SCR1 SDK. Digilent Arty Edition. Quick Start Guide

© Syntacore, scr1@syntacore.com

Version 0.27, 2021-05-17

Table of Contents

Copyright notice

Revision history

Version	Date	Description
0.1	2017-09-09	Initial revision.
0.2	2017-09-25	Updated for Vivado 2018.1 compatibility
0.21	2019-01-30	Sections added: • Starting the OpenOCD; • OpenOCD: loading and running BIN and ELF images; • Windows - USB JTAG Cable drivers installation.
0.22	2019-03-21	OpenOCD section updated for RISC-V debug. JTAG speed requirement added.
0.23	2019-03-29	Fix figures numbers
0.24	2019-03-29	Updated IMPID, BUILDID, updated section FPGA Image Modification
0.25	2019-04-08	Added note for flash programming, fix figures numbers
0.26	2019-09-10	Updated IMPID, BUILDID
0.27	2021-05-17	Updated: SDK Memory Map, bootloader’s logo, FPGA binary files

Introduction

This is a brief user guide allowing to get started with SCR1 SDK based on Arty FPGA Development Board from Digilent.

It describes the board setup, procedure of software uploading and launching, and process of the FPGA's content building and updating.

1. Required Hardware

Minimal set of hardware needed for SCR1 SDK, Arty Edition, includes just the following components:

- **Digilent's Arty FPGA Development Board**
  https://reference.digilentinc.com/reference/programmable-logic/arty/start
- **Standard USB Type A (m) - Type B micro (m) cable**

The minimal setup is shown in Figure 1.

If you are going to debug software running on the SCR1 with GDB/OpenOCD, you need also

- **JTAG Cable Adapter:** Olimex ARM-USB-OCD-H (or ARM-USB-OCD)
  https://www.olimex.com/Products/ARM/JTAG/ARM-USB-OCD-H/
- **Standard USB Type A (m) - Type B (m) cable**
- **Wire Connection** between JTAG Cable Adapter and JD PMod connector on the Arty board. This could be done in different ways. Possible variants are
  - **Small prototype board** with connectors for standard 20-wire ARM JTAG flat ribbon cable, and for JD PMod. Example of this approach is shown below in Figure 2.
  - **Male-to-Female Jumper Wires.** The wires, e.g., might be of the following type:
    https://www.adafruit.com/product/826

2. SDK Setup

2.1. Board’s USB connection

Connect the **USB Type A (m) - Type B micro (m)** cable between Arty’s J10 (Shared USB JTAG/UART port) and your host computer. This connection performs three functions:

- +5V Power Supply for Arty
- FPGA Configuration JTAG port
- Serial Port (for console interface with running software)

USB connection of the Arty board is shown in Figure 1.

