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

v0.1, 2017-09-08

# Table of Contents

# Revision history

| Version | Date | Description |
|---|---|---|
| 0.10 | 2017-09-08 | Initial revision |

*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)

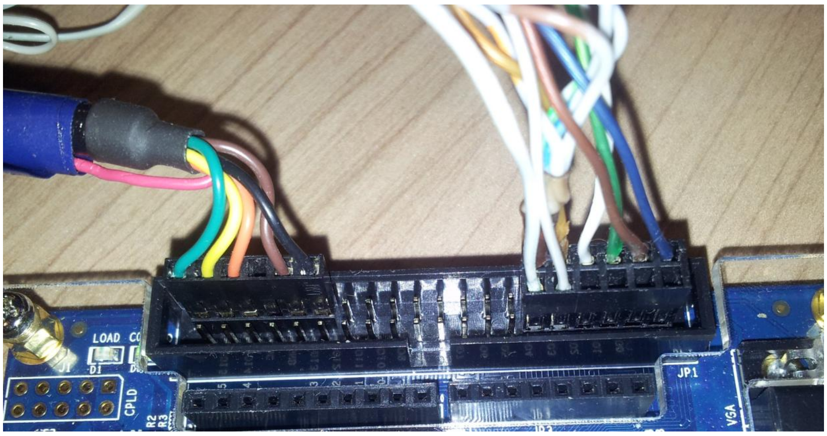
- Connect USB-to-UART pin RXD to the 4 pin (GPIO_D3) on the GPIO header
- Connect USB-to-UART pin TXD to the 6 pin (GPIO_D5) on the GPIO header
- Connect USB-to-UART pin GND to the 12 pin (GPIO_GND) on the GPIO header

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

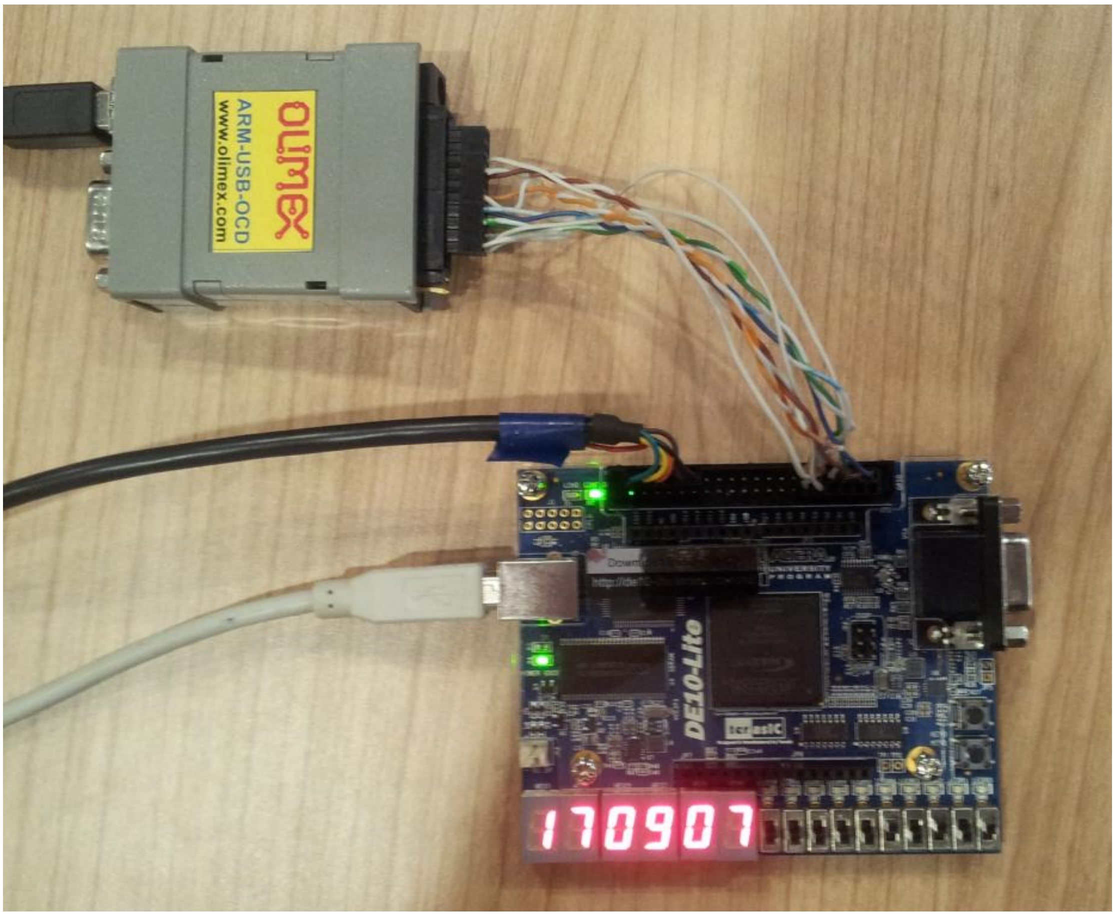
- Connect JTAG pin TCK to the 32 pin (GPIO_D27) on the GPIO header
- Connect JTAG pin TRSTn to the 34 pin (GPIO_D29) on the GPIO header
- Connect JTAG pin TDI to the 36 pin (GPIO_D31) on the GPIO header
- Connect JTAG pin TDO to the 38 pin (GPIO_D33) on the GPIO header
- Connect JTAG pin TMS to the 40 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



