

# SCR1 SDK. Terasic DE10-Lite Edition. Quick Start Guide

© Syntacore, [scr1@syntacore.com](mailto:scr1@syntacore.com)

Version 1.6, 2021-05-18

# Table of Contents

Revision History .....	2
1. Setup equipment .....	3
2. SDK HW assembly .....	4
2.1. Connecting serial console .....	4
2.2. Pins assignment .....	4
2.2.1. UART pins (TTL-232R-3V3) .....	4
2.2.2. JTAG pins (Olimex ARM-USB-OCD-H) .....	4
3. DE10-Lite flash image update .....	5
3.1. Required equipment .....	5
3.2. FPGA binary files .....	6
3.3. Update procedure steps .....	6
4. Resetting the board: .....	8
5. UART connection settings .....	8
6. Using UART terminal .....	9
6.1. Load binary images to the Memory address .....	9
6.2. Example: Dhrystone run from TCM memory .....	11
7. Using OpenOCD .....	13
7.1. Starting the OpenOCD .....	13
7.2. OpenOCD: loading and running BIN and ELF images .....	14
8. Building SDK FPGA-project for the DE10-Lite board .....	16
8.1. General structure of the SDK project .....	16
8.2. Additional requirements for compilation .....	16
8.3. Building SDK FPGA project .....	17
8.3.1. FPGA firmware generation (pof-format) .....	17
8.3.2. SDK-specific pins assignment in FPGA-project .....	17
8.4. SCR1 SDK FPGA-project functional description .....	18
8.4.1. Common project structure .....	18
8.4.2. Qsys SoC module structure .....	18
8.4.3. Description of the blocks used in the SDK project .....	19
8.4.3.1. SCR1-core .....	19
8.4.3.2. AHB-Avalon bridge .....	19
8.4.3.3. Opencores UART 16550 IP .....	19
8.4.4. Description of the IP-components of the module Qsys SOC .....	20
8.4.4.1. SOC_ID .....	20
8.4.4.2. BLD_ID .....	20
8.4.4.3. CORE_CLK_FREQ .....	20
8.4.4.4. Onchip RAM .....	20
8.4.4.5. PIO HEX .....	20

8.4.4.6. PIO LED .....	20
8.4.4.7. PIO SW .....	21
8.4.4.8. SDRAM .....	21
8.4.4.9. UART Bridge .....	21
8.4.4.10. Qsys Default slave .....	21
9. Appendix A. SDK Memory Map .....	22
10. Appendix B. SDK IRQs .....	23
11. Appendix C. Software build instructions .....	24
11.1. SCR bootloader .....	24
11.1.1. Getting the sources .....	24
11.1.2. Building SCR bootloader .....	24
11.2. Zephyr OS .....	24
11.2.1. Getting the sources .....	24
11.2.2. Building Zephyr OS .....	24
11.3. SCR1 OpenOCD .....	24
11.3.1. Getting the latest release .....	24
11.3.2. Getting the sources .....	24
11.3.3. Building and using OpenOCD .....	24
11.3.4. Windows - USB JTAG Cable drivers installation .....	25

Copyright by Syntacore LLC © 2017-2021. ALL RIGHTS RESERVED. STRICTLY CONFIDENTIAL. Information contained in this material is confidential and proprietary to Syntacore LLC and its affiliates and may not be modified, copied, published, disclosed, distributed, displayed or exhibited, in either electronic or printed formats without written authorization of the Syntacore LLC. Subject to License Agreement.

# Revision History

Version	Date	Description
0.1	2017-09-08	Initial revision
1.0	2017-12-14	Changed UART IP to the Opencores UART 16550 IP
1.1	2019-01-30	Modifications: <ul style="list-style-type: none"><li>• New sections: "Using OpenOCD", "Windows - USB JTAG Cable drivers installation"</li><li>• Sections with memory map and IRQ mapping are moved to appendix;</li><li>• Figure numbering is introduced.</li></ul>
1.2	2019-03-21	OpenOCD section updated for RISC-V debug. JTAG speed requirement added.
1.3	2019-03-29	Fix figures numbers
1.4	2019-04-08	Updated IMPID, BUILDID
1.5	2019-09-10	Updated IMPID, BUILDID
1.6	2021-05-18	Updated: SDK Memory Map, bootloader's logo, FPGA binary files

*This is a brief user guide allowing to get started with SCR1 SDK based on [DE10-Lite Board](#) from Terasic. It describes the board setup, procedure of software uploading and launching, and process of the FPGA's content building and updating.*

---

# 1. Setup equipment

DE10-Lite based SCR1 SDK HW platform consist of three mandatory components:

1. DE10-Lite Development System  
<http://de10-lite.terasic.com>
2. Any 3V3 USB-to-UART converter  
For example [TTL-232R-3V3](#)
3. JTAG Cable Adapter: Olimex ARM-USB-OCD-H (or Olimex ARM-USB-OCD)  
<https://www.olimex.com/Products/ARM/JTAG/ARM-USB-OCD-H/>
4. Standard USB Type A (m) - Type B (m) cable (included the DE10-Lite Board Kit contents)
5. Standard USB Type A (m) - Type B (m) cable (for Olimex ARM-USB-OCD-H connection)
6. Male-to-Female Jumper Wires The wires, e.g., might be of the following type:  
[Female/Male 'Extension' Jumper](#)

## 2. SDK HW assembly

### 2.1. Connecting serial console

In order to get access to the board console, it is required to connect any 3V3 USB-to-UART converter to the **GPIO** header with external wiring, as described in this section.