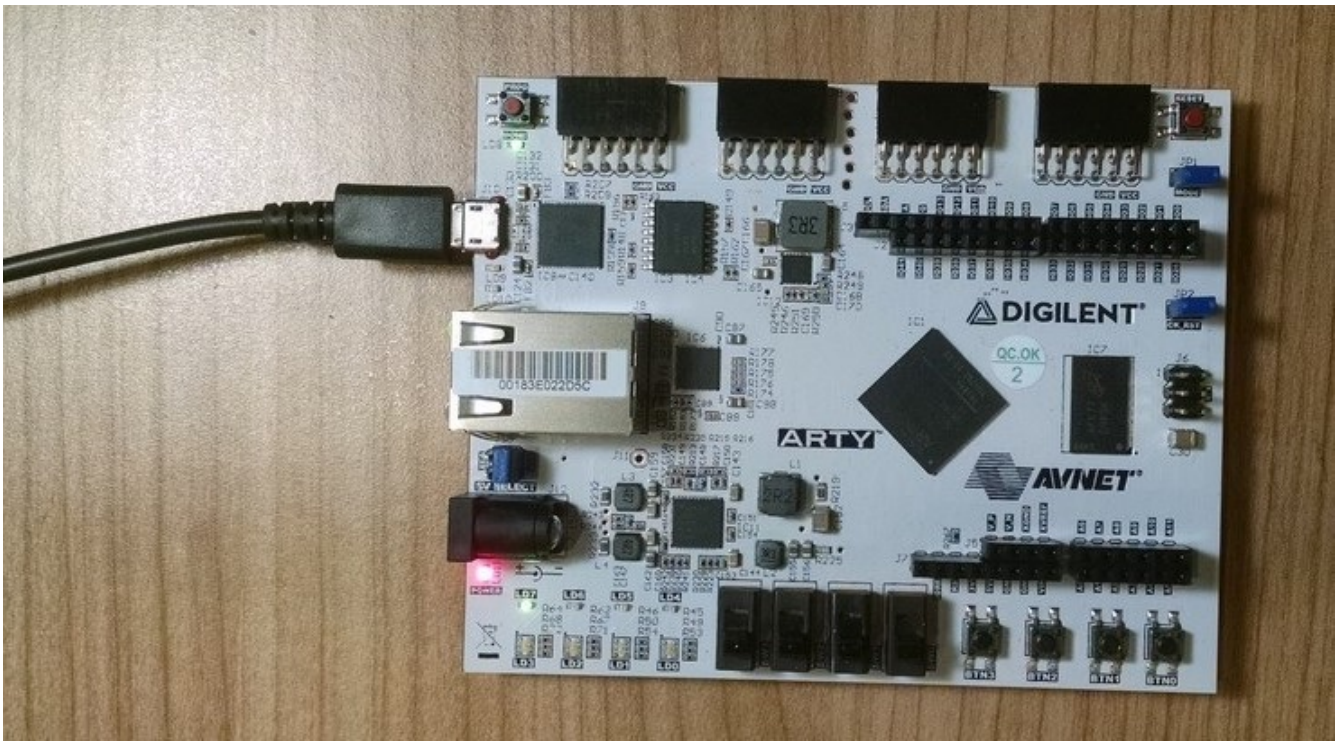

Figure 1. Arty Board with USB Connection

2.2. JTAG Cable Adapter connection

SCR1 JTAG connection on the Arty board is shown in Figure 2. For pin-out of the JTAG port refer to Appendix A. JTAG Pin-Out("Table 1").

IMPORTANT	For proper JTAG interface functioning JTAG clock (TckFreq) and system clock (SysClkFreq) frequencies must satisfy the following relation: SysClkFreq / TckFreq >= 12.

Figure 2. Arty Board with JTAG Cable Adapter

NOTE

Please, take into account that Olimex USB JTAG cable adapters after powering up (just after their USB cable connecting) hold debug connector's SRSTn line in asserted state, actively holding entire processor subsystem in reset state. To release SRSTn line, it is necessary to initialize cable adapter's HW. That could be done, for instance, by launching of OpenOCD configured for joint work with those cable adapters.

3. FPGA Configuration Flash Programming

3.1. Prerequisites

For FLASH programming any of the following software tools is necessary:

- **Xilinx Vivado Design Suite 2018.3 or newer**
- **Xilinx Vivado Lab Edition 2018.3 or newer**
- **Xilinx Vivado WebPack Edition 2018.3 or newer**

3.2. MCS-file

BIT-, MCS- and PRM- files are included into SDK repositories tree and could be obtained from here:

```
<SDK_HOME>/images/arty/scr1/scr1-arty-fpga.7z
```

Unpack this 7-Zip archive. After that you should get three files:

- arty_scr1_new.bit
- arty_scr1_new.mcs
- arty_scr1_new.prm

3.3. Procedure

IMPORTANT	Check that only one USB cable is connected to the board as shown in the minimal setup Figure 1.

1. Power-up the board and launch Vivado tool
2. Open Hardware Manager, then open appropriate target
3. In the FPGA device's context menu (right click) select "Add Configuration Memory Device"
4. In the menu select
    - Part: n25q128-3.3v
    - Manufacturer: Micron
    - Family: n25q
    - Type: SPI
    - Density: 128
    - Width: x1_x2_x4
5. On the question "Do you want to program the configuration memory device now?" - click OK
6. Add MCS and PRM files (see above: arty_scr1_new.mcs, arty_scr1_new.prm)

7. Click OK to start programming
8. After the programming completion, it is necessary to reload new image into FPGA. For that press the “PROG” Button on the Arty Board.

4. Getting SCR1 Running

4.1. Starting Up

1. Connect your host PC to the Arty's USB port (J10).
2. Open any terminal program. In the example below we use **minicom** terminal. Adjust its UART parameters as follows:
   - **Bps/Par/Bits** - 115200 8N1
   - **speed** - 115200
   - **bits** - 8
   - **stop bits** - 1
   - **parity** - none
   - **Hardware Flow Control:** No
3. Initiate FPGA re-configuration process (push **PROG** button) or SOC internal circuitry reset (push **RESET** button).
4. Terminal program should display SCR bootloader's banner and prompt:

```
SCR loader v1.2-scr1_RC
Copyright (C) 2015-2021 Syntacore. All rights reserved.
ISA: RV32IMC [40001104] IMPID: 21051400
SOCID: 21050500 BLDID: 21051402
Platform: arty_scr1, cpuclk 25MHz, sysclk 25MHz
Memory map:
00000000-0FFFFFFF       00000000        DDR
F0000000-F000FFFF       00000000        TCM
F0040000-F0040FFF       00000000        MTimer
FF000000-FF0FFFFF       00000000        MMIO
FFFF0000-FFFFFFFF       00000000        On-Chip RAM

1: xmodem load @addr
g: start @addr
d: dump mem
m: modify mem
i: platform info
:
```

4.2. Loading Binary Images to Memory

NOTE	SCR bootloader supports only loading of binary files using XMODEM file transferring protocol.

