SoC-FPGA Design Guide

Real Time Embedded Systems

LAP – IC – EPFL

Version 0.13 (Preliminary)

René Beuchat

Table of Contents

[2 List of Figures 2](#_Toc398730265)

[3 Introduction 3](#_Toc398730266)

[4 Terasic DE1-SoC Board 4](#_Toc398730267)

[4.1 Specifications 4](#_Toc398730268)

[4.1.1 FPGA Device 4](#_Toc398730269)

[4.1.2 Configuration and Debug 4](#_Toc398730270)

[4.1.3 Memory Device 4](#_Toc398730271)

[4.1.4 Communication 4](#_Toc398730272)

[4.1.5 Connectors 5](#_Toc398730273)

[4.1.6 Display 5](#_Toc398730274)

[4.1.7 Audio 5](#_Toc398730275)

[4.1.8 Video Input 5](#_Toc398730276)

[4.1.9 ADC 5](#_Toc398730277)

[4.1.10 Switches, Buttons and Indicators 5](#_Toc398730278)

[4.1.11 Sensors 5](#_Toc398730279)

[4.1.12 Power 5](#_Toc398730280)

[4.1.13 Block Diagram 6](#_Toc398730281)

[4.2 Layout 6](#_Toc398730282)

[5 Cyclone V Overview 8](#_Toc398730283)

[5.1 Introduction to the Cyclone V Hard Processor System 8](#_Toc398730284)

[5.2 Features of the HPS 9](#_Toc398730285)

[5.3 System Integration Overview 10](#_Toc398730286)

[5.3.1 MPU Subsystem 10](#_Toc398730287)

[5.3.2 SDRAM Controller Subsystem 10](#_Toc398730288)

[5.3.3 Support Peripherals 10](#_Toc398730289)

[5.3.4 Interface Peripherals 11](#_Toc398730290)

[5.3.5 On-Chip Memory 11](#_Toc398730291)

[5.4 HPS-FPGA Interfaces 11](#_Toc398730292)

[5.5 HPS Address Map 12](#_Toc398730293)

[5.5.1 HPS Address Spaces 12](#_Toc398730294)

[5.5.2 HPS Peripheral Region Address Map 13](#_Toc398730295)

[6 Using the Cyclone V 15](#_Toc398730296)

[6.1 Hardware 15](#_Toc398730297)

[6.1.1 FPGA-only 15](#_Toc398730298)

[6.1.2 SoC-only 15](#_Toc398730299)

[6.1.3 SoC & FPGA 15](#_Toc398730300)

[6.2 Software 15](#_Toc398730301)

[7 todo 15](#_Toc398730302)

[9 SoC part test 16](#_Toc398730303)

[9.1 HPS Architecture 16](#_Toc398730304)

[9.2 Hardware development 16](#_Toc398730305)

[9.2.1 Qsys integration 16](#_Toc398730306)

[9.3 Software development 18](#_Toc398730307)

[9.3.1 ARM DS-5 tools 18](#_Toc398730308)

[9.3.2 Hello World on ARM HPS part 18](#_Toc398730309)

[9.3.3 GPIO access 19](#_Toc398730310)

# List of Figures

[Figure 1. Terasic DE1-SoC Board 5](#_Toc398672830)

[Figure 2. Block Diagram of the DE1-SoC Board 7](#_Toc398672831)

[Figure 3. Front 7](#_Toc398672832)

[Figure 4. Back 8](#_Toc398672833)

[Figure 5. Altera SoC FPGA Device Block Diagram 9](#_Toc398672834)

[Figure 6. HPS Block Diagram 10](#_Toc398672835)

[Figure 7. HPS Address Space Relations 13](#_Toc398672836)

# Introduction

The development of embedded systems based on chips containing one or more microprocessors and hardcore peripherals, as well as an FPGA part is becoming more and more important. This technology gives the designer a lot of freedom and powerful abilities. Classical design flows with microcontrollers are emphasized with the full power of FPGAs.

Mixed designs are becoming a reality with. One can now design specific accelerators to greatly improve algorithms, or create specific programmable interfaces with the external world.

Two main HDL (**H**ardware **D**esign **L**anguage) languages are available for the design of the FPGA part: **VHDL** and Verilog. There also exist other tools that perform automatic translations from C to HDL. New emerging technologies like OpenCL allow compatibility between high-level software design, and low-level hardware implementations as:

* Compilation for single or multicore processors
* Compilation for GPUs (Graphical Processing Unit)
* Translation and compilation for FPGAs. The latest models use a PCIe interface or some other way of parameters passing between the main processor and the FPGA

*This guide assumes users know how to use Quartus II, Nios II, Qsys and ModelSim-Altera.*

We will be using the Terasic DE1-SoC board: <http://de1-soc.terasic.com>

# Terasic DE1-SoC Board

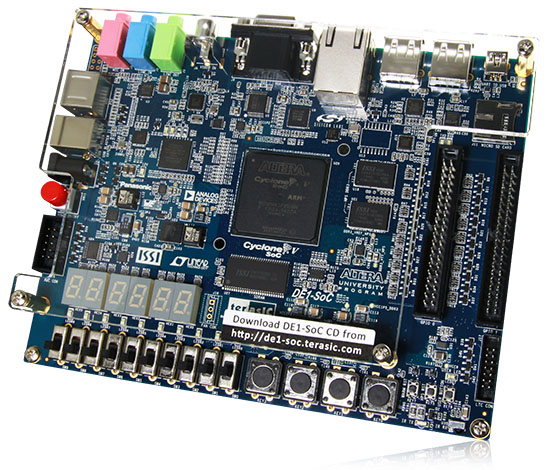


Figure . Terasic DE1-SoC Board

The DE1-SoC board has many features that allow users to implement a wide range of designed circuits. We will discuss some noteworthy features in this guide.

