DE1-SoC FPGA Board: Peripheral Buttons

By: Adam Narten; Winter 2018

Tools

Quartus Prime 17.0

Eclipse for DS-5 v.5.25.0

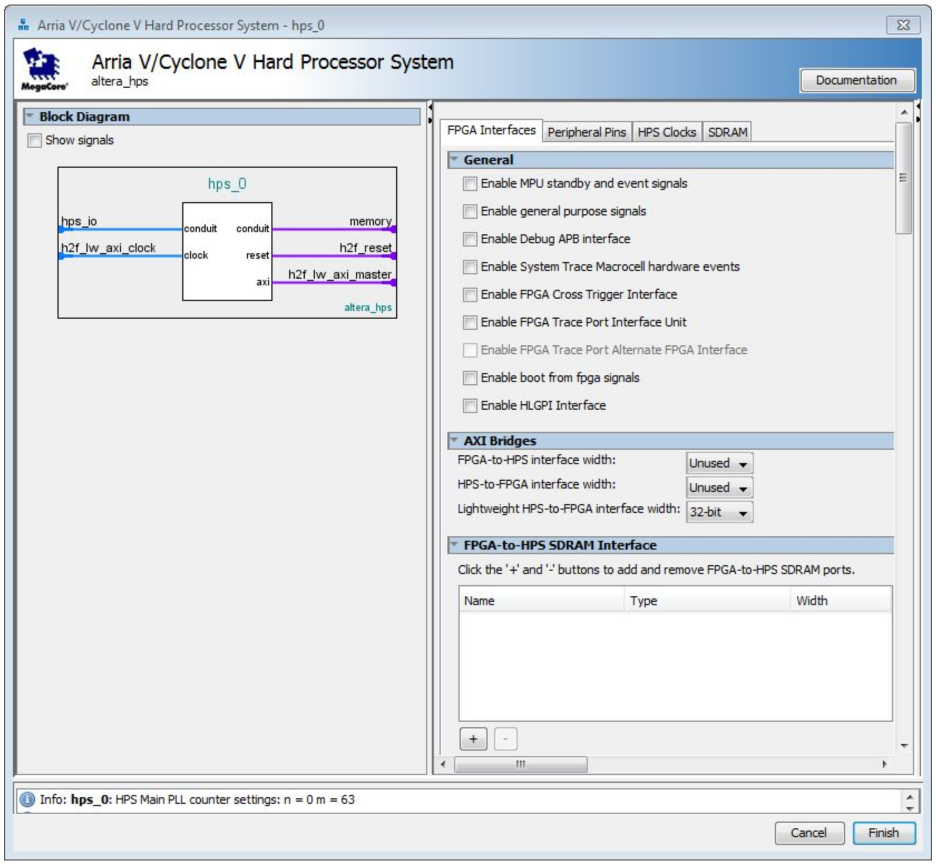
Introduction

These notes will show how to setup and configure the on board buttons, as well as provide a sample task that shows button reactions on the red LEDS.

Hardware Procedure

The following images show the minimum required components in the Qsys system as well as the required settings.

1. Clock
   - Use the default clock source created by the New Project Wizard
2. HPS System
   - Note this is the same as the tutorial setup done in the ECE 492 Labs.



General

- [ ] Enable MPU standby and event signals
- [ ] Enable general purpose signals
- [ ] Enable Debug APB interface
- [ ] Enable System Trace Macrocell hardware events
- [ ] Enable FPGA Cross Trigger Interface
- [ ] Enable FPGA Trace Port Interface Unit
- [ ] Enable FPGA Trace Port Alternate FPGA Interface
- [ ] Enable boot from fpga signals
- [ ] Enable HLGPI Interface