1. Wait for the bootloader's menu and prompt

```
1: xmodem load @addr
g: start @addr
d: dump mem
m: modify mem
i: platform info
:
```

2. Press button **“1”**
3. Enter required starting TCM address (in hex), and press **“Enter”**. Terminal starts to print **“C”** character periodically, indicating receiver’s requests to transmitter in accordance with XMODEM protocol.

```
xload @addr
addr: f0000000
CCCCCCCC
```

4. In terminal program, open XMODEM upload menu. In **minicom** terminal you need to press **“Ctrl+A”** and press **“S”**. Then select “xmodem”:

```
 +-[Upload]--+
 | zmodem    |
 | ymodem    |
 | xmodem    |
 | kermit    |
 | ascii     |
 +-----------+
```

5. In this example we use binary file with Dhrystone benchmark, which could be found in SDK repository by the following path:

```
<SDK_HOME>/images/arty/scr1/scr1-arty-dhry21.tar.gz
```

   Unpack it for further use.

6. Press **“Enter”**. Then select required bin-file for loading (mark it and press **“space”** button for **minicom**).

```
+-----------------------[Select a file for upload]-------------------------+
|Directory: images/arty/scr1                                               |
| [..]                                                                     |
| dhry21-o3lto.bin                                                         |
|                                                                          |
```

7. Press **“Enter”** button. Image transfer will start.

```
+-----------[xmodem upload - Press CTRL-C to quit]---------------------+
|Sending dhry21-o3lto.bin, 107 blocks: Give your local XMODEM receive  |
|command now.                                                          |
|Bytes Sent:  13952   BPS:5468                                         |
|                                                                      |
|Transfer complete                                                     |
|                                                                      |
| READY: press any key to continue...                                  |
+----------------------------------------------------------------------+
```

8. When loading completes, status information will be shown:

```
Xmodem successfully received 13952 bytes
```

4.3. Example: Dhrystone run from TCM memory

1. Load **dhry21-o3lto.bin** to the TCM base address (0xf0000000) as described in the previous section.
2. Select **"g"** menu item, then enter the test's launching address **0xf0000200**. That will start program execution.

```
1: xmodem load @addr
g: start @addr
d: dump mem
m: modify mem
i: platform info
xload @addr
addr: f0000000
CC
Xmodem successfully received 13952 bytes
start @addr
addr: f0000200


test start...
```

3. After benchmark completion you should see its results

```
Time: begin= 513848033, end= 513854416, diff= 6383
Microseconds for one run through Dhrystone: 12.766
Dhrystones per Second:                      78333
```

4.4. Starting the OpenOCD

1. For details on how to get and use the OpenOCD refer to the "SCR1 OpenOCD" appendix section.
2. Setting environment variables:

```
$ export OOCD_ROOT=<Path to the OpenOCD installation directory>
```

3. OpenOCD start-up is entered in one line (Ubuntu):

```
$ sudo ${OOCD_ROOT}/bin/openocd                                          \
-s ${OOCD_ROOT}/share/openocd/scripts                                    \
-f ${OOCD_ROOT}/share/openocd/scripts/interface/ftdi/olimex-arm-usb-ocd-h.cfg  \
-f ${OOCD_ROOT}/share/openocd/scripts/target/syntacore_riscv.cfg

or if you build it from sources:

$ sudo ${OOCD_ROOT}/src/openocd                              \
-s ${OOCD_ROOT}/tcl                                          \
-f ${OOCD_ROOT}/tcl/interface/ftdi/olimex-arm-usb-ocd-h.cfg  \
-f ${OOCD_ROOT}/tcl/target/syntacore_riscv.cfg
```

   After execution in the current terminal, you will receive a message about the connection to the RISC-V kernel:

```
Open On-Chip Debugger 0.10.0+dev-01972-g01f0c8951 (2019-03-20-20:10)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
sw_reset_halt
Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
adapter speed: 2000 kHz
trst_and_srst separate srst_gates_jtag trst_push_pull srst_open_drain
connect_deassert_srst
Info : auto-selecting first available session transport "jtag". To override use
'transport select <transport>'.
riscv.cpu
Info : clock speed 2000 kHz
Info : JTAG tap: riscv.cpu tap/device found: 0xdeb11001 (mfg: 0x000 (<invalid>),
part: 0xeb11, ver: 0xd)
Info : riscv.cpu: datacount=2 progbufsize=6
Info : riscv.cpu: Examined RISC-V core; found 1 harts
Info : riscv.cpu: hart 0: XLEN=32, misa=0x40001104
Info : Listening on port 3333 for gdb connections
```

4. Open the second terminal (terminal 2) and enter the command:

```
$ telnet localhost 4444
```

The command terminal (terminal 1) confirm the telnet session start:

```
Info : accepting 'telnet' connection on tcp/4444
```

5. OpenOCD is up and ready to go. Terminal 2 is an interactive console of the OpenOCD.

4.5. OpenOCD: loading and running BIN and ELF images

IMPORTANT	In contrast to the bootloader, OpenOCD allows loading of program images in both .bin and .elf formats.