## Specifications

### FPGA Device

* Cyclone V SoC 5CSEMA5F31C6 Device
* Dual-core ARM Cortex-A9 (HPS)
* 85K Programmable Logic Elements
* 4’450 Kbits embedded memory
* 6 Fractional PLLs
* 2 Hard Memory Controllers

### Configuration and Debug

* Quad Serial Configuration device – EPCQ256 on FPGA
* On-Board USB Blaster II (Normal type B USB connector)

### Memory Device

* 64 MB (32Mx16) SDRAM on FPGA
* 1 GB (2x256Mx16) DDR3 SDRAM on HPS
* Micro SD Card Socket on HPS

### Communication

* Two Port USB 2.0 Host (ULPI interface with USB type A connector)
* USB to UART (micro USB type B connector)
* 10/100/1000 Ethernet
* PS/2 mouse/keyboard
* IR Emitter/Receiver

### Connectors

* Two 40-pin Expansion Headers
* One 10-pin ADC Input Header
* One LTC connector (One Serial Peripheral Interface (SPI) Master, one I2C and one GPIO interface)

### Display

* 24-bit VGA DAC

### Audio

* 24-bit CODEC, line-in, line-out, and microphone-in jacks

### Video Input

* TV Decoder (NTSC/PAL/SECAM) and TV-in connector

### ADC

* Fast throughput rate: 1 MSPS
* Channel number: 8
* Resolution: 12 bits
* Analog input range : 0 ~ 2.5 V or 0 ~ 5V as selected via the RANGE bit in the control register

### Switches, Buttons and Indicators

* 4 User Keys (FPGA x4)
* 10 User switches (FPGA x10)
* 11 User LEDs (FPGA x10; HPS x 1)
* 2 HPS Reset Buttons (HPS\_RST\_n and HPS\_WARM\_RST\_n)
* Six 7-segment displays

### Sensors

* G-Sensor on HPS

### Power

* 12V DC input

### Block Diagram

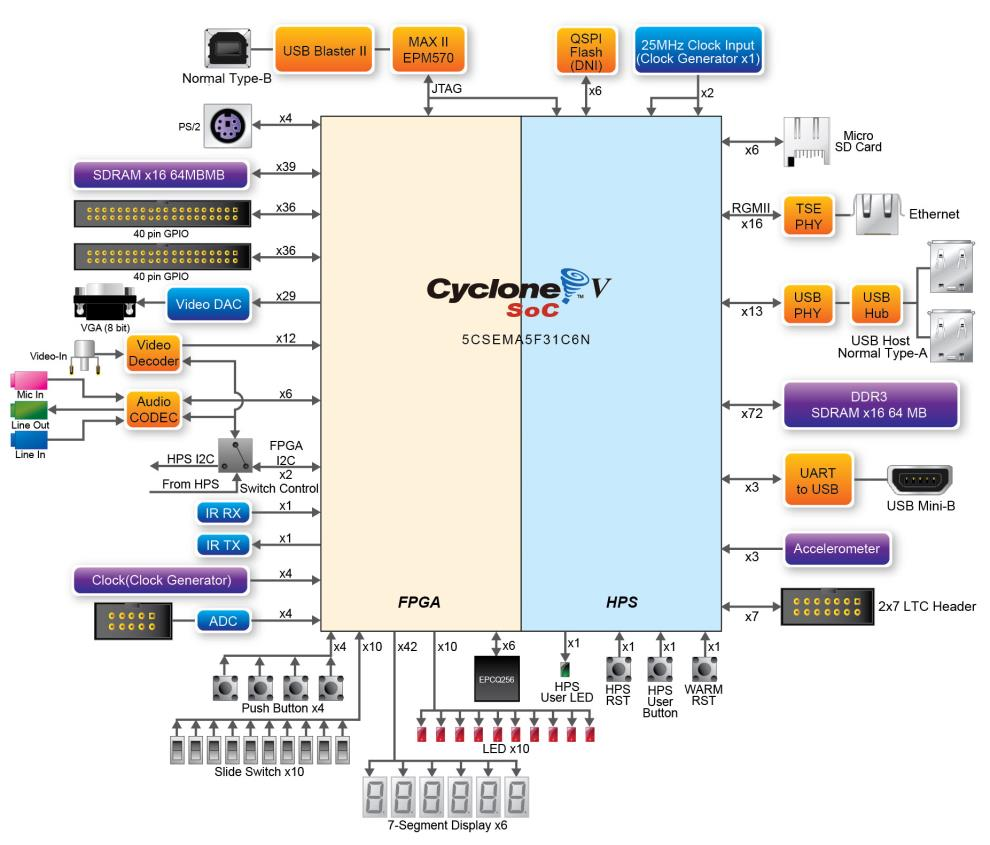


Figure . Block Diagram of the DE1-SoC Board

## Layout

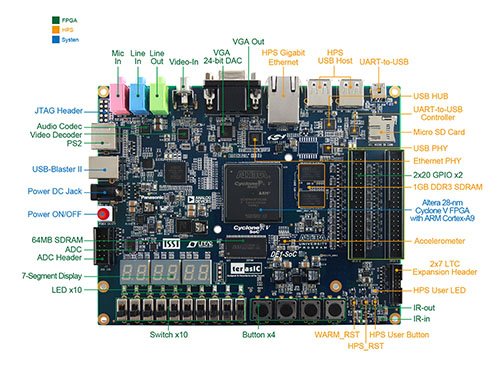


Figure . Front

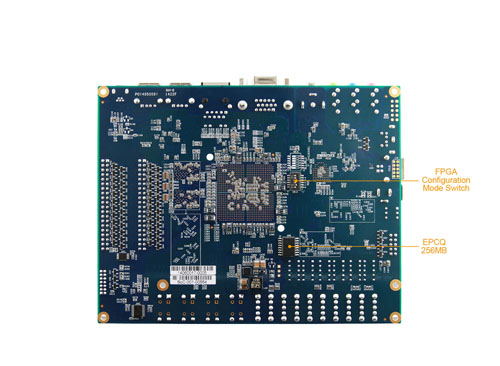


Figure . Back

* Green for peripherals directly connected to the FPGA
* Orange for peripherals directly connected to the HPS
* Blue for board control

Manuals and resources are available on the DE1-SoC [resources](http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=165&No=836&PartNo=4ParagrapheTexte) page.

# Cyclone V Overview

This section describes some features of the Cyclone V family of devices. We do not list all features, but only the ones most important to us. All this information, along with the most complete documentation regarding this family can be found on the [Cyclone V Device Handbook](http://www.altera.com/literature/hb/cyclone-v/cyclone5_handbook.pdf), more specifically [Volume 3: Hard Processor System Technical Reference Manual](http://www.altera.com/literature/hb/cyclone-v/cv_5v4.pdf).

## Introduction to the Cyclone V Hard Processor System

The Cyclone V device is a single-die system on a chip (SoC) that consists of two distinct parts – a hard processor system (HPS) portion and an FPGA portion.