AXI Bridges

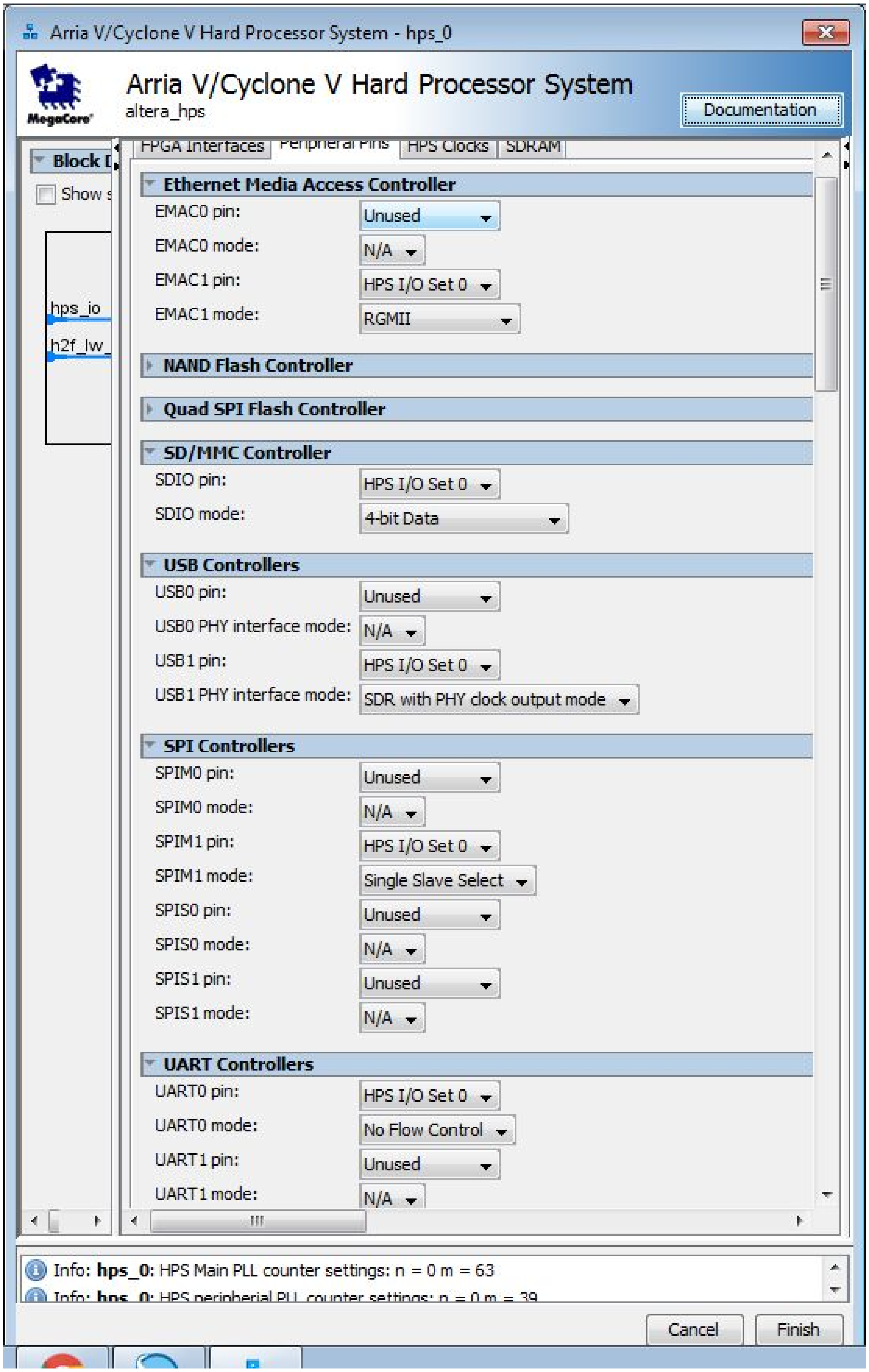
FPGA-to-HPS interface width:	Unused
HPS-to-FPGA interface width:	Unused
Lightweight HPS-to-FPGA interface width:	32-bit

FPGA-to-HPS SDRAM Interface

Click the '+' and '-' buttons to add and remove FPGA-to-HPS SDRAM ports.