### 2.2. Pins assignment

#### 2.2.1. UART pins (TTL-232R-3V3)

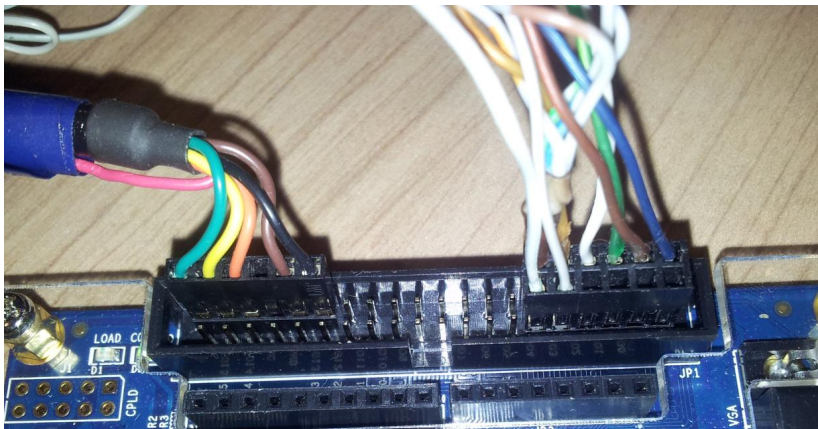
- Connect USB-to-UART pin RXD to the 4 pin (GPIO\_D3) on the GPIO header
- Connect USB-to-UART pin TXD to the 6 pin (GPIO\_D5) on the GPIO header
- Connect USB-to-UART pin GND to the 12 pin (GPIO\_GND) on the GPIO header

#### 2.2.2. JTAG pins (Olimex ARM-USB-OCD-H)

- Connect JTAG pin TCK to the 32 pin (GPIO\_D27) on the GPIO header
- Connect JTAG pin TRSTn to the 34 pin (GPIO\_D29) on the GPIO header
- Connect JTAG pin TDI to the 36 pin (GPIO\_D31) on the GPIO header
- Connect JTAG pin TDO to the 38 pin (GPIO\_D33) on the GPIO header
- Connect JTAG pin TMS to the 40 pin (GPIO\_D35) on the GPIO header
- Connect JTAG pin GND to the 30 pin (GPIO\_GND) on the GPIO header
- Connect JTAG pin VCC to the 29 pin (GPIO\_VCC33) on the GPIO header

As shown in the figures below:

- Wires connection to UART pins and JTAG pins



*Figure 1. UART and JTAG connection*

## IMPORTANT

For proper JTAG interface functioning JTAG clock (TckFreq) and system clock (SysClkFreq) frequencies must satisfy the following relation:  $\text{SysClkFreq} / \text{TckFreq} \geq 12$ .