Figure . Altera SoC FPGA Device Block Diagram

The HPS contains a microprocessor unit (MPU) subsystem with single or dual ARM Cortex-A9 MPCore processors, flash memory controllers, SDRAM L3 Interconnect, on-chip memories, support peripherals, interface peripherals, debug capabilities, and phase-locked loops (PLLs). The dual-processor HPS supports symmetric (SMP) and asymmetric (AMP) multiprocessing.

The DE1-SoC has a dual-processor HPS.

The FPGA portion of the device contains the FPGA fabric, a control block (CB), phase-locked loops (PLLs), and depending on the device variant, high-speed serial interface (HSSI) transceivers, hard PCI Express (PCIe) controllers, and hard memory controllers.

The DE1-SoC does not contain any HSSI transceivers, or hard PCIe controllers.

The HPS and FPGA portions of the device are distinctly different. The HPS can boot from multiple sources, including the FPGA fabric and external flash. In contrast, the FPGA must be configured through either the HPS or an externally supported device.

The MPU subsystem can boot from flash devices connected to the HPS pins. Or, when the FPGA portion is configured by an external source, the MPU subsystem can boot from memory available on the FPGA portion of the device.

The HPS and FPGA portions of the device each have their own pins. Pins are not freely shared between the HPS and the FPGA fabric. The FPGA I/O pins are configured by an FPGA configuration image through the HPS or any external source supported by the device. The HPS I/O pins are configured by software executing in the HPS. Software executing on the HPS accesses control registers in the system manager to assign HPS I/O pins to the available HPS modules.

The software that configures the HPS I/O pins is called the Preloader.

The HPS and FPGA portions of the device have separate external power supplies and independently power on. You can power on the HPS without powering on the FPGA portion of the device. However, to power on the FPGA portion, the HPS must already be on or powered on at the same time as the FPGA portion. You can also turn off the FPGA portion of the device while leaving the HPS power on.

## Features of the HPS



Figure . HPS Block Diagram

The following list contains the main modules of the HPS:

* MPU subsystem featuring dual ARM Cortex-A9 MPCore processors
* General-purpose Direct Memory Access (DMA) controller
* Two Ethernet media access controllers (EMACs)
* Two USB 2.0 On-The-Go (OTG) controllers
* NAND flash controller
* Quad SPI flash controller
* Secure Digital (SD) / MultiMediaCard (MMC) controller
* Two serial peripheral interface (SPI) master controllers
* Two SPI slave controllers
* Four inter-integrated circuit (I2C) controllers
* 64 KB on-chip RAM
* 64 KB on-chip boot ROM
* Two UARTs
* Four timers
* Two watchdog timers
* Three general-purpose I/O (GPIO) interfaces
* Two controller area network (CAN) controllers
* ARM CoreSight debug components
* System manager
* Clock manager
* Reset manager
* Scan manager
* FPGA manager

## System Integration Overview

In this part, we briefly go through some features provided by the most important HPS components.

### MPU Subsystem

Here are a few important features of the MPU subsystem:

* Interrupt controller
* One general-purpose timer and one watchdog timer per processor
* One Memory management unit (MMU) per processor

The HPS masters the L3 interconnect and the SDRAM controller subsystem.

### SDRAM Controller Subsystem

The SDRAM controller subsystem is mastered by HPS masters and FPGA fabric masters. It supports DDR2, DDR3, and LPDDR2 devices. It is composed of 2 parts:

* SDRAM controller
* DDR PHY (interfaces the single port memory controller to the HPS I/O)

The DE1-SoC contains DDR3 SDRAM on the HPS

### Support Peripherals

#### System Manager

This is one of the most essential HPS components. It offers a few important features:

* Pin multiplexing (term used for the software configuration of the HPS I/O pins by the Preloader)
* Freeze controller that places I/O elements into a safe state for configuration
* Low-level control of peripheral features not accessible through the control and status registers (CSRs)

The low-level control of some peripheral features that are not accessible through the CSRs is not externally documented. You will see this type of code when you generate your custom preloader, but must not use the constructs in your own code.

#### FPGA Manager

The FPGA manager offers the following features:

* Manages configuration of the FPGA portion of the device
* Monitors configuration-related signals in the FPGA
* Provides 32 general-purpose inputs and 32 general-purpose outputs to the FPGA fabric

### Interface Peripherals

#### GPIO Interfaces

The HPS provides three GPIO interfaces and offer the following features:

* Supports digital de-bounce
* Configurable interrupt mode
* Supports up to 71 I/O pins and 14 input-only pins, based on device variant
* Supports up to 67 I/O pins and 14 input-only pins

The DE1-SoC has 67 I/O pins and 14 input-only pins

### On-Chip Memory

The following on-chip memories are different from any on-chip memories located in the FPGA fabric.

#### On-Chip RAM

The on-chip RAM offers the following features:

* 64 KB size
* High performance for all burst lengths

#### Boot ROM

The boot ROM offers the following features:

* 64 KB size
* Contains the code required to support HPS boot from cold or warm reset
* Used exclusively for booting the HPS

The code in the boot ROM cannot be changed.

## HPS-FPGA Interfaces

The HPS-FPGA interfaces provide a variety of communication channels between the HPS and the FPGA fabric. The HPS-FPGA interfaces include:

* FPGA-to-HPS bridge – a high performance bus with a configurable data width of 32, 64, or 128 bits. It allows the FPGA fabric to master transactions to slaves in the HPS. This interface allows the FPGA fabric to have full visibility into the HPS address space.
* HPS-to-FPGA bridge – a high performance bus with a configurable data width of 32, 64, or 128 bits. It allows the HPS to master transactions to slaves in the FPGA fabric. I will sometimes call this the “heavyweight” HPS-to-FPGA bridge to distinguish its “lightweight” counterpart (see below).
* Lightweight HPS-to-FPGA bridge – a bus with a 32-bit fixed data width. It allows the HPS to master transactions to slaves in the FPGA fabric.
* FPGA manager interface – signals that communicate with FPGA fabric for boot and configuration.
* Interrupts – allow soft IP to supply interrupts directly to the MPU interrupt controller.
* HPS debug interface – an interface that allows the HPS debug control domain to extend into the FPGA.

## HPS Address Map

### HPS Address Spaces

The HPS address map specifies the address of slaves, such as memory and peripherals, as viewed by the HPS masters. The HPS has 3 address spaces:

|  |  |  |
| --- | --- | --- |
| Name | Description | Size |
| MPU | MPU subsystem | 4 GB |
| L3 | L3 interconnect | 4 GB |
| SDRAM | SDRAM controller subsystem | 4 GB |

Table . HPS Address Spaces

The following figure shows the relationships between the different HPS address spaces. The figure is not to scale.