Name	Type	Width

Info: **hps_0:** HPS Main PLL counter settings: n = 0 m = 63



Ethernet Media Access Controller

EMAC0 pin: Unused
EMAC0 mode: N/A
EMAC1 pin: HPS I/O Set 0
EMAC1 mode: RGMII

NAND Flash Controller

Quad SPI Flash Controller

SD/MMC Controller

SDIO pin: HPS I/O Set 0
SDIO mode: 4-bit Data

USB Controllers

USB0 pin: Unused
USB0 PHY interface mode: N/A
USB1 pin: HPS I/O Set 0
USB1 PHY interface mode: SDR with PHY clock output mode

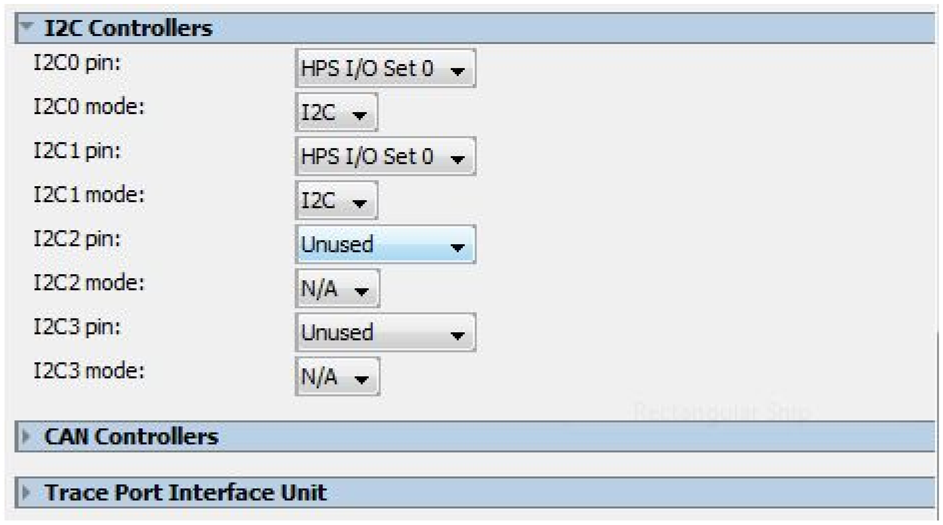
SPI Controllers

SPIM0 pin: Unused
SPIM0 mode: N/A
SPIM1 pin: HPS I/O Set 0
SPIM1 mode: Single Slave Select
SPIS0 pin: Unused
SPIS0 mode: N/A
SPIS1 pin: Unused
SPIS1 mode: N/A

UART Controllers

UART0 pin: HPS I/O Set 0
UART0 mode: No Flow Control
UART1 pin: Unused
UART1 mode: N/A

```
Info: hps_0: HPS Main PLL counter settings: n = 0 m = 63
Info: hps_0: HPS peripheral PLL counter settings: n = 0 m = 39
```