- Resulting setup

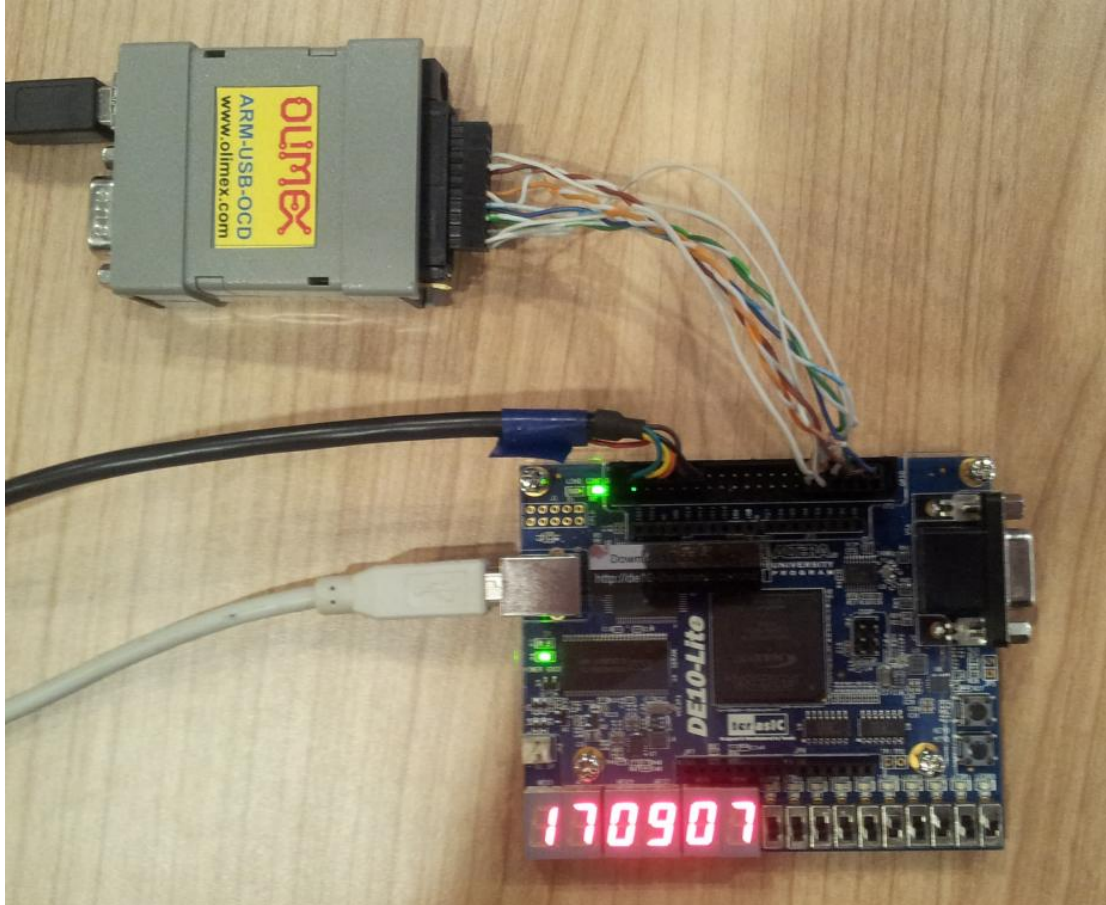


Figure 2. DE10-Lite SDK setup

## 3. DE10-Lite flash image update

Image update procedure will load FPGA firmware to the FPGA flash memory (MAX10). The FPGA firmware image in the flash memory is then loaded upon every board power on. Binary file used for the update is in the Altera standard .pof format.

### 3.1. Required equipment

- USB A (m) - USB B (m) cable
- "Quartus II Programmer" tool (version 17.0 or earlier).  
(Can be downloaded from [Altera site](#) after registration)
- Linux/Windows PC with USB port



## 3.2. FPGA binary files

FPGA binary files could be obtained from here:

```
<SDK_HOME>/images/de10lite/scr1/scr1-de10lite-fpga.7z
```

This 7-Zip archive contains the following files:

- de10lite\_scr1.sof
- de10lite\_scr1.pof

Use de10lite\_scr1.pof file in the following FLASH update procedure.

## 3.3. Update procedure steps

1. Power on DE10-Lite board
2. Run "Quartus II Programmer" tool.  
Select "Hardware Setup" button, and the select "USB-Blaster" as shown below:

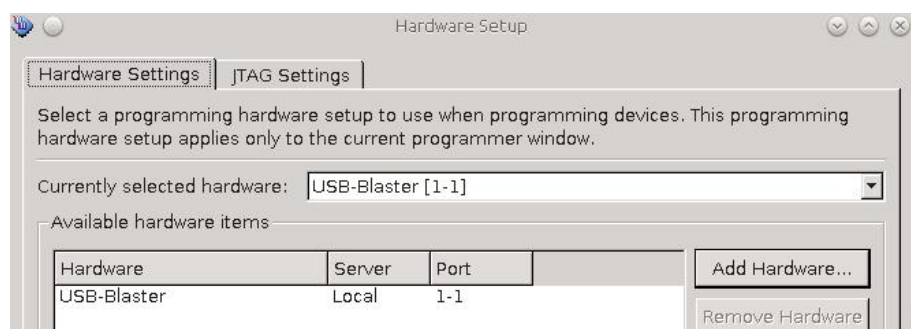


Figure 3. Quartus II Programmer hardware setup

3. Press "Add File" button and select .pof file
4. Select "Program/Configure" checkboxes end press "Start" button

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase	ISP CLAMP	IPS File
output/de10lite.pof	10M50DAF484	10DE67D0	00BECE80	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
CFM0				<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
UFM				<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figure 4. POF-file programming setup

5. Wait for the loading to complete



Figure 5. POF-file programming process indication

6. MAX10 flash update is complete.

New FPGA firmware is already running.

## 4. Resetting the board:

Press *Key0* button if you need to reset the board and go back into the bootloader at any time. Corresponding button is shown in the figure below:

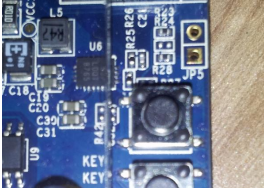


Figure 6. Reset button

## 5. UART connection settings

### NOTE

- Bps/Par/Bits - 115200 8N1
- speed - 115200
- bits - 8
- stop bits - 1
- parity - none
- Hardware Flow Control: No

## 6. Using UART terminal

1. Connect PC to the uart port and open any terminal (minicom is used in the example below)  
After reset or FPGA firmware update you will see the bootloader prompt:

```
SCR loader v1.2-scr1_RC
Copyright (C) 2015-2021 Syntacore. All rights reserved.
ISA: RV32IMC [40001104] IMPID: 21051400
SOCID: 21041500 BLDID: 21051403
Platform: de10lite_scr1, cpuclock 20MHz, sysclk 20MHz
Memory map:
00000000-03FFFFFF      00000000      SDRAM
F0000000-F000FFFF      00000000      TCM
F0040000-F0040FFF      00000000      MTimer
FF000000-FF00FFFF      00000000      MMIO
FFFF0000-FFFFFFFF      00000000      On-Chip RAM

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

1. If you press "i" button you can see additional info about the platform

```
ISA: RV32IMC [40001104] IMPID: 21051400
SOCID: 21041500 BLDID: 21051403
Platform: de10lite_scr1, cpuclock 20MHz, sysclk 20MHz
Memory map:
00000000-03FFFFFF      00000000      SDRAM
F0000000-F000FFFF      00000000      TCM
F0040000-F0040FFF      00000000      MTimer
FF000000-FF00FFFF      00000000      MMIO
FFFF0000-FFFFFFFF      00000000      On-Chip RAM
Platform configuration:
FF010000      irq 0      UART16550
FF020000      LED
FF022000      Hex LED
FF029000      DIP sw
:
```

### 6.1. Load binary images to the Memory address

**TIP***SCR bootloader supports only .binary files loading using x-modem*

1. Wait for the bootloader prompt

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

2. Press button “1”

3. Print required TCM address (in hex) and press “**Enter**”. “C” character starts to print continuously