Figure . HPS Address Space Relations

The window regions provide access to other address spaces. The thin black arrows indicate which address space is accessed by a window region (arrows point to accessed address space).

The SDRAM window in the MPU can grow and shrink at the top and bottom (short blue vertical arrows) at the expense of the FPGA slaves and boot regions. The ACP window can be mapped to any 1 GB region in the MPU address space (blue vertical bidirectional arrow), on gigabyte-aligned boundaries.

The following table shows the base address and size of each region that is common to the L3 and MPU address spaces.

|  |  |  |  |
| --- | --- | --- | --- |
| Region Name | Description | Base Address | Size |
| FPGA slaves | FPGA slaves connected to the HPS-to-FPGA bridge | 0xC0000000 | 960 MB |
| HPS peripherals | Slaves directly connected to the HPS (corresponds to all orange colored elements on Figure 3 and Figure 4) | 0xFC000000 | 64 MB |
| Lightweight FPGA slaves | FPGA slaves connected to the lightweight HPS-to-FPGA bridge | 0xFF200000 | 2 MB |

Table . Common Address Space Regions

### HPS Peripheral Region Address Map

The following table lists the slave identifier, slave title, base address, and size of each slave in the HPS peripheral region. The Slave Identifier column lists the names used in the HPS register map file provided by Altera (more on this later).

|  |  |  |  |
| --- | --- | --- | --- |
| Slave Identifier | Slave Title | Base Address | Size |
| STM | STM | 0xFC000000 | 48 MB |
| DAP | DAP | 0xFF000000 | 2 MB |
| LWFPGASLAVES | FPGA slaves accessed with lightweight HPS-to-FPGA bridge | 0xFF200000 | 2 MB |
| LWHPS2FPGAREGS | Lightweight HPS-to-FPGA bridge GPV | 0xFF400000 | 1 MB |
| HPS2FPGAREGS | HPS-to-FPGA bridge GPV | 0xFF500000 | 1 MB |
| FPGA2HPSREGS | FPGA-to-HPS bridge GPV | 0xFF600000 | 1 MB |
| EMAC0 | EMAC0 | 0xFF700000 | 8 KB |
| EMAC1 | EMAC1 | 0xFF702000 | 8 KB |
| SDMMC | SD/MMC | 0xFF704000 | 4 KB |
| QSPIREGS | Quad SPI flash controller registers | 0xFF705000 | 4 KB |
| FPGAMGRREGS | FPGA manager registers | 0xFF706000 | 4 KB |
| ACPIDMAP | ACP ID mapper registers | 0xFF707000 | 4 KB |
| GPIO0 | GPIO0 | 0xFF708000 | 4 KB |
| GPIO1 | GPIO1 | 0xFF709000 | 4 KB |
| GPIO2 | GPIO2 | 0xFF70A000 | 4 KB |
| L3REGS | L3 interconnect GPV | 0xFF800000 | 1 MB |
| NANDDATA | NAND controller data | 0xFF900000 | 1 MB |
| QSPIDATA | Quad SPI flash data | 0xFFA00000 | 1 MB |
| USB0 | USB0 OTG controller registers | 0xFFB00000 | 256 KB |
| USB1 | USB1 OTG controller registers | 0xFFB40000 | 256 KB |
| NANDREGS | NAND controller registers | 0xFFB80000 | 64 KB |
| FPGAMGRDATA | FPGA manager configuration data | 0xFFB90000 | 4 KB |
| CAN0 | CAN0 controller registers | 0xFFC00000 | 4 KB |
| CAN1 | CAN1 controller registers | 0xFFC01000 | 4 KB |
| UART0 | UART0 | 0xFFC02000 | 4 KB |
| UART1 | UART1 | 0xFFC03000 | 4 KB |
| I2C0 | I2C0 | 0xFFC04000 | 4 KB |
| I2C1 | I2C1 | 0xFFC05000 | 4 KB |
| I2C2 | I2C2 | 0xFFC06000 | 4 KB |
| I2C3 | I2C3 | 0xFFC07000 | 4 KB |
| SPTIMER0 | SP Timer0 | 0xFFC08000 | 4 KB |
| SPTIMER1 | SP Timer1 | 0xFFC09000 | 4 KB |
| SDRREGS | SDRAM controller subsystem registers | 0xFFC20000 | 128 KB |
| OSC1TIMER0 | OSC1 Timer0 | 0xFFD00000 | 4 KB |
| OSC1TIMER1 | OSC1 Timer1 | 0xFFD01000 | 4 KB |
| L4WD0 | Watchdog0 | 0xFFD02000 | 4 KB |
| L4WD1 | Watchdog1 | 0xFFD03000 | 4 KB |
| CLKMGR | Clock manager | 0xFFD04000 | 4 KB |
| RSTMGR | Reset manager | 0xFFD05000 | 4 KB |
| SYSMGR | System manager | 0xFFD08000 | 16 KB |
| DMANONSECURE | DMA nonsecure registers | 0xFFE00000 | 4 KB |
| DMASECURE | DMA secure registers | 0xFFE01000 | 4 KB |
| SPIS0 | SPI slave0 | 0xFFE02000 | 4 KB |
| SPIS1 | SPI slave1 | 0xFFE03000 | 4 KB |
| SPIM0 | SPI master0 | 0xFFF00000 | 4 KB |
| SPIM1 | SPI master1 | 0xFFF01000 | 4 KB |
| SCANMGR | Scan manager registers | 0xFFF02000 | 4 KB |
| ROM | Boot ROM | 0xFFFD0000 | 64 KB |
| MPUSCU | MPU SCU registers | 0xFFFEC000 | 8 KB |
| MPUL2 | MPU L2 cache controller registers | 0xFFFEF000 | 4 KB |
| OCRAM | On-chip RAM | 0xFFFF0000 | 64 KB |

Table . HPS Peripheral Region Address Map

The programming model for accessing the HPS peripherals in Table 3 is the same as for peripherals created on the FPGA fabric. That is, every peripheral has a base address at which a certain number of registers can be found. You then read and write to a certain set of these registers in order to modify the peripheral’s behavior.

You do not need to hard-code any base addresses or peripheral register maps in your programs, as Altera provides a header file for each HPS peripheral in Table 3.

Two directories contain all HPS-related header files:

* “<altera\_install\_directory>\embedded\ip\altera\hps\altera\_hps\hwlib\include”

Contains high-level header files that typically contain a few functions which facilitate control over the HPS components.