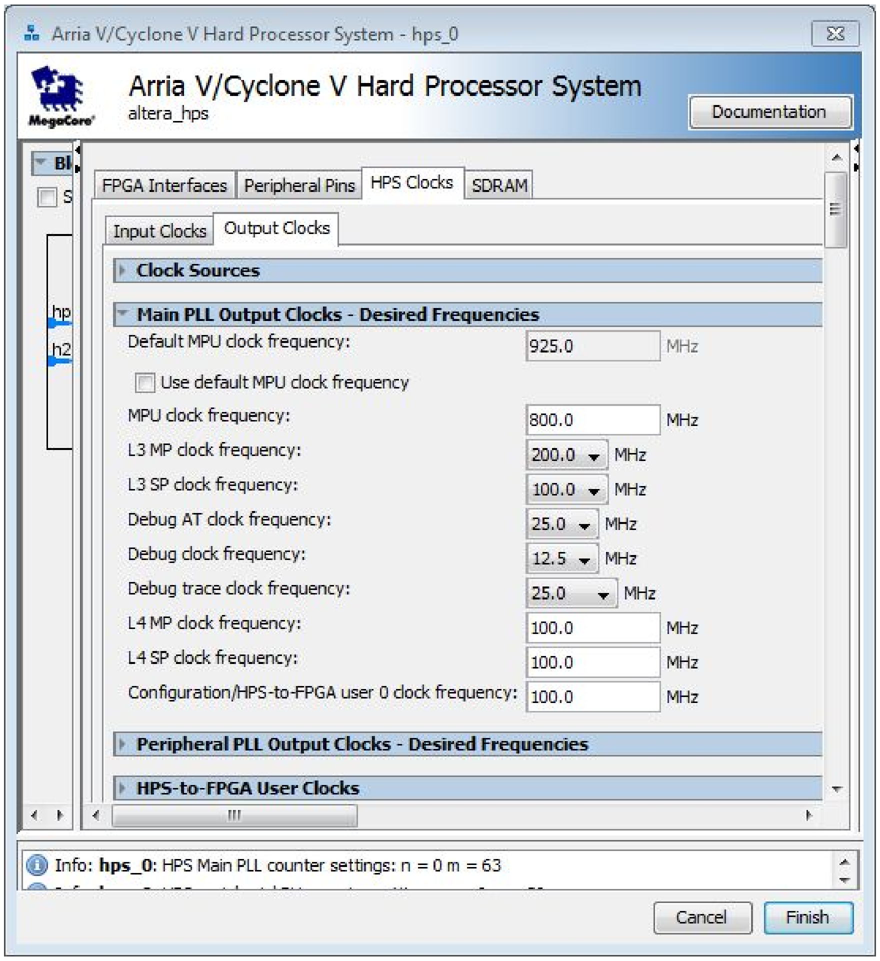


I2C Controllers

I2C0 pin:	HPS I/O Set 0
I2C0 mode:	I2C
I2C1 pin:	HPS I/O Set 0
I2C1 mode:	I2C
I2C2 pin:	Unused
I2C2 mode:	N/A
I2C3 pin:	Unused
I2C3 mode:	N/A

In the Peripheral Mux Table click on the GPIO09, GPIO35, GPIO40, GPIO48, GPIO53, GPIO54 and GPIO61 pins to appropriately map them manually. .

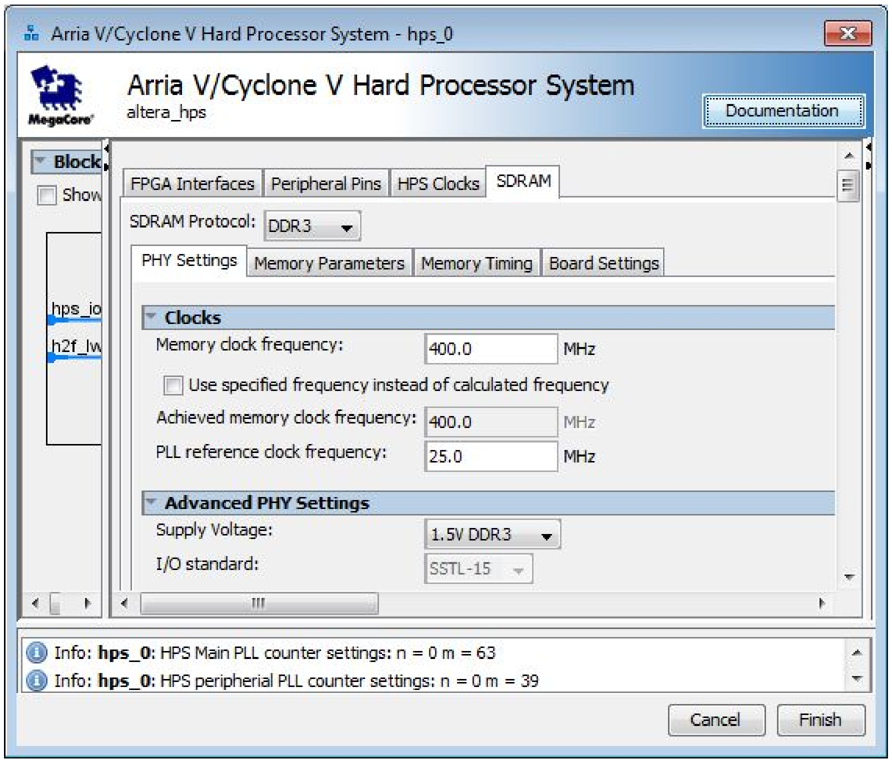
Under the HPS Clocks tab leave the Input Clocks tab as the default.
The output clocks are as follows:



Main PLL Output Clocks - Desired Frequencies

Default MPU clock frequency:	925.0	MHz
Use default MPU clock frequency	☐	
MPU clock frequency:	800.0	MHz
L3 MP clock frequency:	200.0	MHz
L3 SP clock frequency:	100.0	MHz
Debug AT clock frequency:	25.0	MHz
Debug clock frequency:	12.5	MHz
Debug trace clock frequency:	25.0	MHz
L4 MP clock frequency:	100.0	MHz
L4 SP clock frequency:	100.0	MHz
Configuration/HPS-to-FPGA user 0 clock frequency:	100.0	MHz

Info: **hps_0:** HPS Main PLL counter settings: n = 0 m = 63



SDRAM Protocol: DDR3

Clocks

Memory clock frequency: 400.0 MHz

☐ Use specified frequency instead of calculated frequency

Achieved memory clock frequency: 400.0 MHz

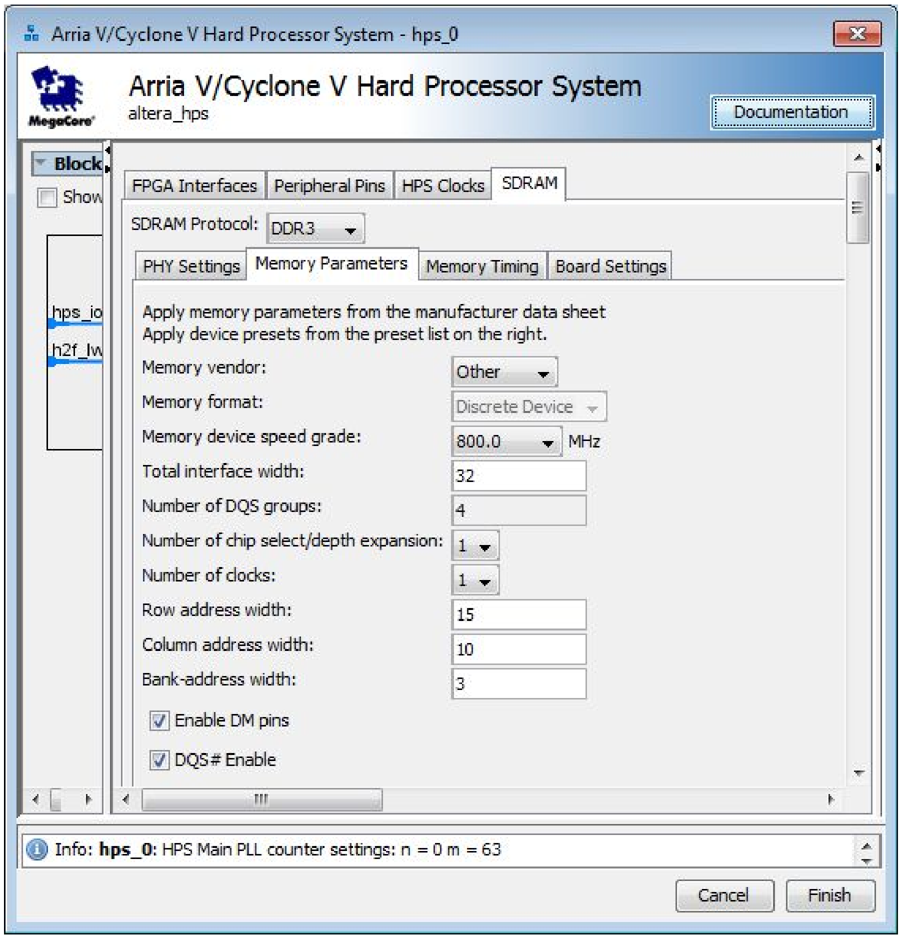
PLL reference clock frequency: 25.0 MHz

Advanced PHY Settings

Supply Voltage: 1.5V DDR3

I/O standard: SSTL-15

Info: **hps_0:** HPS Main PLL counter settings: n = 0 m = 63
Info: **hps_0:** HPS peripherial PLL counter settings: n = 0 m = 39



SDRAM Protocol: DDR3

Apply memory parameters from the manufacturer data sheet
Apply device presets from the preset list on the right.