```
xload @addr
addr: f0000000
CCCCCCCCCCCCCCC
```

1. Open xmodem upload menu (for minicom terminal you need to press “**Ctrl+A**” and press “**S**”). Then select “xmodem”:

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

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

```
+-----[Select a file for upload]-----+
|Directory: /images/de10lite/scr1/          |
| [..]                                     |
| dhry21-o3lto.bin                          |
|                                           |
```

1. 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...                        |  
+-----+
```

- ```
Xmodem successfully received 13952 bytes
```

## 6.2. Example: Dhrystone run from TCM memory

1. Load **dhry21-o3lto.bin** to the TCM base address (0xf0000000)  
And run test from **0xf0000200** address:

```
1: xmodem load @addr
g: start @addr
d: dump mem
m: modify mem
i: platform info
start @addr
addr: f000200
```

1. After run you will see test results

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 20000000, CPU MHz 20.000

Execution starts, 500 runs through Dhrystone

...

Time: begin= 258424349, end= 258432378, diff= 8029

Microseconds for one run through Dhrystone: 16.058

Dhrystones per Second: 62664

# 7. Using OpenOCD

## 7.1. 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
```

### IMPORTANT

For compliance with the requirement {fCoreClkFreq/fTckFreq > 12} you should modify the file **syntacore\_riscv.cfg** to decrease JTAG TCK frequency down to 1 MHz as follows:

```
# Original line:
...
adapter_khz 2000
...

# Modified line:
...
adapter_khz 1000
...
```

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: 1000 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 1000 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.

## 7.2. 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 0.532163s (25.552 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 20.000  
Execution starts, 500 runs through Dhrystone  
Execution ends  
  
...  
  
Time: begin= 126054014, end= 126061992, diff= 7978  
Microseconds for one run through Dhrystone: 15.956  
Dhrystones per Second: 62672
```

# 8. Building SDK FPGA-project for the DE10-Lite board

## 8.1. General structure of the SDK project

The composition of the SDK folders is:

- doc - SDK and SCR1 user guides
- fpga
  - arty
  - de10lite
    - scr1 - DE10-Lite FPGA project
      - ip - additional RTL IPs + bootloader image
        - uart - Opencores UART 16550 IP
- images
  - arty
  - de10lite
    - scr1 - pre-built FPGA image
- scr1 - SCR1 repository, included as sub-module
  - src - SCR1 core RTL sources
- sw
  - sc-bl - FPGA-bootloader
  - tests - some benchmark tests

Essential files: FPGA project file - de10lite\_scr1.qpf (fpga/de10lite/scr1/de10lite\_scr1.qpf)

Top module - de10lite\_scr1 (fpga/de10lite/scr1/de10lite\_scr1.sv)

## 8.2. Additional requirements for compilation

FPGA build requires "Altera Quartus 17.0.1 Build 598" tool or earlier.

### NOTE

FPGA-project compilation was verified for "Altera Quartus 17.0.1 Build 598" Standart Edition on Linux xUbuntu 16.04 with 8 GB of RAM.  
Some build steps may be different for other Quartus versions.  
Free **Quartus Prime Lite Edition** is required for a non time-limited HW firmware generation for MAX10 FPGA devices.

## 8.3. Building SDK FPGA project

The step-by-step FPGA project build procedure is described below:

### 8.3.1. FPGA firmware generation (pof-format)

- Run Quartus 17.0.1 in GUI-mode
- Select and open fpga-project file (de10lite\_scr1.qpf)
- Press "Start Compilation" button or select from the menu Processing → "Start Compilation"

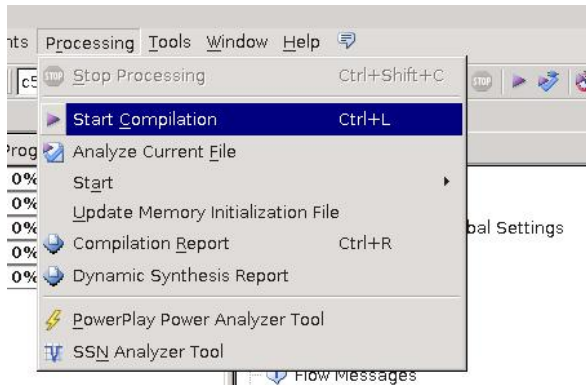


Figure 7. Selection of the "Start Compilation" option

- Wait for the compilation to complete (build time is typically 10-15 minutes)
- New "output" subfolder should appear in the FPGA project "fpga" folder. It contains de10lite\_scr1.pof file (FPGA image in pof-format).