- Resulting setup



# 3. DE10-Lite flash image update

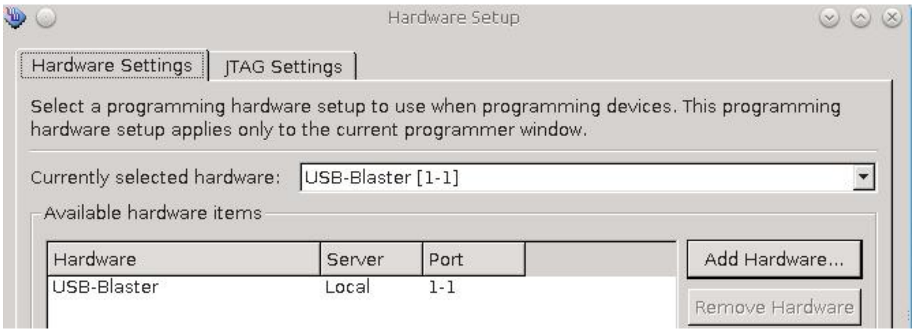
Image update procedure will load FPGA firmware to the FPGA flash memory (MAX10).
The FPGA firmware image in the flash memory is then loaded upon every board power on.
Binary file used for the update is in the Altera standard .pof format.

## 3.1. Required equpment

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

## 3.2. 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:



Hardware Setup

Hardware Settings | JTAG Settings

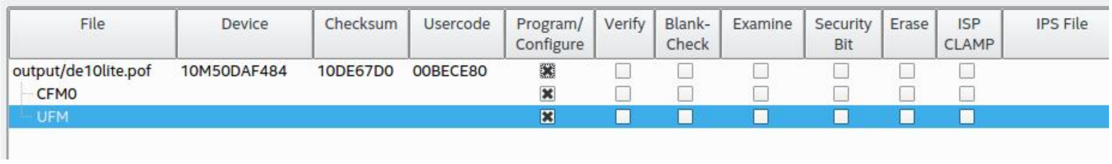
Select a programming hardware setup to use when programming devices. This programming hardware setup applies only to the current programmer window.

Currently selected hardware: USB-Blaster [1-1]

Available hardware items

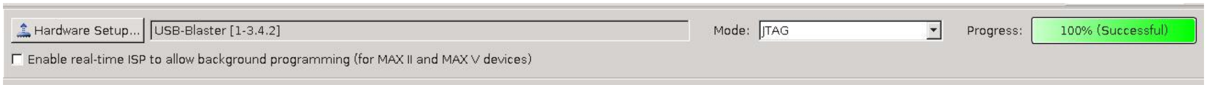
| Hardware | Server | Port |
|---|---|---|
| USB-Blaster | Local | 1-1 |

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 | ☒ | □ | □ | □ | □ | □ | □ | |
| CFM0 | | | | ☒ | □ | □ | □ | □ | □ | □ | |
| UFM | | | | ☒ | □ | □ | □ | □ | □ | □ | |

5. Wait for the loading to complete



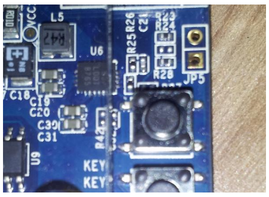
Hardware Setup... USB-Blaster [1-3.4.2] Mode: JTAG Progress: 100% (Successful)

□ Enable real-time ISP to allow background programming (for MAX II and MAX V devices)

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:



# 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 v1.0-scr1_RC
Copyright (C) 2015-2017 Syntacore. All rights reserved.
ISA: RV32IMC [40001104] IMPID: 17090600
BLDID: 17090700
Platform: de10lite_scr1, cpuclk 20MHz, sysclk 20MHz
Memory map:
00000000-003FFFFF       00000000        SDRAM
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
:
```

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