Memory vendor: Other

Memory format: Discrete Device

Memory device speed grade: 800.0 MHz

Total interface width: 32

Number of DQS groups: 4

Number of chip select/depth expansion: 1

Number of clocks: 1

Row address width: 15

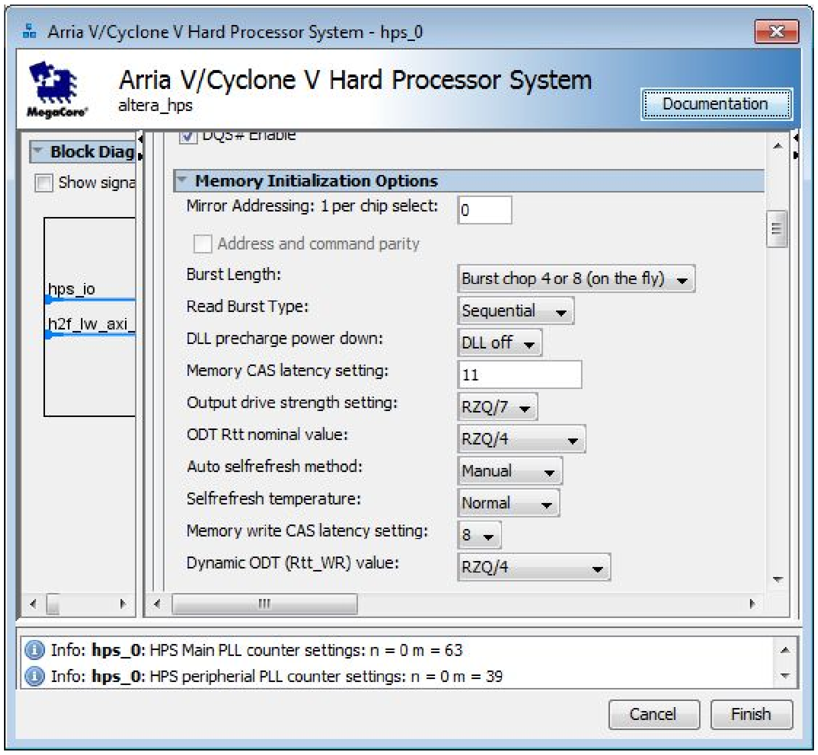
Column address width: 10

Bank-address width: 3

☑ Enable DM pins

☑ DQS# Enable

Info: **hps_0:** HPS Main PLL counter settings: n = 0 m = 63


Arria V/Cyclone V Hard Processor System - hps_0
Arria V/Cyclone V Hard Processor System
altera_hps
Documentation
Block Diag
Show signa
hps_io
h2f_lw_axi_
Cancel
Finish

