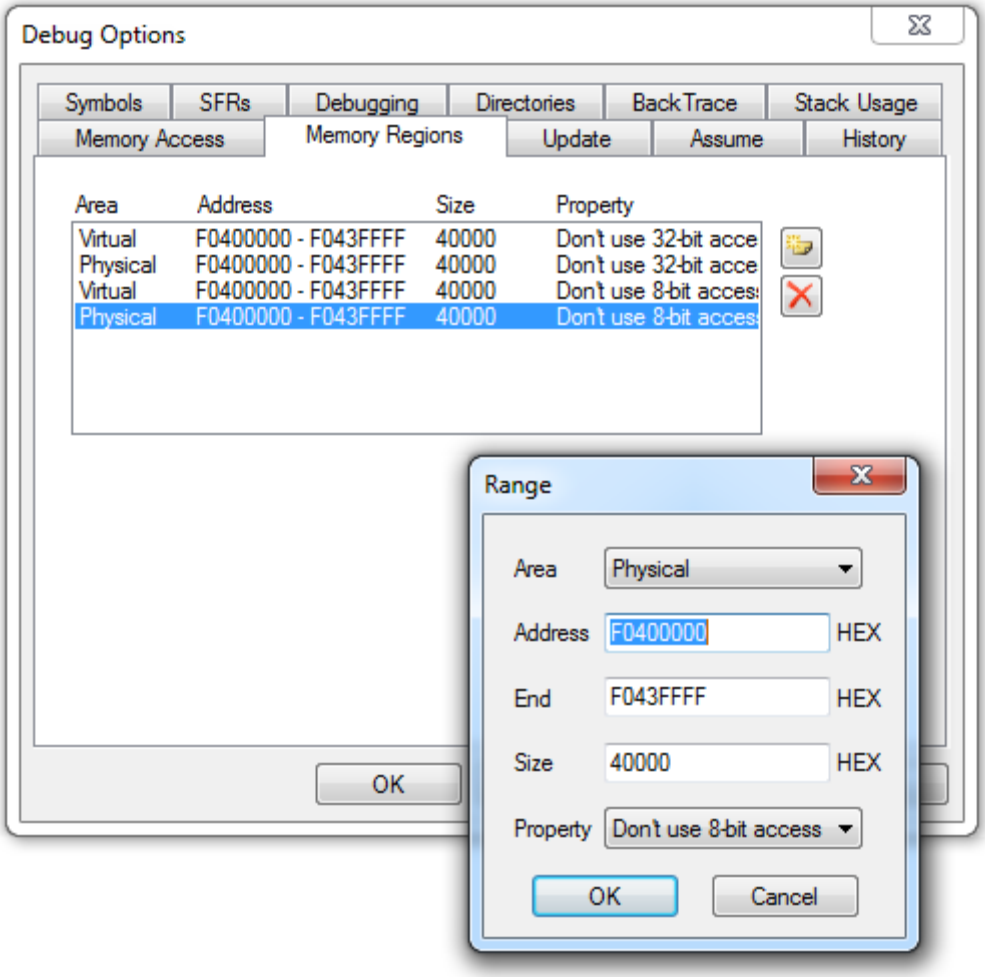
Additional **'ExtendedOperationDuration'** can be enabled in same dialog. It extends the timeout for flash operations on some devices.



TMS570 automatic ECC generation option

Automatic ECC generation is suggested for faster download. User can still provide own ECC download data and by enabling **'Verfy'** in Debug\"Files For Download…"\Options pane to ensure that data written to ECC area matches the downloaded data.

Reads from ECC area must be done with 16bit accesses. Thus 8bit and 32bit accesses should be disabled in Debug\"Debug Options…"\"Memory Regions" pane.