1. Start OpenOCD as described in the previous section.
2. Enter the following commands in the OpenOCD console (terminal 2) to halt the core and load an executable code:

```
> halt
> load_image dhry21-o3lto.bin 0xf0000000 bin
```

or

```
> halt
> load_image dhry21-o3lto.elf 0x0 elf
```

IMPORTANT	The boot command assumes the location of the file in the current directory. For a different location, the name of the uploaded file must include a relative path.

3. After entering the command, the progress of the loading is displayed

```
13924 bytes written at address 0xf0000000
downloaded 13924 bytes in 0.392940s (34.605 KiB/s)
```

4. When the loading is complete, start the program:

```
> resume 0xf0000200
```

5. An example of the benchmark's output to the uart terminal is below:

```
Dhrystone Benchmark, Version 2.1 (Language: C)

Program compiled without 'register' attribute

Compiler flags: -O3 -funroll-loops -fpeel-loops -fgcse-sm -fgcse-las  -flto
HZ 1000000, CPU MHz 25.000
Execution starts, 500 runs through Dhrystone
Execution ends

...

Time: begin= 354370015, end= 354376397, diff= 6382
Microseconds for one run through Dhrystone: 12.764
Dhrystones per Second:                      78345
```

5. FPGA Image Modification

5.1. Prerequisites

The following components are necessary:

- **Xilinx Vivado Design Suite 2018.3 or newer**

IMPORTANT	Xilinx Vivado Design Suite older than 2018.3 (for example, 2018.1 and 2018.2 versions) are also supported. Please, rename *system.bd.old* to *system.bd* and replace existing file in *fpga/arty/scr1/bd/system* directory.

5.2. FPGA Project Structure

The SDK project is configured and ready to be built immediately from the repository. The project contains the following main modules:

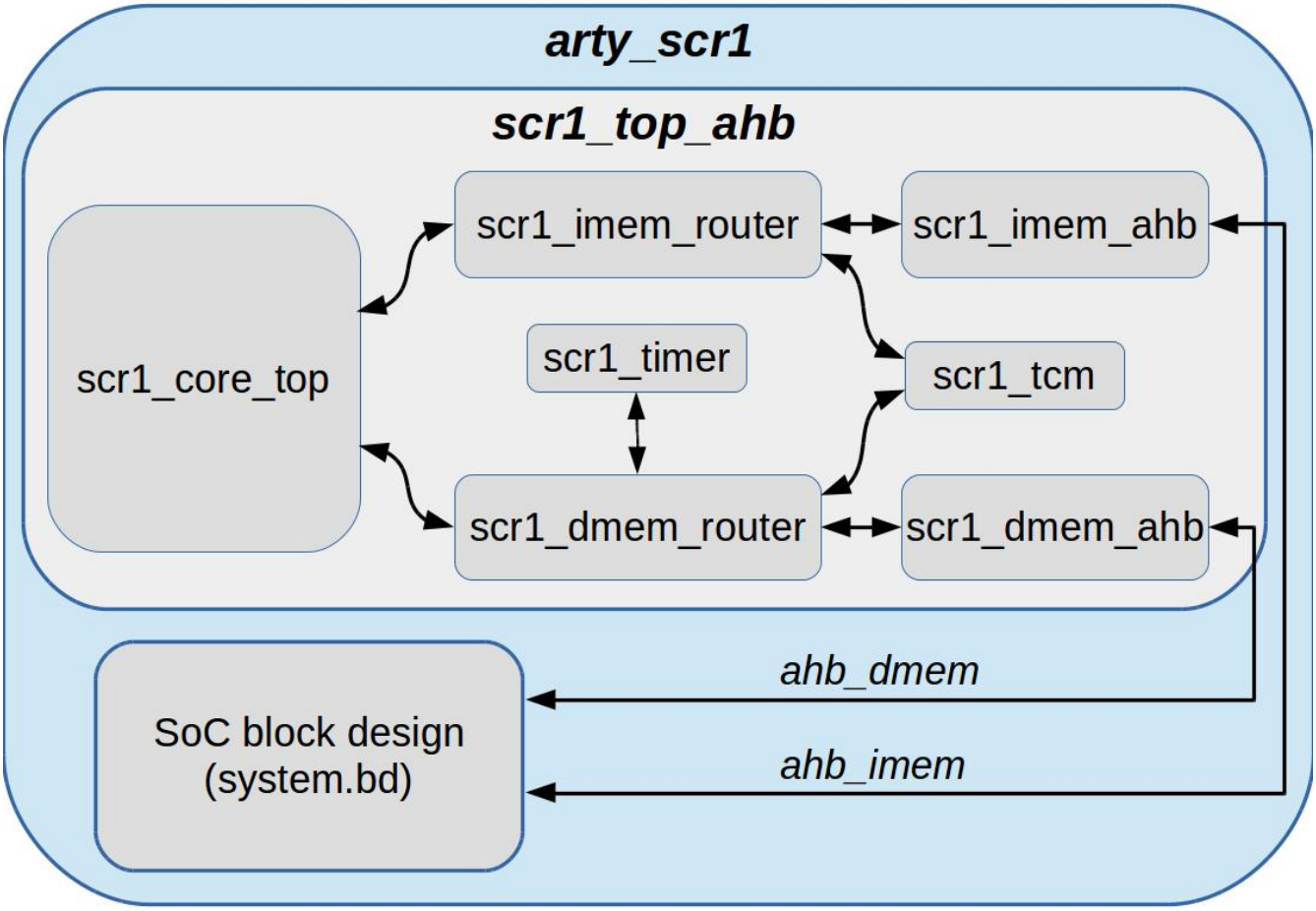

Figure 3. Arty FPGA project structure

Modules description:

- **SCR1-core** - (supplied as an SystemVerilog RTL, is available from the repository)
- **Two data routers** - (imem_router/dmem_router - instruction/data transfers, supplied as an SystemVerilog RTL, is available from the repository)

- **Two AHB bridges** - (imem_ahb/dmem_ahb - instruction/data transfers, supplied as an SystemVerilog RTL, is available from the repository)
- **Timer block** - (external timer block, supplied as an SystemVerilog RTL, is available from the repository)
- **scr1_tcm** - (Tightly Coupled Memory (TCM), supplied as an SystemVerilog RTL, is available from the repository)
- **SoC block design** – (Block Design component, containing the generated IP-components)

5.3. FPGA Project Deployment

1. Install Arty's board files in Vivado directory structure, as described here: https://reference.digilentinc.com/reference/software/vivado/board-files
2. Launch Vivado IDE, and in its Tcl Console change current directory to the <FPGA_PROJECT_HOME_DIR>, where

   ```
   <FPGA_PROJECT_HOME_DIR> = <SDK_HOME>/fpga/arty/scr1
   ```

3. In Tcl Console, execute the following command

   ```
   source ./arty_scr1.tcl
   ```

The script "arty_scr1.tcl" creates Vivado project arty_scr1 and prepares used IPs for further synthesis.

5.4. Building Bitstream File

In the just deployed and open project, click on

- Project Navigator / Program and Debug / Generate Bitstream button

and press OK on the following Vivado confirmation request. This will start the process of full design rebuilding, from synthesis through bitstream file generation.

5.5. Onchip Memory Update

Due to Vivado Design Suite specifics described in the Xilinx AR #63042, initialization of the onchip memories is performed after bitstream file generation, by a standalone script mem_update.tcl.

In the Tcl Console, execute the following commands:

```
source ./mem_update.tcl
```

After successful completion, the folder

```
<FPGA_PROJECT_HOME_DIR>/arty_scr1/arty.runs/impl_1
```

should contain updated bit-file arty_scr1_top_new.bit and MCS-file arty_scr1_top_new.mcs for configuration FLASH chip programming.

5.6. Configuration FLASH Updating

Refer to the section FPGA Configuration Flash Programming for details of that procedure.

6. Appendix A. JTAG Pin-Out

SCR1 JTAG port is routed to the onboard Pmod connector JD in accordance with "Table 1".

Table 1. SCR1 JTAG Pin-Out

Net	JD bit	PMod JD pin
TRSTn	2	3
TCK	3	4
TDO	4	7
TDI	5	8
SRSTn	6	9
TMS	7	10
GND	-	5
GND	-	11
VCC (3.3V)	-	6
VCC (3.3V)	-	12

7. Appendix B. SDK Memory Map

Table 2. SCR1 SDK Memory Map

Base Address	Length	Name	Description
0x00000000	256 MB	SDRAM	Onboard DDR3L SDRAM.
0xF0000000	64 kB	TCM	SCR1 Tightly-Coupled Memory (refer to SCR1 EAS for details).
0xF0040000	32 B	Timer	SCR1 Timer registers (refer to SCR1 EAS for details).
0xFF000000		MMIO BASE	Base address for Memory-Mapped Peripheral IO resources, resided externally to SCR1 core.
0xFF000000	4 kB	SOC_ID	32-bit System-On-Chip ID register.
0xFF001000	4 kB	BLD_ID	32-bit Build ID register.
0xFF002000	4 kB	CORE_CLK_FREQ	32-bit Core Clock Frequency register (frequency value in Herz).
0xFF010000	4 kB	UART	16550 UART registers (refer to Xilinx IP description for details). Interrupt line is assigned to IRQ[0].
0xFF020000	4 kB	LED	LED PIO registers: PIO_LED.
0xFF021000	4 kB	LED_RGB	RGB LED PIO registers: PIO_LED_RGB.
0xFF028000	4 kB	BTN	Push Button PIO register: PIO_PBUTTON. Has associated interrupt line assigned to IRQ[1].
0xFFFF0000	64 kB	SRAM	Onchip SRAM containing pre-programmed SCR Loader firmware. SCR1_RST_VECTOR and SCR1_CSR_MTVEC_BASE are both mapped here: • SCR1_RST_VECTOR = 0xFFFFFF00 • SCR1_CSR_MTVEC_BASE = 0xFFFFFF80