* “<altera\_install\_directory>\embedded\ip\altera\hps\altera\_hps\hwlib\include\socal”

Contains low-level header files that provide a peripheral’s bit-level register details. For example, any bits in a peripheral’s register that correspond to undefined behavior will be specified in these header files.

To illustrate the differences among the high and low-level header files, we can compare the ones related to the FPGA manager peripheral:

* “…\hwlib\include\alt\_fpga\_manager.h”

1. ALT\_STATUS\_CODE alt\_fpga\_reset\_assert(void);
2. ALT\_STATUS\_CODE alt\_fpga\_configure(const void\* cfg\_buf, size\_t cfg\_buf\_len);

* “…\hwlib\include\socal\alt\_fpgamgr.h”

1. /\* The width in bits of the ALT\_FPGAMGR\_CTL\_EN register field. \*/
2. #define ALT\_FPGAMGR\_CTL\_EN\_WIDTH 1
3. /\* The mask used to set the ALT\_FPGAMGR\_CTL\_EN register field value. \*/
4. #define ALT\_FPGAMGR\_CTL\_EN\_SET\_MSK 0x00000001
5. /\* The mask used to clear the ALT\_FPGAMGR\_CTL\_EN register field value. \*/
6. #define ALT\_FPGAMGR\_CTL\_EN\_CLR\_MSK 0xfffffffe

An important header file is “…\hwlib\include\socal\hps.h”. It contains the HPS component’s full register map, as provided in Table 3. Note however, that there exists no header file for the “heavyweight” HPS-to-FPGA bridge, as it is not located in the “HPS peripherals” region in Figure 7. Indeed, the “heavyweight” HPS-to-FPGA bridge is not considered a HPS peripheral, whereas the lightweight HPS-to-FPGA bridge is.

Therefore, in order to use the “heavyweight” HPS-to-FPGA bridge, you will have to define a macro in your code, as follows:

1. #define ALT\_HWFPGASLVS\_OFST 0xc0000000

The reason why the “lightweight” HPS-to-FPGA bridge is considered a HPS peripheral may be related to the fact that it has a fixed 32-bit bus width (coincidence that this corresponds to the HPS’ native data size?)

# Using the Cyclone V

## Hardware

The HPS component is a soft component, but it does not mean that the HPS is a softcore processor. In fact, the HPS exclusively contains hard logic. The reason it is considered a softcore component originates from the fact that it enables other soft components to interface with the HPS hard logic. As such, the HPS component has a small footprint in the FPGA fabric, as its only purpose is to connect the soft and hard logic together.

Therefore, it is possible to use the Cyclone V SoC in 3 different configurations:

* FPGA-only
* HPS-only
* HPS & FPGA

We will look at these different configurations below. The HPS configurations are more difficult to configure than the FPGA-only one.

### FPGA-only

### HPS-only

Bare-metal vs Linux

Requires and explanation of “booting”

No hardware setup is necessary. You can simply boot off of the pre-made

### HPS & FPGA

## Software

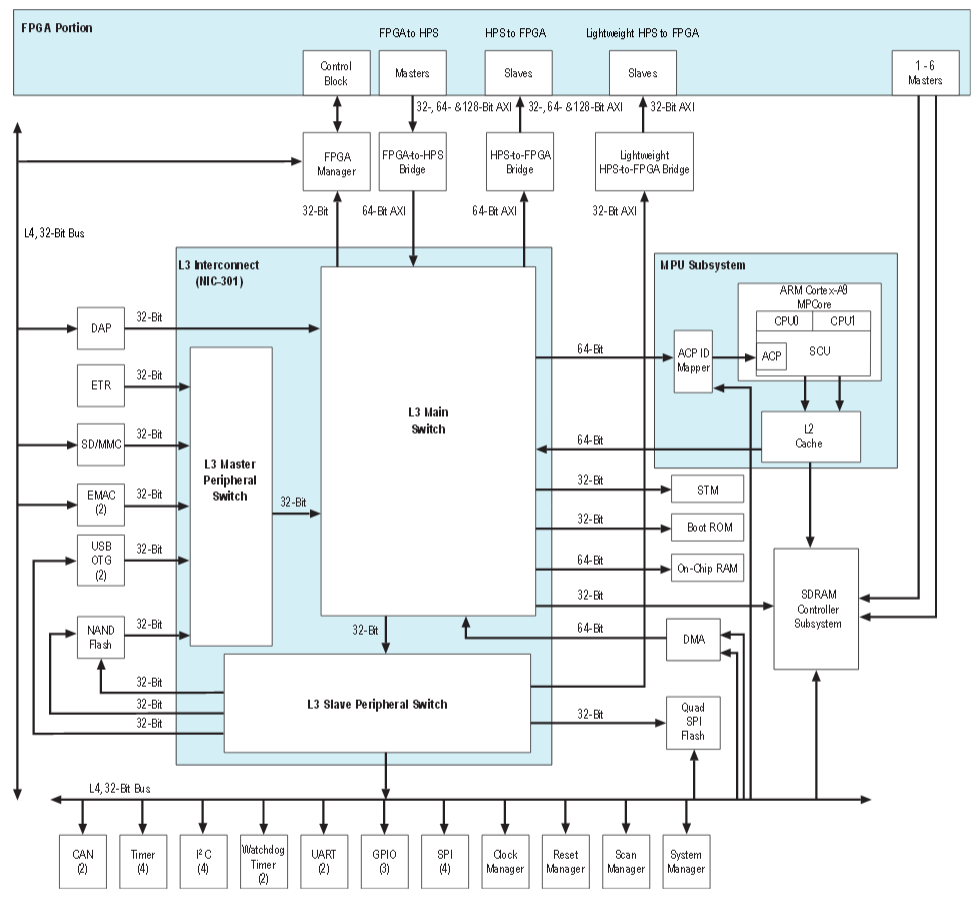
# todo

Find a way to write code in word, otherwise it will get annoying.

# SoC part test

## HPS Architecture

To be able to program the ARM9’s processors it is almost necessary to have the global view of the HPS architecture.