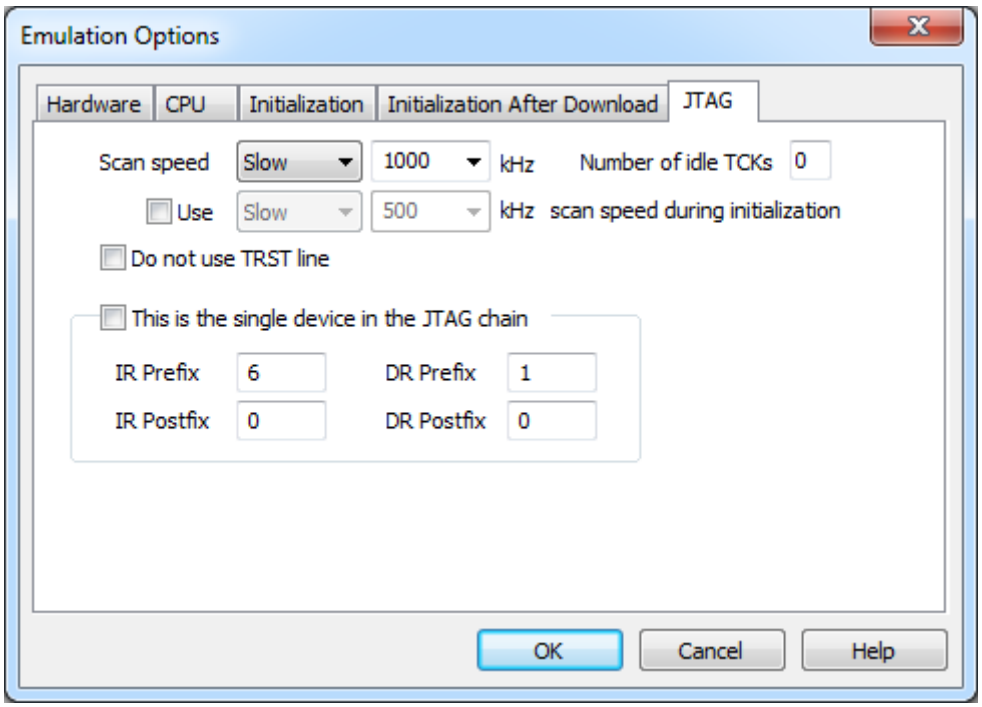
TMS570 ECC memory region configuration

Internal flash programming monitor is made with flash programming library from TI. The library must be initialized with correct CPU frequency for proper timings generation. The value for CPU frequency is taken from Hardware\"CPU Setup…"\Debugging\"CPU clock" setting. Note that this same value is used for download operations and for the memory writes through memory window during debug session. Use initialization sequence to set CPU clock to final value before download to be enable to make flash writes anytime.

7 Xilinx Zynq

7.1 JTAG Chain

JTAG chain is configurable within Zynq SoC. Note that only “Cascaded JTAG” works with empty device. Also note that user must provide soft core to connect external pins to ARM DAP if “Independent JTAG” mode is desired. Refer to “Mode Pin Settings” and “JTAG and DAP Subsystem” chapters in Zynq reference manual for more information.
As JTAG chain is configurable it has to be properly set in JTAG settings. Set JTAG prefixes as depicted below for “Cascaded JTAG”.



Zynq "Cascaded JTAG" mode configuration

7.2 Reset

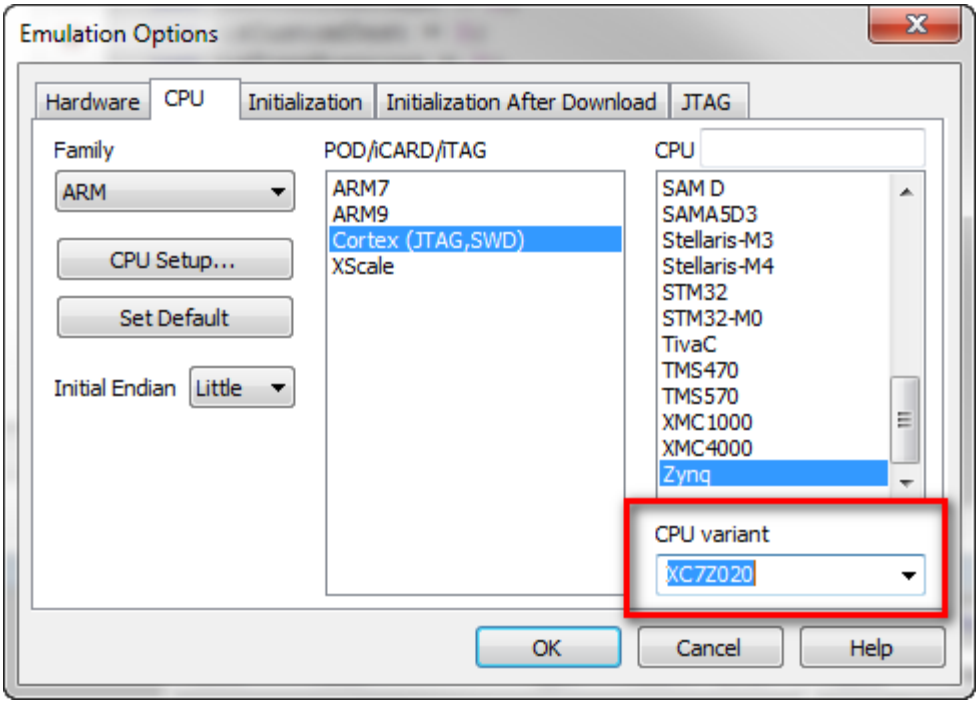
Stop and Preset reset method should be used when nSRST signal from debug connector is connected to device.