### 8.3.2. SDK-specific pins assignment in FPGA-project

Most of the pins assignments of the SDK project "inherit" from the basic design:

[DE10-Lite User manual](#)

SDK-specific connection pins are used for interfaces UART and OpenOCD/JTAG. The purpose of these pins is shown below:

Table 1. SDK-specific pins assignment

| FPGA-pin | Port name   | I/O Standard | Descrpition      |
|----------|-------------|--------------|------------------|
| PIN_Y5   | JTAG_TRST_N | 3.3V         | Input JTAG TRSTn |
| PIN_Y4   | JTAG_TDI    | 3.3V         | Input JTAG TDI   |
| PIN_AA2  | JTAG_TMS    | 3.3V         | Input JTAG TMS   |
| PIN_Y6   | JTAG_TCK    | 3.3V         | Input JTAG TCK   |
| PIN_Y3   | JTAG_TDO    | 3.3V         | Inout JTAG TDO   |
| PIN_W9   | UART_RXD    | 3.3V         | Inout UART RXD   |
| PIN_W8   | UART_TXD    | 3.3V         | Inout UART TXD   |

## 8.4. SCR1 SDK FPGA-project functional description

### 8.4.1. Common project structure

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

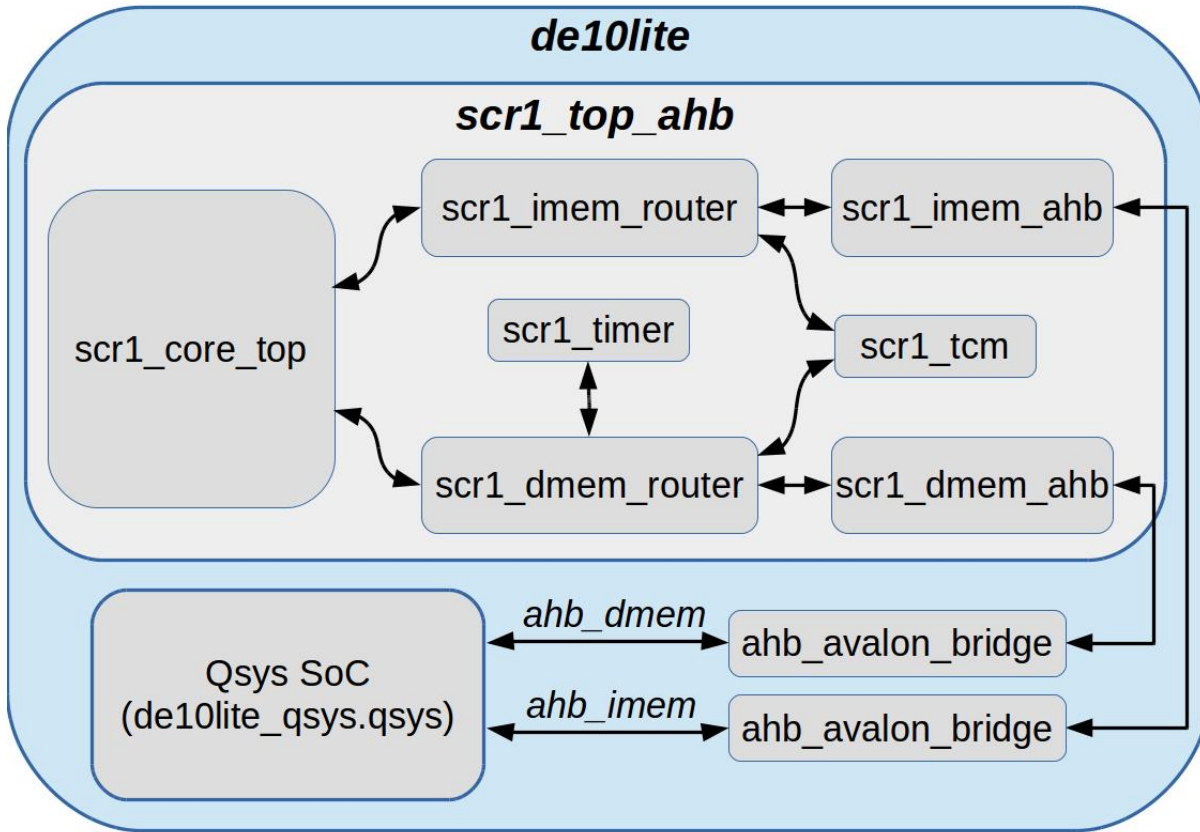


Figure 8. DE10-Lite FPGA project structure

Modules description:

- **SCR1-core** - (supplied as an SystemVerilog RTL, is available from the repository)
- **Two data routers** (imem\_route/dmem\_router - instruction/data transfers, supplied as an SystemVerilog RTL, is available from the repository)
- **Two AHB-Avalon bridges** (ahb\_imem/ahb\_dmem - 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)
- **Qsys SoC block** - (Qsys component, containing the generated IP-components)

### 8.4.2. Qsys SoC module structure

Qsys SoC module consists of:

- BUILD ID
- SDRAM
- UART Bridge
- Onchip RAM
- PIO HEX
- PIO SW
- PIO LED
- Qsys Default slave