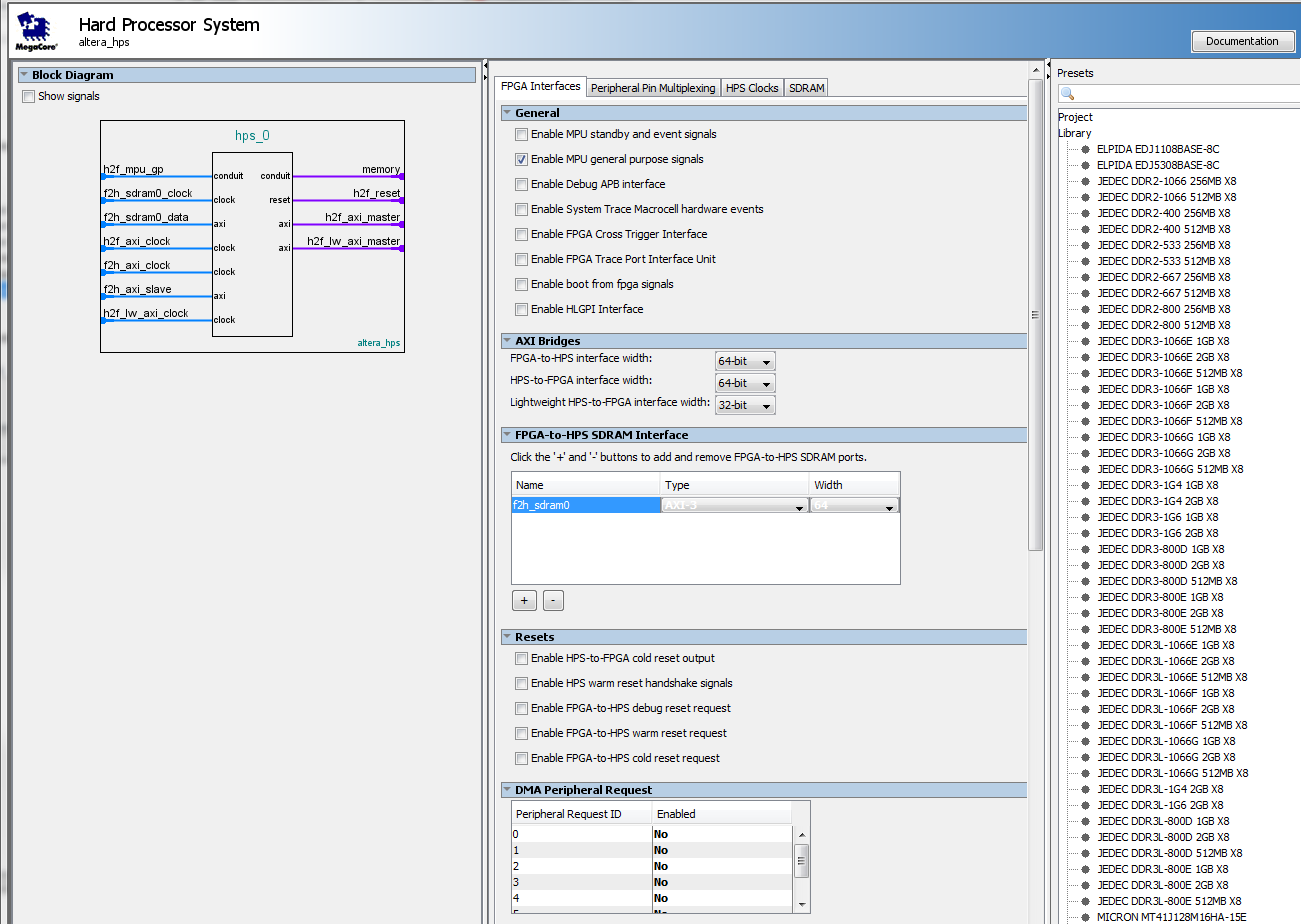
## Hardware development

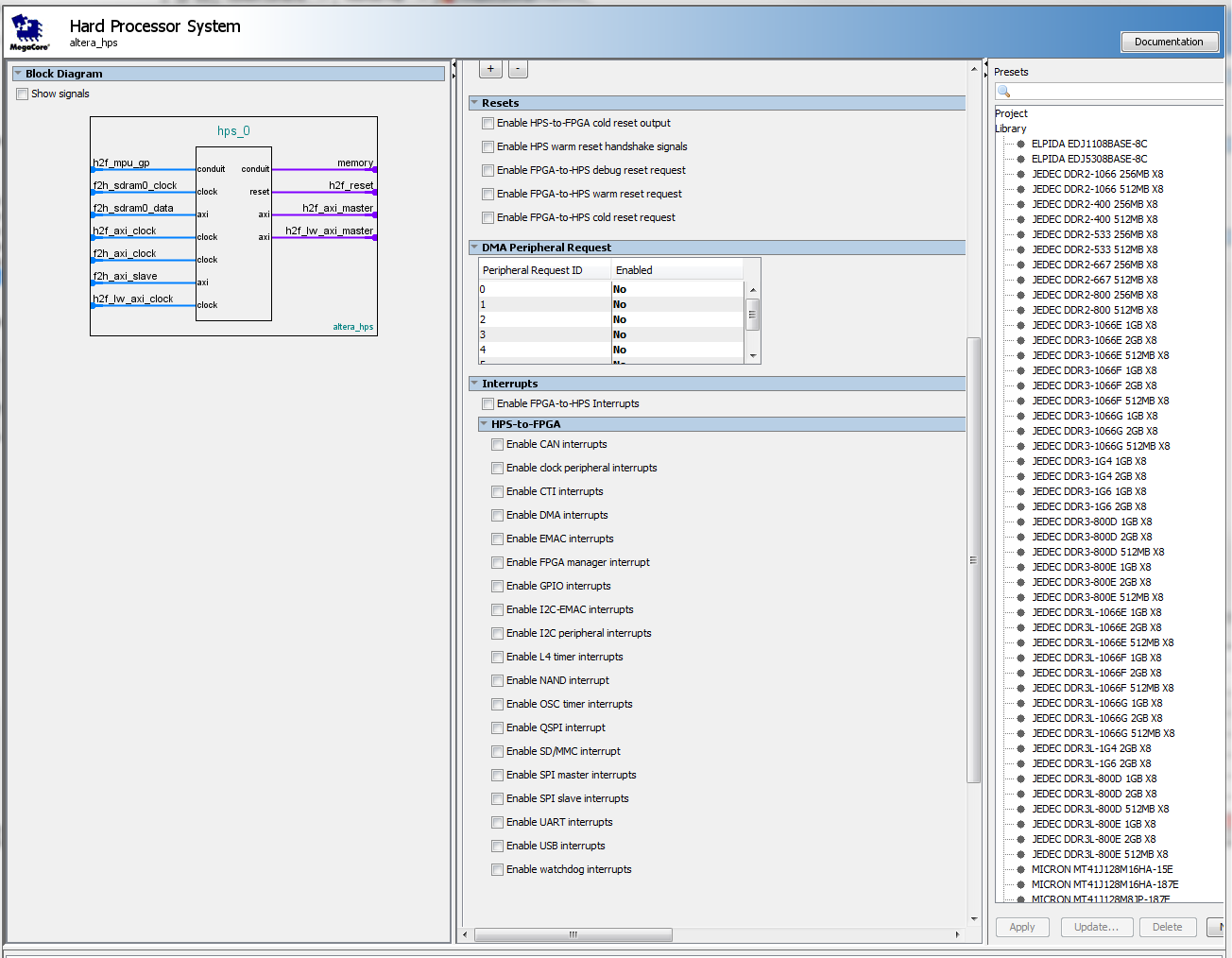
### Qsys integration

Starting with **QuartusII** and after creating a project, select ***Tools🡪 Qsys***

In **Qsys**, open ***Library 🡪 Embedded Processors 🡪 Hard Processor System*** the window with description of the parameters for the HPS is open.

The ***FPGA Interface*** tab allows the access from to the FPGA part with the HPS part.





With the ***PeripheralPin Multiplexing***, some I/O interface can be used by the HPS part or the FPGA part. The selection is done here.

## Software development

### ARM DS-5 tools

They are some differences between the versions of DS-5.

The one installed for the test is:

ARM DS-5 (DS-5 Altera Edition (Evaluation))

Version: 5.18.0

Build number: 5180018

### Hello World on ARM HPS part

Copy the directory from Altera examples:

C:\altera\13.1\embedded\examples\software

And un-gz the file: Altera-SoCFPGA-HelloWorld-Baremetal-ARMCC.tar.gz

Then un-tar it.

The directory **Altera-SoCFPGA-HelloWorld-Baremetal-ARMCC** can then be copied in the Eclipse WorkSpace and Imported as a new project. The files inside are:

* .cproject used by Eclipse
* . project used by Eclipse
* \*\*\*\*.launch ??
* Makefile for the Compiler/Assembler/Linker  
   An important info is the flag for the cpu: --cpu=Cortex-A9.no\_neon.no\_vfp
* scatter.scat Info for the compiler for the Code, Data, Stack and Heap addresses  
   in this case in the internal SRAM

#### Scatter.scat

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Copyright (c) 2013 Altera All Rights Reserved.

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Scatter-file for OnChip RAM based example

; This scatter-file places application code, data, stack and heap at suitable addresses in the memory map.

; Altera SoC-FPGA has **64kB of internal OnChip RAM**

**OCRAM 0xFFFF0000 0x10000**

{

APP\_CODE +0

{

\* (+RO, +RW, +ZI)