7.3 Bitstream download

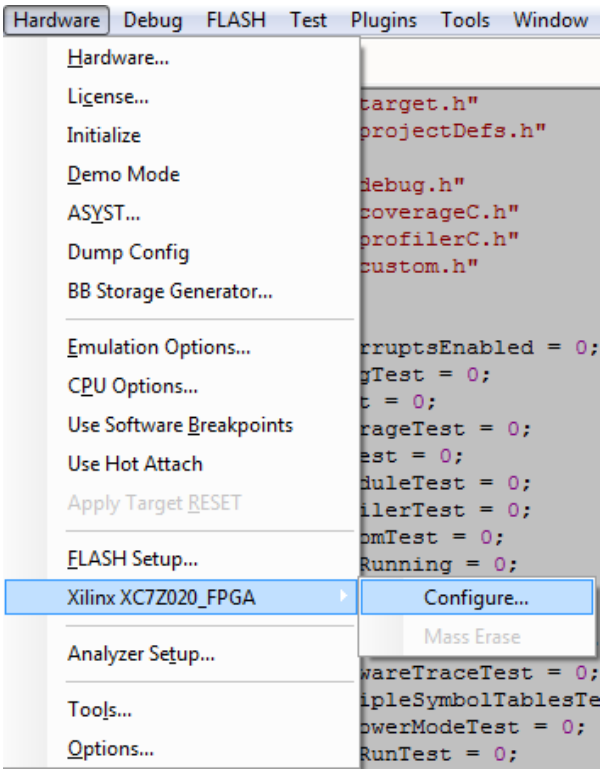
Note: Following example is created for Xilinx XC7Z020 target device.

7.3.1 Configuring Device

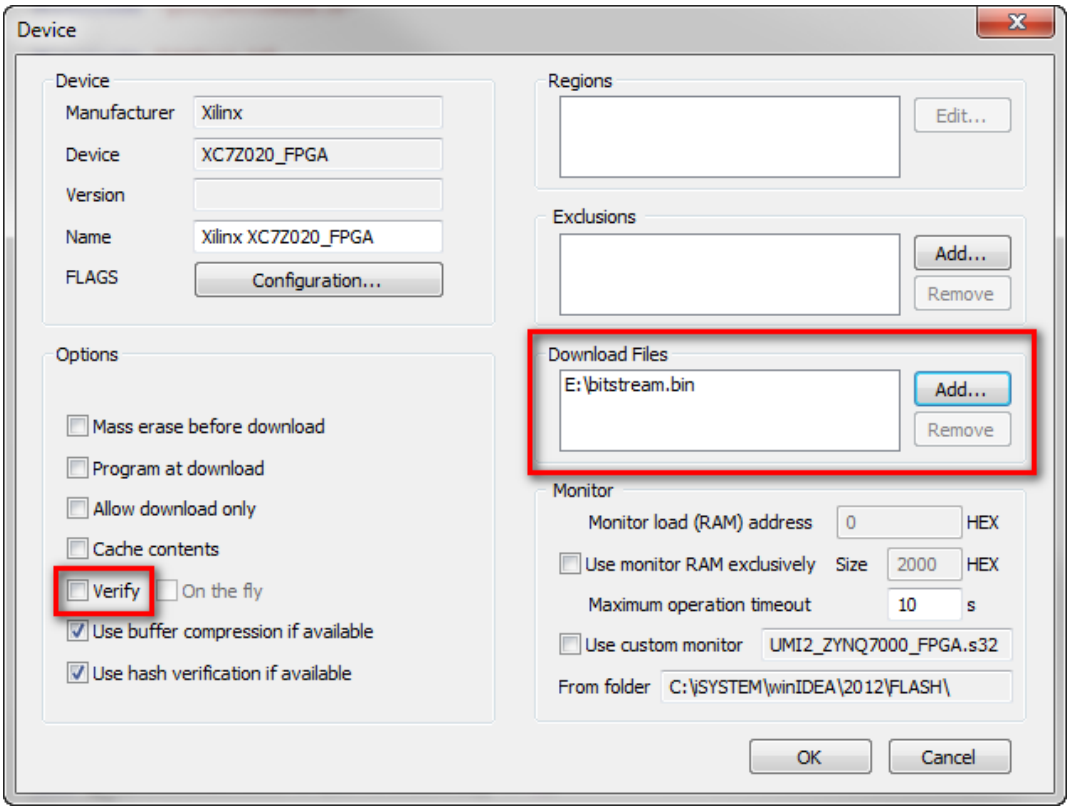
1. First make sure that the specific Zynq version is selected in the hardware / Emulation Options:



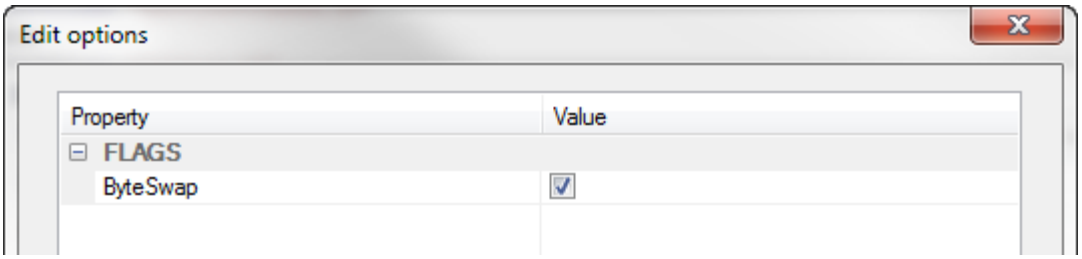
2. Selecting specific Zynq version will add the FPGA entry to the Hardware drop-down menu. Open the FPGA Configuration window:



3. Under *Download Files* add your prepared bitstream file and deselect the Verify option.

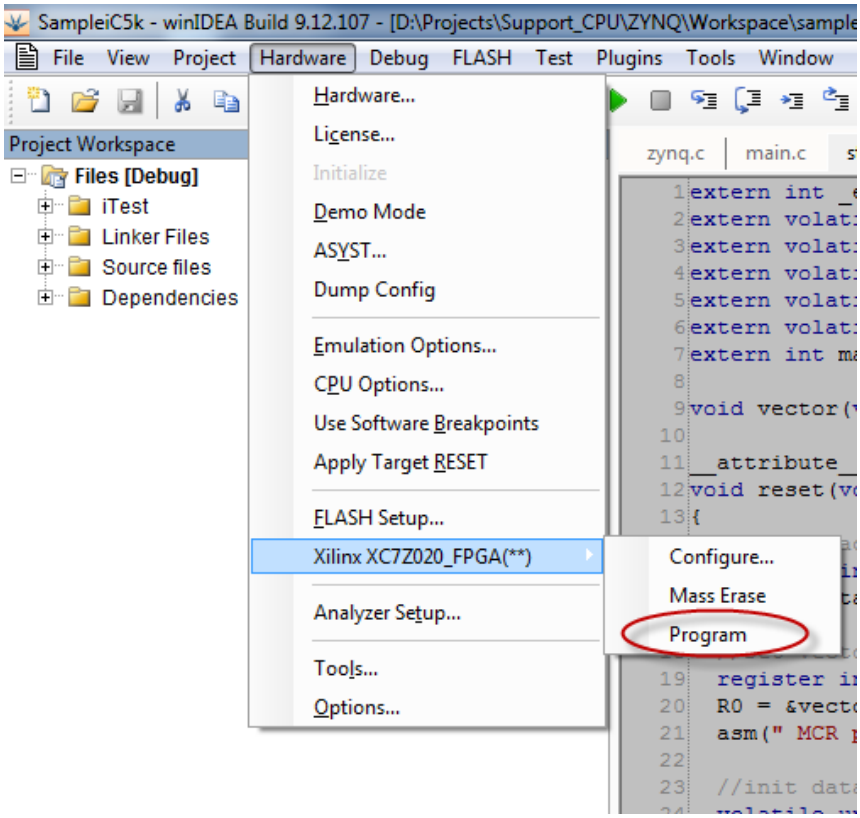


4. OPTIONAL: Default bitstream files are generated requiring byte swap. When manual conversion was done on bitstream file, this option can be disabled under *FLAGS* => *Configuration*….



7.3.2 Program

Downloading bitstream can now be initiated through *Hardware => Xilinx XC7Z020_FPGA => Program*.



7.3.3 Mass Erase

Mass Erase command will clear FPGA.