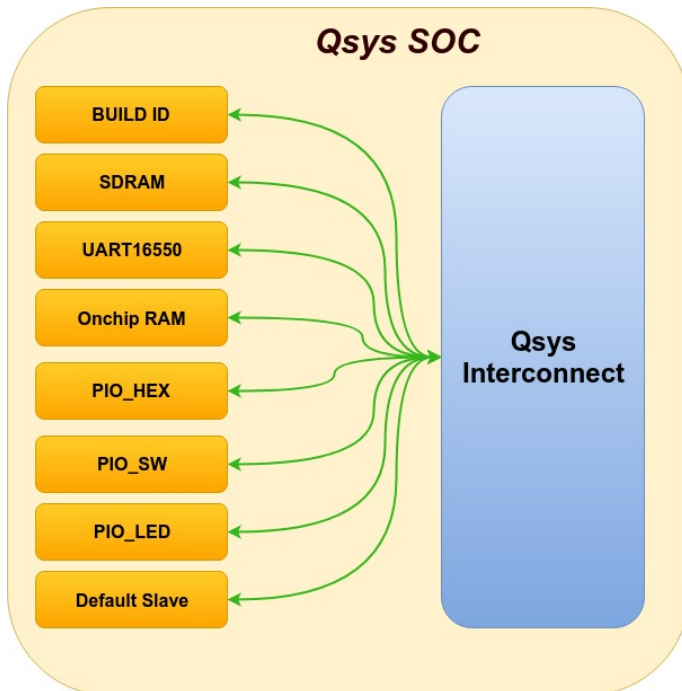


Figure 9. DE10-Lite QSys SOC module structure

### 8.4.3. Description of the blocks used in the SDK project

#### 8.4.3.1. SCR1-core

Syntacore RISC-V core.

The core is supplied as SystemVerilog RTL sources.

A detailed description of the external interfaces of the core and other details are described in the document "SCR1 External Architecture Specification".

#### 8.4.3.2. AHB-Avalon bridge

AHB bridge converts internal imem/dmem bus interface to the [Altera Avalon](#) imem/dmem bus interface. Provided in SV sources.

#### 8.4.3.3. Opencores UART 16550 IP

There is a 16550 compatible (mostly) UART core. The bus interface is WISHBONE SoC bus Rev. B. Features all the standard options of the 16550 UART: FIFO based operation, interrupt requests and other.

#### 8.4.4. Description of the IP-components of the module Qsys SOC

Detailed description of Altera's common Qsys components used in the project:

- [Embedded Peripheral IP User Guide](#)
- [Qsys System Design Tutorial](#)
- [Qsys Interconnect](#)

##### 8.4.4.1. SOC\_ID

PIO-block contains the project's SOPC version ID, which is available for reading by the processor. The parameter SCR1\_PTFM\_SOC\_ID is defined in the file scr1\_arch\_custom.svh. Component base address - 0xFF000000.

##### 8.4.4.2. BLD\_ID

PIO-block contains the project build date parameter, which is available for reading by the processor. The parameter SCR1\_PTFM\_BLD\_ID is defined in the file scr1\_arch\_custom.svh. Component base address - 0xFF001000.

##### 8.4.4.3. CORE\_CLK\_FREQ

PIO-block contains register with the core clock frequency value (in Herz), which is available for reading by the processor. The parameter SCR1\_PTFM\_CORE\_CLK\_FREQ is defined in the file scr1\_arch\_custom.svh. Component base address - 0xFF002000.

##### 8.4.4.4. Onchip RAM

Internal bootload memory with bootloader code resides in the FPGA. Memory size - 64KB. Memory base address - 0xFFFF0000.

Memory initialization data is supplied in sclb.hex (hex format). The code and instructions for building the bootloader are available from the current repository.

Further details:

[http://www.altera.com/literature/hb/qts/qts\\_qii54006.pdf](http://www.altera.com/literature/hb/qts/qts_qii54006.pdf)

##### 8.4.4.5. PIO HEX

Three PIO-blocks for the 7-segments display control. Available for CPU write and read. Components base addresses - 0xFF022000, 0xFF022010, 0xFF022020.

##### 8.4.4.6. PIO LED

PIO-block for the LED indication control. Available for CPU write and read. Component base address - 0xFF020000.

#### **8.4.4.7. PIO SW**

PIO-block for switches position read. Available for CPU read  
Component base address - 0xFF029000.

#### **8.4.4.8. SDRAM**

SDRAM-controller for the external 64MB (32Mx16) SDRAM chip on the DE10-Lite board.  
Component base address - 0x00000000.

#### **8.4.4.9. UART Bridge**

Altera avalon bridge for the Opencores UART 16550 IP.  
Component base address - 0xFF010000.

#### **8.4.4.10. Qsys Default slave**

The slave responder component by "default". The main function is to generate an error status for any transactions in the unused ranges of the addresses of the Qsys SoC.



## 9. Appendix A. SDK Memory Map

Memory map is shown in the table below:

Table 2. SCR1 DE10-Lite SDK memory map

| Base Address | Length | Name          | Description                                                                                                                                                                                                                                         |
|--------------|--------|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0x00000000   | 64 MB  | SDRAM         | External SDRAM memory                                                                                                                                                                                                                               |
| 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   | 16 B   | SOC_ID        | 32-bit System-On-Chip ID register.                                                                                                                                                                                                                  |
| 0xFF001000   | 16 B   | BLD_ID        | 32-bit Build ID register.                                                                                                                                                                                                                           |
| 0xFF002000   | 16 B   | 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   | 16 B   | PIO LED       | PIO-block for the LED indication control                                                                                                                                                                                                            |
| 0xFF022000   | 16 B   | PIO HEX 1_0   | PIO-block for the 7-segments display control 1:0                                                                                                                                                                                                    |
| 0xFF022010   | 16 B   | PIO HEX 3_2   | PIO-block for the 7-segments display control 3:2                                                                                                                                                                                                    |
| 0xFF022020   | 16 B   | PIO HEX 5_4   | PIO-block for the 7-segments display control 5:4                                                                                                                                                                                                    |
| 0xFF029000   | 16 B   | PIO SW        | PIO-block for switches position read (Read only)                                                                                                                                                                                                    |
| 0xFFFF0000   | 64 kB  | SRAM          | Onchip SRAM containing pre-programmed SCR Loader firmware. SCR1_RST_VECTOR and SCR1_CSR_MTVEC_BASE are both mapped here: <ul style="list-style-type: none"><li>• SCR1_RST_VECTOR = 0xFFFFFFF0</li><li>• SCR1_CSR_MTVEC_BASE = 0xFFFFFFF80</li></ul> |