8. Appendix C. SDK IRQs

Table 3. SCR1 IRQ Mapping

IRQ line	Device	Notes
0	UART	Xilinx 16550 UART IP
1	BTN	Xilinx PIO input register, connected to 4 onboard push-buttons.

9. Appendix D. SDK Memory Mapped Registers

9.1. SOC_ID, SOC Identifier Register (0xFF000000)

Table 4. SOC Identifier Register (SOC_ID)

Bit(s)	Name	Description
0 .. 31	ID	32-bit System-On-Chip Identifier. It specifies version of the FPGA's System-On-Programmable-Chip RTL module, main part of this platform's FPGA which is outside the SCR1 processor cluster.

9.2. BLD_ID, Build Identifier Register (0xFF001000)

Table 5. Build Identifier Register (BLD_ID)

Bit(s)	Name	Description
0 .. 31	ID	32-bit Build Identifier. It specifies FPGA build as a whole.

9.3. CORE_CLK_FREQ, Core Clock Frequency Register (0xFF002000)

Table 6. Core Clock Frequency Register (CORE_CLK_FREQ)

Bit(s)	Name	Description
0 .. 31	FREQ	32-bit value of the Core Clock Frequency in Herz.

9.4. PIO_LED, Programmable IO LED Control Register (0xFF020000)

Table 7. Programmable IO LED Control Register

Bit(s)	Name	Description
0	LED0	LED[0] control: corresponds to the onboard LD4. If a bit is 1, LED is illuminated.
1	LED1	LED[1] control (onboard LD5).

9.5. PIO_LED_RGB, Programmable IO LED RGB Control Register (0xFF021000)

Table 8. Programmable IO LED RGB Control Register

Bit(s)	Name	Description
0..2	LED0	LED[0] control: bits [2:0] correspond to {red, green, blue} partial LEDs of the onboard LD0. If a bit is 1, appropriate internal LED is illuminated.
3..5	LED1	LED[1] control (onboard LD1).
6..8	LED2	LED[2] control (onboard LD2).
9..11	LED3	LED[3] control (onboard LD3).

9.6. PIO_PBUTTON, Programmable IO Push Button Status Register (0xFF028000)

Table 9. Programmable IO Push Button Status Register

Bit(s)	Name	Description
0..3	BTN	BTN status: bits [3:0] correspond to {BTN3, BTN2, BTN1, BTN0} onboard push buttons. For details refer to the Xilinx AXI GPIO IP documentation.

10. Appendix E. Software build instructions

This build guide describes how to build software provided as a part of the SCR1 SDK.

10.1. SCR bootloader

10.1.1. Getting the sources

```
$ git clone git@github.com:syntacore/sc-bl.git
```

10.1.2. Building SCR bootloader

Follow the instructions in sc-bl/README.md to build bootloader for target plaforms ('scbl.hex' for Terasic DE10-Lite, 'scbl.mem' for Digilent Arty and Nexys4DDR).

10.2. Zephyr OS

10.2.1. Getting the sources

```
$ git clone git@github.com:syntacore/zephyr.git
```

10.2.2. Building Zephyr OS

Follow the instructions in https://www.zephyrproject.org/doc/getting_started/getting_started.html and zephyr/README.md to build Zephyr OS image for target plaform.

10.3. SCR1 OpenOCD

10.3.1. Getting the latest release

```
The latest release (sc-riscv-0.10.0-1972) can be downloaded from the link:
https://github.com/syntacore/openocd/releases
or you can build it from sources.
```

10.3.2. Getting the sources

```
$ git clone -b syntacore https://github.com/syntacore/openocd
```

10.3.3. Building and using OpenOCD

Please, refer to the Syntacore OpenOCD wiki page for instructions: https://github.com/syntacore/

10.3.4. Windows - USB JTAG Cable drivers installation

In order to use Olimex and Digilent JTAG cable with the OpenOCD the correct drivers should be installed at the host PC. After cable is connected to the host PC, the properly installed drivers should appear in the device manager as shown in the figure below:

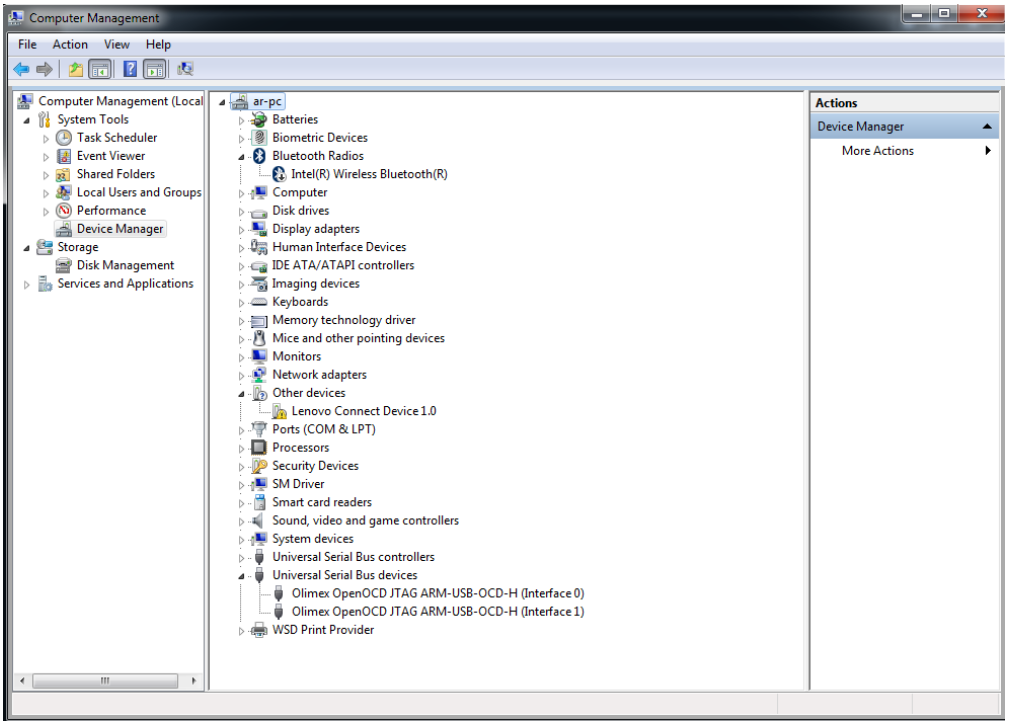

Figure 4. Windows device manager with properly installed USB JTAG Cable drivers

If you system doesn't recognize devices properly (as in the figure below), you may need to install the latest availalbe drivers.

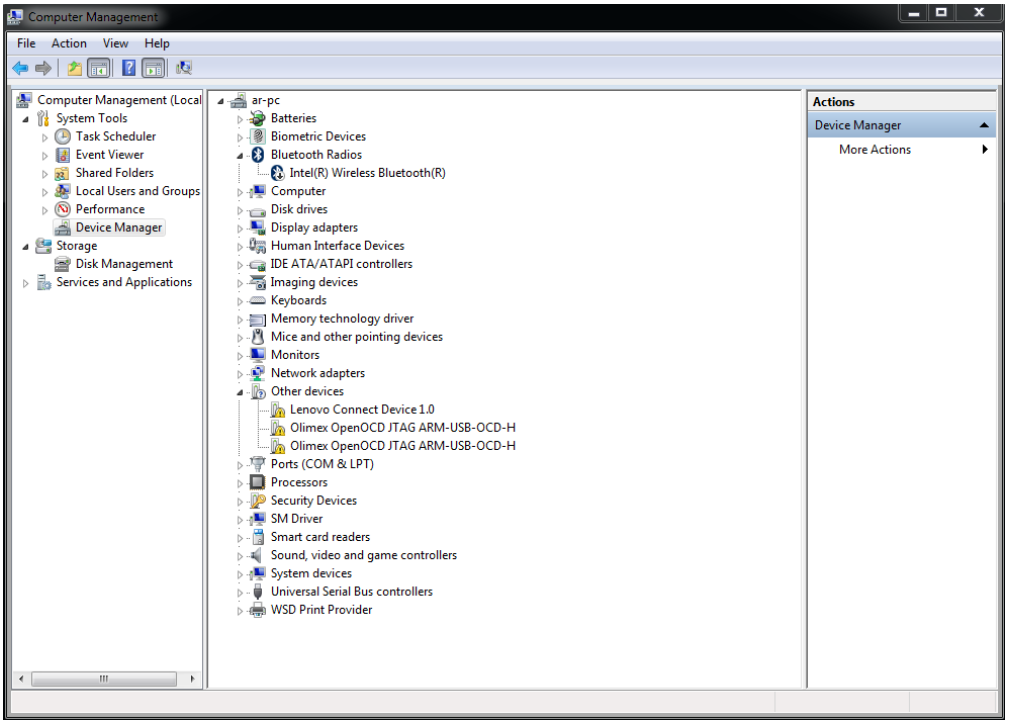

Figure 5. Windows device manager: USB JTAG Cable drivers are not installed

In many cases, generic WinUSB driver by Microsoft, which can be enforced using Zadig application, can solve the problem:

http://zadig.akeo.ie/

IMPORTANT	Be very very careful! You should see and select the exactly proper USB device/channel before pressing 'Zadig' WinUSB replace driver button! Don't press button with no selection or without proper selection!