}

ARM\_LIB\_STACKHEAP 0xFFFF8000 EMPTY 0x8000 ; Application heap and stack

{ }

}

#### Makefile

Makefile for the ARM compiler

# Copyright (C) ARM Limited, 2011. All rights reserved.

#

# This example is intended to be built with the ARM Compiler armcc

TARGET=Altera-SoCFPGA-HelloWorld-Baremetal-ARMCC.axf

CC=armcc

AS=armasm

LD=armlink

AR=armar

# Select build rules based on Windows or Unix

ifdef WINDIR

DONE=@if exist $(1) echo Build completed.

RM=if exist $(1) del /q $(1)

SHELL=$(WINDIR)\system32\cmd.exe

else

ifdef windir

DONE=@if exist $(1) echo Build completed.

RM=if exist $(1) del /q $(1)

SHELL=$(windir)\system32\cmd.exe

else

DONE=@if [ -f $(1) ]; then echo Build completed.; fi

RM=rm -f $(1)

endif

endif

all: $(TARGET)

$(call DONE,$(TARGET))

rebuild: clean all

clean:

$(call RM,\*.o)

$(call RM,$(TARGET))

hello.o: hello.c

$(CC) -c -g --cpu=Cortex-A9.no\_neon.no\_vfp -O0 hello.c

$(TARGET): hello.o scatter.scat

$(LD) hello.o -o $(TARGET) --cpu=Cortex-A9.no\_neon.no\_vfp --scatter=scatter.scat

### GPIO access

The references for gpio are:

* <http://www.altera.com/literature/hb/cyclone-v/cv_54022.pdf>
* <http://www.altera.com/literature/hb/cyclone-v/hps.html>
* Supports up to 71 I/O pins and 14 input-only pins depend on device variant

On de1-soc:

* Only 1 Button for HPS GPIO 1
* Only 1 LED for HPS GPIO 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Pin Name** | **HPS GPIO** | **Register [bit]** | **Function** | **Address** | **Dir** |
| HPS\_KEY | GPIO54 | GPIO1[25] | I/O | 0xFF20 9000 | In |
| HPS\_LED | GPIO53 | GPIO1[24] | I/O | 0xFF20 9000 | Out |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

HPS peripherals are mapped to HPS base address space 0xFC00 0000 with 64KB size.

Registers of GPIO0 controller are mapped to the base address 0xFF20 8000 - 0xFF20 8FFF (4KB size)

Registers of GPIO1 controller are mapped to the base address 0xFF20 9000 - 0xFF20 9FFF (4KB size)