```
ISA: RV32IMC [40001104] IMPID: 17090600
BLDID: 17090700
Platform: de10lite_scr1, cpuclk 20MHz, sysclk 20MHz
Memory map:
00000000-003FFFFF       00000000        SDRAM
F0000000-F000FFFF       00000000        TCM
F0040000-F0040FFF       00000000        MTimer
FF000000-FF0FFFFF       00000000        MMIO
FFFF0000-FFFFFFFF       00000000        On-Chip RAM
Platform configuration:
FF010000        irq 0   UART16550
FF020000                Hex LED
FF021000                LED
FF022000                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 booloader 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
CCCCCCCCCCCCCC
```

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

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

5. 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                                                        |
|                                                                         |
```

6. 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...                         |
+-------------------------------------------------------------+
```

7. After loading completes, status information will be shown:

```
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
```

2. After run you will see test results

```
Time: begin= 258424349, end= 258432378, diff= 8029
Microseconds for one run through Dhrystone: 16.058
Dhrystones per Second:                      62274
```

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

## 7.1. Folder structure

- doc - SDK and SCR1 user guides
- fpga
  - arty
  - de10lite
    - scr1 - DE10-Lite FPGA project (project files + additional RTL IPs + bootloader image)
- images
  - arty
  - de10lite
    - scr1 - pre-built FPGA image
- scr1 - SCR1 repository, included as sub-module
  - src - SCR1 core RTL sources
- sw
  - fsbl - FPGA-bootloader
  - tests - some benchmark tests

Essential files: FPGA project file - de10lite.qpf (fpga/de10lite/scr1/de10lite.qpf)
Top module - de10lite (fpga/de10lite/scr1/de10lite.sv)

## 7.2. 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.<br>Some build steps may be different for other Quartus versions.<br>Free **Quartus Prime Lite Edition** is required for a non time-limited HW firmware generation for MAX10 FPGA devices. |

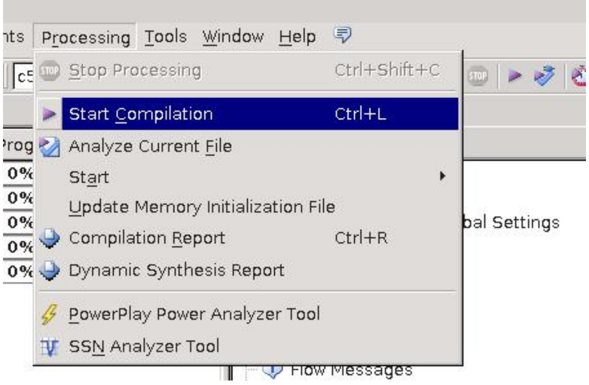
## 7.3. Building SDK FPGA project

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

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