To apply WinUSB driver to Olimex and Digilent devices, just start application, make sure "Options→List all devices" menu item is checked:

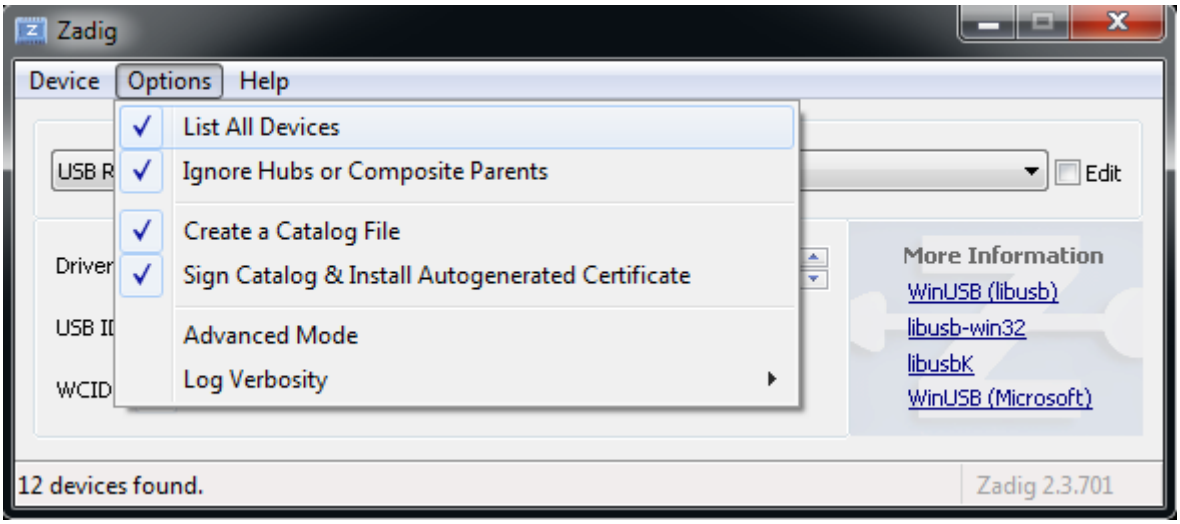

Figure 6. Zadig program: choose to enumerate all devices

Then, choose WinUSB driver for the device, and press Install. This should be done two times, for Olimex both interfaces.

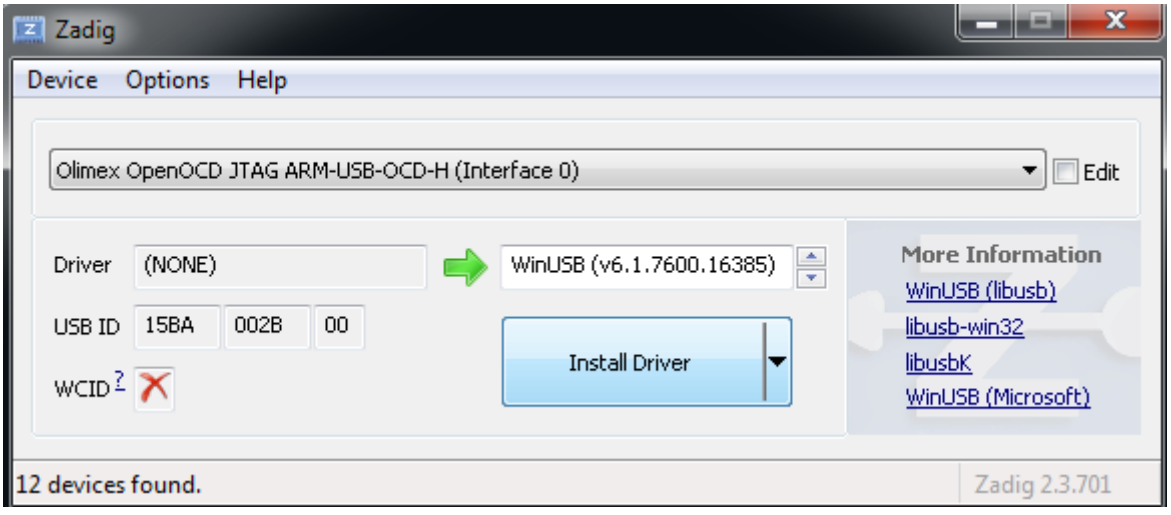

Figure 7. Zadig program: replace previously assigned driver with WinUSB

You can also check this page for the latest information on the Olimex drivers availability for your platform:

https://www.olimex.com/wiki/ARM-USB-OCD