Registers of GPIO2 controller are mapped to the base address 0xFF20 A000 - 0xFF20 8FFF (4KB size)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | http://www.altera.com/literature/hb/cyclone-v/cv\_5v4.pdf |  |  |
| GPIO0 | 0xFF20 8000 - 0xFF20 8FFF | 0xFF70 8000 |  |  |
| GPIO1 | 0xFF20 9000 - 0xFF20 9FFF | 0xFF70 9000 |  |  |
| GPIO2 | 0xFF20 A000 - 0xFF20 8FFF | 0xFF70 A000 |  |  |
| LWFPGASLAVES |  | 0xFF20 0000 |  |  |
|  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| gpio0 | 0xFF70 8000 | HPS\_GPIO0\_ADDRESS | HPS\_GPIO0\_OFFSET |  |  |
| gpio\_swporta\_dr | 0 | HPS\_GPIO0\_GPIO\_SWPORTA\_DR\_ADDRESS | GPIO\_GPIO\_SWPORTA\_DR\_OFFSET |  |  |
| gpio\_swporta\_ddr | 0x04 | HPS\_GPIO0\_GPIO\_SWPORTA\_DDR\_ADDRESS | GPIO\_GPIO\_SWPORTA\_DDR\_OFFSET |  |  |
| gpio\_inten | 0x30 | HPS\_GPIO0\_GPIO\_INTEN\_ADDRESS | GPIO\_GPIO\_INTEN\_OFFSET |  |  |
| gpio\_intmask | 0x34 | HPS\_GPIO0\_GPIO\_INTMASK\_ADDRESS | GPIO\_GPIO\_INTMASK\_OFFSET |  |  |
| gpio\_inttype\_level | 0x38 | HPS\_GPIO0\_GPIO\_INTTYPE\_LEVEL\_ADDRESS | GPIO\_GPIO\_INTTYPE\_LEVEL\_OFFSET |  |  |
| gpio\_int\_polarity | 0x3c | HPS\_GPIO0\_GPIO\_INT\_POLARITY\_ADDRESS | GPIO\_GPIO\_INT\_POLARITY\_OFFSET |  |  |
| gpio\_intstatus | 0x40 | HPS\_GPIO0\_GPIO\_INTSTATUS\_ADDRESS | GPIO\_GPIO\_INTSTATUS\_OFFSET |  |  |
| gpio\_raw\_intstatus | 0x44 | HPS\_GPIO0\_GPIO\_RAW\_INTSTATUS\_ADDRESS | GPIO\_GPIO\_RAW\_INTSTATUS\_OFFSET |  |  |
| gpio\_debounce | 0x48 | HPS\_GPIO0\_GPIO\_DEBOUNCE\_ADDRESS | GPIO\_GPIO\_DEBOUNCE\_OFFSET |  |  |
| gpio\_porta\_eoi | 0x4c | HPS\_GPIO0\_GPIO\_PORTA\_EOI\_ADDRESS | GPIO\_GPIO\_PORTA\_EOI\_OFFSET |  |  |
| gpio\_ext\_porta | 0x50 | HPS\_GPIO0\_GPIO\_EXT\_PORTA\_ADDRESS | GPIO\_GPIO\_EXT\_PORTA\_OFFSET |  |  |
| gpio\_ls\_sync | 0x60 | HPS\_GPIO0\_GPIO\_LS\_SYNC\_ADDRESS | GPIO\_GPIO\_LS\_SYNC\_OFFSET |  |  |
| gpio\_id\_code | 0x64 | HPS\_GPIO0\_GPIO\_ID\_CODE\_ADDRESS | GPIO\_GPIO\_ID\_CODE\_OFFSET |  |  |
| gpio\_ver\_id\_code | 0x6c | HPS\_GPIO0\_GPIO\_VER\_ID\_CODE\_ADDRESS | GPIO\_GPIO\_VER\_ID\_CODE\_OFFSET |  |  |
| gpio\_config\_reg2 | 0x70 | HPS\_GPIO0\_GPIO\_CONFIG\_REG2\_ADDRESS | GPIO\_GPIO\_CONFIG\_REG2\_OFFSET |  |  |
| gpio\_config\_reg1 | 0x74 | HPS\_GPIO0\_GPIO\_CONFIG\_REG1\_ADDRESS | GPIO\_GPIO\_CONFIG\_REG1\_OFFSET |  |  |
|  |  |  |  |  |  |

#### Library installation

C:\altera\13.1\embedded\ip\altera\hps\altera\_hps\hwlib

HERE

#### Reference files

|  |  |  |
| --- | --- | --- |
| hps.h |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

##### Titre5

###### Titre6

Titre7

Titre8

Titre9

References

* Altera, Cyclone V Devices documentation,  
  <http://www.altera.com/literature/lit-cyclone-v.jsp?ln=devices_fpga&l3=Low-Cost%20FPGAs-Cyclone%20V%20%28E,%20GX,%20GT,%20SE,%20SX,%20ST%29&l4=Documentation>
* Cyclone V Device Handbook Volume 3: Hard Processor System Technical Reference Manual  
  <http://www.altera.com/literature/hb/cyclone-v/cv_5v4.pdf>
* Cyclone V Hard Processor System User Guide   
  <http://www.altera.com/literature/hb/cyclone-v/cv_5v4_08.pdf>
* Cyclone V, Device Datasheet   
  <http://www.altera.com/literature/hb/cyclone-v/cv_51002.pdf>
* Cylone V HPS addresses   
  <http://www.altera.com/literature/hb/cyclone-v/hps.html>
* Cyclone V Device Handbook Volume 1: Device Interfaces and Integration  
  <http://www.altera.com/literature/hb/cyclone-v/cyclone5_handbook.pdf>
* Cyclone V, Device Overview  
  <http://www.altera.com/literature/hb/cyclone-v/cv_51001.pdf>
* SoCAL documentation (html), The Altera SoC Abstraction Layer (SoCAL) API Reference Manual  
  <file:///C:/altera/13.1/embedded/ip/altera/hps/altera_hps/doc/socal/html/index.html>
* Altera HWLIB, The Altera HW Manager API Reference Manual  
  <file:///C:/altera/13.1/embedded/ip/altera/hps/altera_hps/doc/hwmgr/html/index.html>
* Cyclone V, A Bare-Metal Debugging using ARM DS-5 Altera Edition  
  <http://www.youtube.com/watch?v=CJ0EHJ9oQ7Y>
* Linux Kernel Debug using ARM DS-5 Altera Edition  
  <http://www.youtube.com/watch?v=QcA39O6ofGw>
* FPGA-adaptive debug on the Altera SoC using ARM DS-5  
  <http://www.youtube.com/watch?v=2NBcUv2TxbI>
* A Look Inside: SoC FPGAs Introduction (Part 1 of 5)  
  <http://www.youtube.com/watch?v=RVM-ESUMOMU> (Part 1 of 5)  
  <http://www.youtube.com/watch?v=Ssxf8ggmQk4> (Part 2 of 5)  
  <http://www.youtube.com/watch?v=cWIaqt2RU84> (Part 3 of 5)  
  <http://www.youtube.com/watch?v=gUE669XKhUY> (Part 4 of 5)  
  <http://www.youtube.com/watch?v=NxZznvf5EKc> (Part 5 of 5)
* DS-5 Altera Edition: Bare-metal Debug and Trace  
  <http://www.youtube.com/watch?v=u_xKybPhcHI>
* OpenCL on FPGAs Accelerating Performance and Design Productivity — Altera   
  <http://www.youtube.com/watch?v=M6vpq6s1h_A>