## 10. Appendix B. SDK IRQs

The connection scheme for interrupt lines is given below:

*Table 3. SCR1 core IRQ connection*

| IRQ line for the SCR1 core | IRQ init block                   |
|----------------------------|----------------------------------|
| 0                          | UART (UART 16550)                |
| 1-31                       | Not connected (constant level 0) |

# 11. Appendix C. Software build instructions

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

## 11.1. SCR bootloader

### 11.1.1. Getting the sources

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

### 11.1.2. 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).

## 11.2. Zephyr OS

### 11.2.1. Getting the sources

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

### 11.2.2. Building Zephyr OS

Follow the instructions in [https://www.zephyrproject.org/doc/getting\\_started/getting\\_started.html](https://www.zephyrproject.org/doc/getting_started/getting_started.html) and `zephyr/README.md` to build Zephyr OS image for target plaform.

## 11.3. SCR1 OpenOCD

### 11.3.1. 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.

### 11.3.2. Getting the sources

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

### 11.3.3. Building and using OpenOCD

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

### 11.3.4. 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:

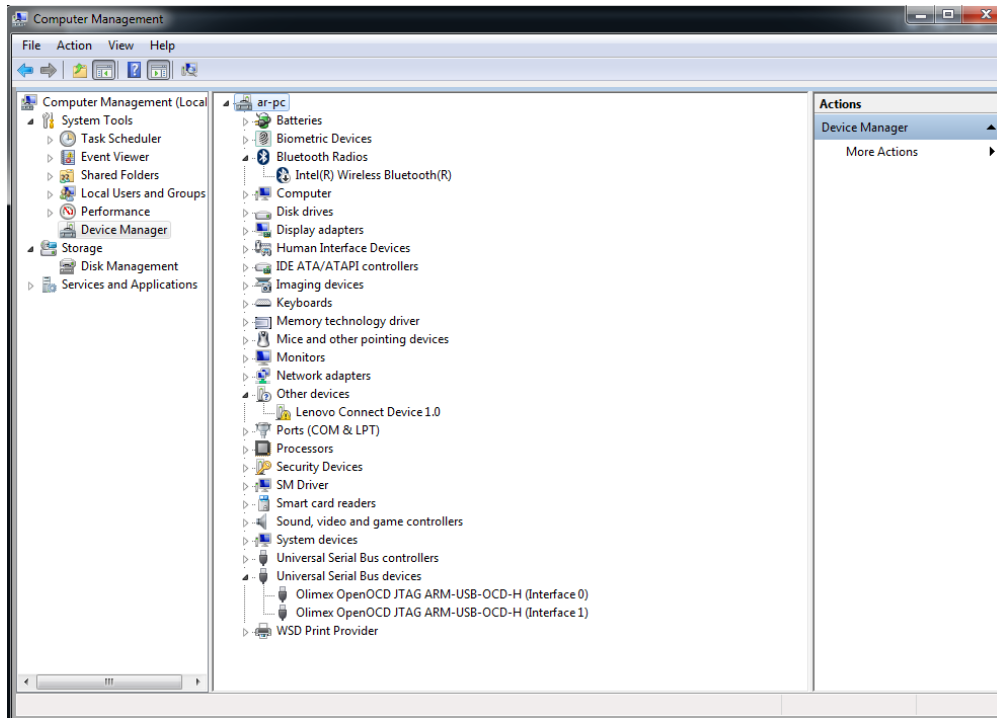


Figure 10. Windows device manager with properly installed USB JTAG Cable drivers

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

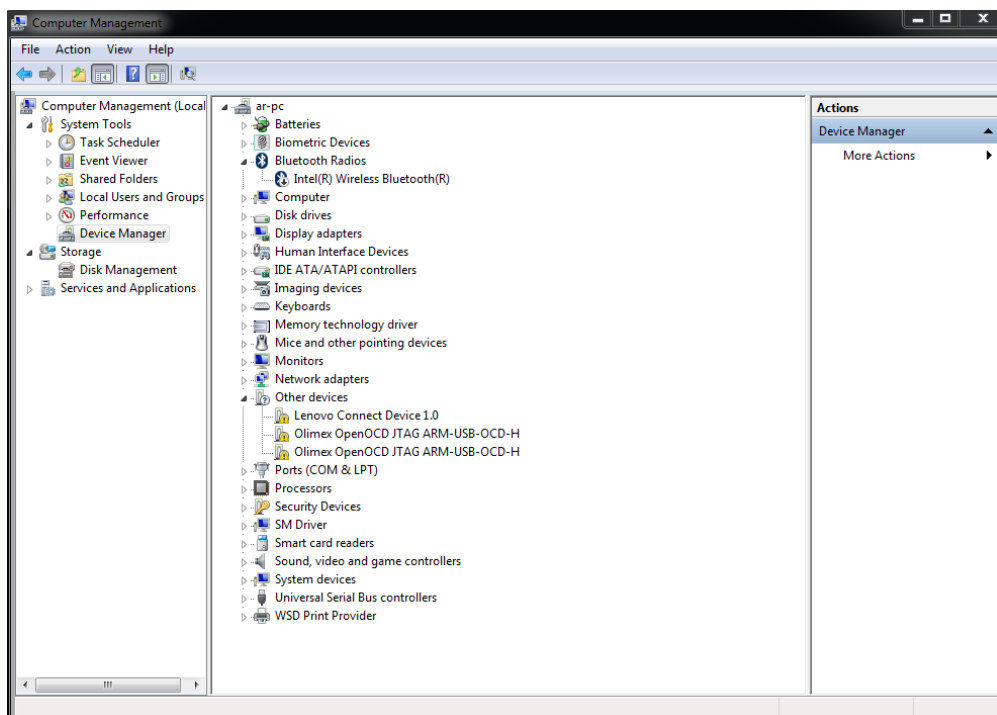


Figure 11. Windows device manager: USB JTAG Cable drivers are not installed

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:

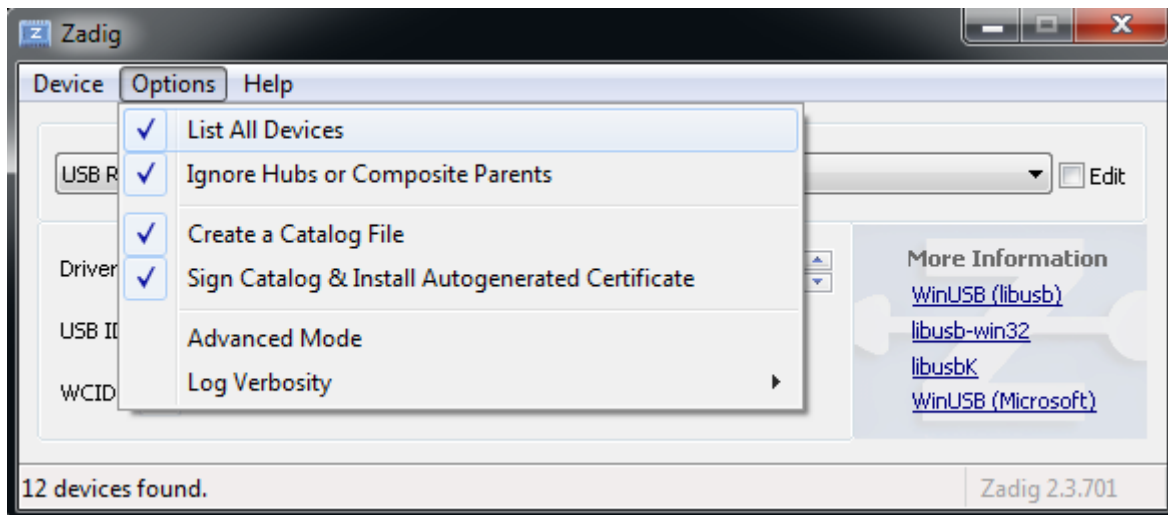


Figure 12. Zadig program: choose to enumerate all devices

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

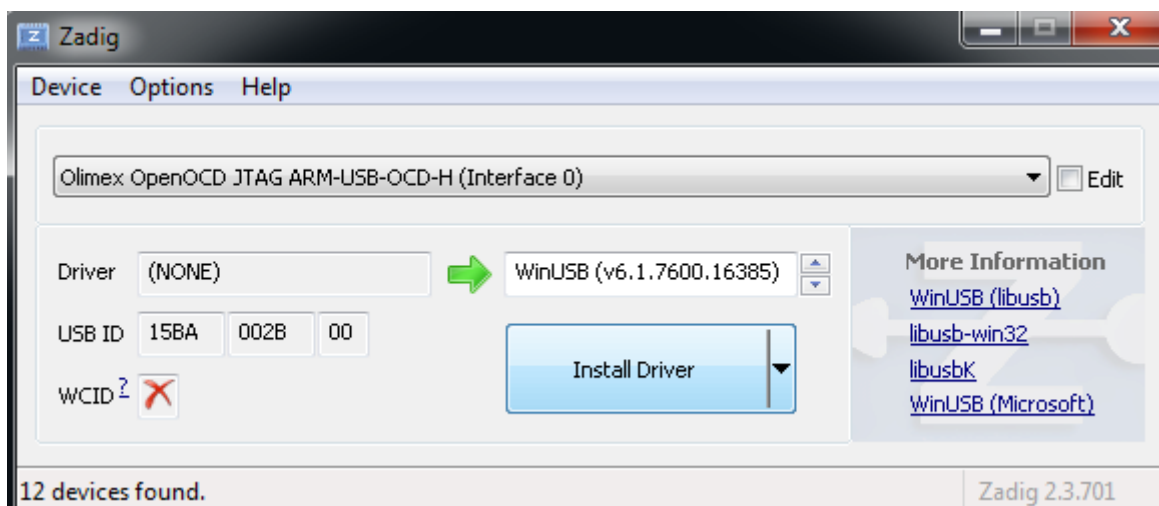


Figure 13. Zadig program: replace previously assigned driver with WinUSB

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>