- Run Quartus 17.0.1 in GUI-mode

- Select and open fpga-project file (de10lite.qpf)
- Press "Start Compilation" button or sellect from the menu Processing → "Start Compilation"



- Wait for the compilation to complete (build time is typically 10-15 minutes)
- New "output" subfolder should appear in the FPGA project folder (scr1). It contains de10lite.pof file (FPGA image in pof-format).

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

All common pins assignment is described in **DE10-Lite User manual**

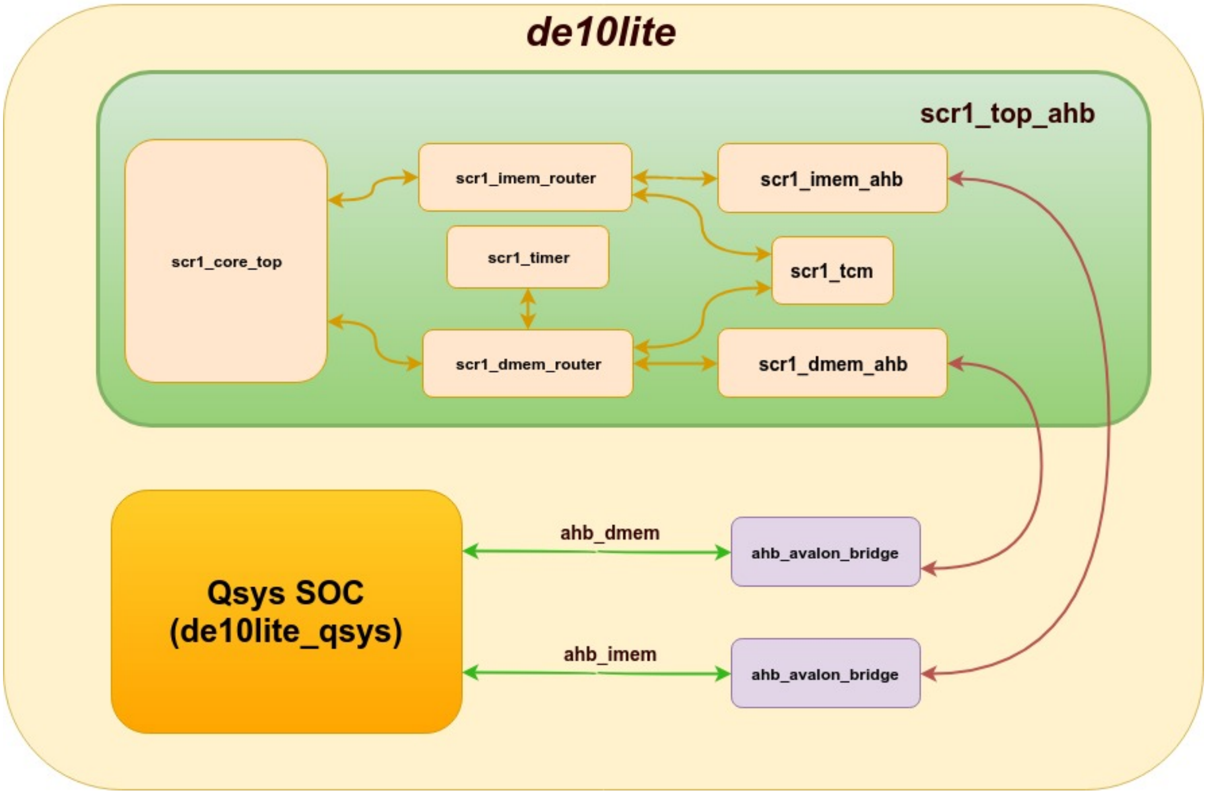
SDK-specific pin-assignment for UART and OpenOCD/JTAG are described in Table below:

| 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 |

# 7.4. SCR1 FPGA-project functional description

## 7.4.1. Common project structure

Functional schema for the project:

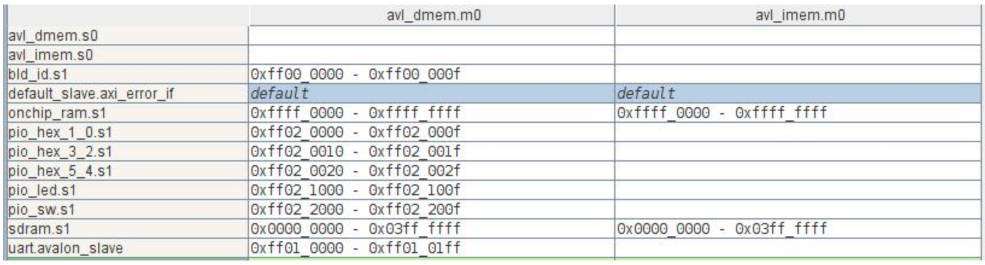


Modules description:

- **SCR1-core** (RTL sources in SystemVerilog)
- **Two data routers**
  (imem_route/dmem_router - instruction/data transfers, SystemVerilog RTL sources)
- **Two AHB-Avalon bridges**
  (ahb_imem/ahb_dmem - instruction/data transfers, SystemVerilog RTL sources)
- **scr1_timer block**
  (external timer block, SystemVerilog RTL sources)
- **scr1_tcm**
  (Tightly Coupled Memory (TCM), SystemVerilog RTL sources)
- **Qsys SoC block**
  (Qsys-component with generated SoC IPs)

## 7.4.2. Memory map for de10lite

Memory map is shown in the table below:



| | avl_dmem.m0 | avl_imem.m0 |
|---|---|---|
| avl_dmem.s0 | | |
| avl_imem.s0 | | |
| bld_id.s1 | 0xff00_0000 - 0xff00_000f | |
| default_slave.axi_error_if | *default* | *default* |
| onchip_ram.s1 | 0xffff_0000 - 0xffff_ffff | 0xffff_0000 - 0xffff_ffff |
| pio_hex_1_0.s1 | 0xff02_0000 - 0xff02_000f | |
| pio_hex_3_2.s1 | 0xff02_0010 - 0xff02_001f | |
| pio_hex_5_4.s1 | 0xff02_0020 - 0xff02_002f | |
| pio_led.s1 | 0xff02_1000 - 0xff02_100f | |
| pio_sw.s1 | 0xff02_2000 - 0xff02_200f | |
| sdram.s1 | 0x0000_0000 - 0x03ff_ffff | 0x0000_0000 - 0x03ff_ffff |
| uart.avalon_slave | 0xff01_0000 - 0xff01_01ff | |

### 7.4.3. SCR1 core IRQ-mapping

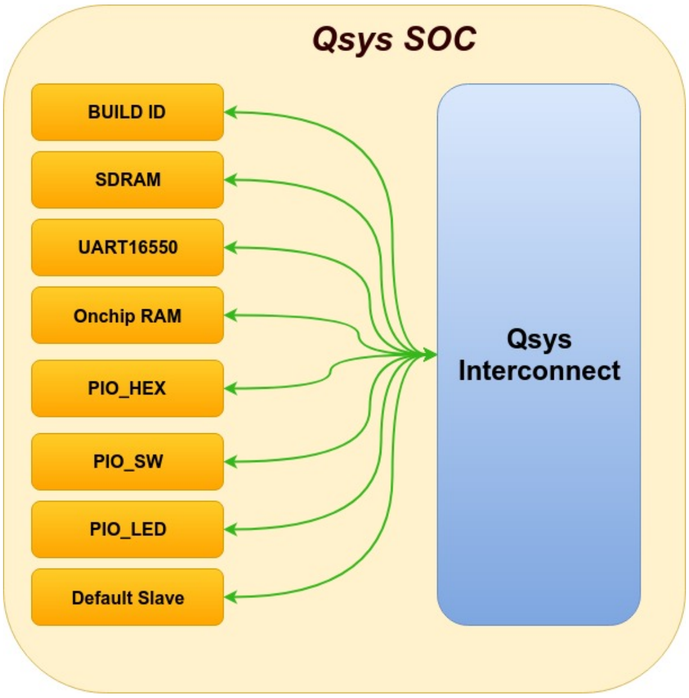
Table with IRQ lines mapping:

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

### 7.4.4. Qsys SOC structure

Qsys SOC contents the following blocks:

- BUILD ID
- SDRAM
- UART 16550
- Onchip RAM
- PIO HEX
- PIO SW
- PIO LED
- Qsys Default slave



## 7.4.5. SDK modules description

### 7.4.5.1. SCR1-core

Syntacore RISC-V core (SCR1 core RTL sources in SystemVerilog).
Please, refer to the "SCR1 External Architecture Specification" for detailed description.

### 7.4.5.2. AHB-Avalon bridge

AHB bridge converts internal imem/dmem bus inteface to the Altera Avalon imem/dmem bus interface. Provided in SV sources.

## 7.4.6. Qsys SoC IP-block descrption

Other Qsys modules desciption can be found at Altera site:

- Embedded Peripheral IP User Guide
- Qsys Interconnect
- SOPC Builder User Guide

### 7.4.6.1. BUILD ID

PIO-block with date of the project modification. Available for CPU to read.
Set by define statement in the de10lite block (de10lite.sv, parameter FPGA_DE10_BUILD_ID)
Base address: 0xFF000000.

### 7.4.6.2. PIO HEX

Three PIO-blocks for the 7-segments display control. Available for CPU write and read
Base address: 0xFF020000, 0xFF020010, 0xFF020020.

### 7.4.6.3. PIO LED

PIO-block for the LED indication control. Available for CPU write and read
Base address: 0xFF021000.

### 7.4.6.4. PIO SW

PIO-block for switches position read. Available for CPU read
Base address: 0xFF022000.

### 7.4.6.5. SDRAM

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

### 7.4.6.6. Onchip RAM

Internal RAM for the DE10-Lite bootloader. Initialized after reset with pre-built hex image (de10lite.hex).

Size - 64KB. Base address: 0xFFFF0000.

#### 7.4.6.7. UART 16550

UART block to implement SDK console interface.
UART interrupt is connected to irq[0] of the SCR1-core.
Base address: 0xFF010000.

#### 7.4.6.8. Qsys Default slave

Default slave responder for Qsys-interconnect.
Main function is to generate error response for any transaction to the unmapped address.

# 8. Appendix A. Software build instructions

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

## 8.1. SCR bootloader

### 8.1.1. Getting the sources

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

### 8.1.2. 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).

## 8.2. Zephyr OS

### 8.2.1. Getting the sources

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

### 8.2.2. 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.

## 8.3. SCR1 OpenOCD

### 8.3.1. Getting the sources

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

### 8.3.2. Building and using OpenOCD

Please, refer to the Syntacore OpenOCD wiki page for instructions: https://github.com/syntacore/openocd/wiki/OpenOCD-for-sc_riscv32