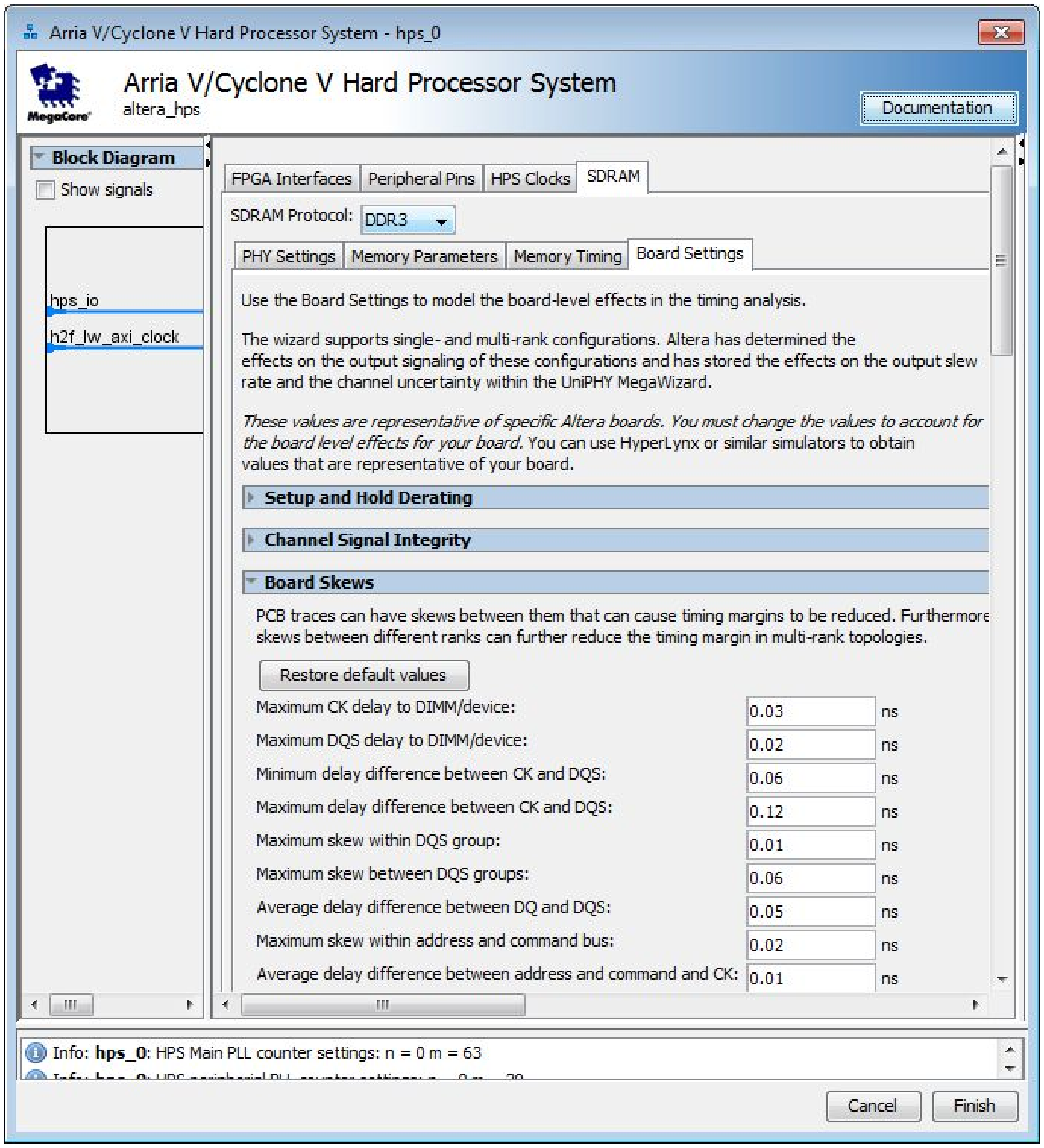
☑ DQS# Enable

Memory Initialization Options

Mirror Addressing: 1 per chip select:	0
☐ Address and command parity	
Burst Length:	Burst chop 4 or 8 (on the fly)
Read Burst Type:	Sequential
DLL precharge power down:	DLL off
Memory CAS latency setting:	11
Output drive strength setting:	RZQ/7
ODT Rtt nominal value:	RZQ/4
Auto selfrefresh method:	Manual
Selfrefresh temperature:	Normal
Memory write CAS latency setting:	8
Dynamic ODT (Rtt_WR) value:	RZQ/4

```
Info: hps_0: HPS Main PLL counter settings: n = 0 m = 63
Info: hps_0: HPS peripherial PLL counter settings: n = 0 m = 39
```

Under Board Settings -> Setup and Hold Derating and Channel Signal Integrity, leave Altera's default settings checked.
The remaining configuration is as follows:



SDRAM Protocol: DDR3

Use the Board Settings to model the board-level effects in the timing analysis.

The wizard supports single- and multi-rank configurations. Altera has determined the effects on the output signaling of these configurations and has stored the effects on the output slew rate and the channel uncertainty within the UniPHY MegaWizard.

These values are representative of specific Altera boards. You must change the values to account for the board level effects for your board. You can use HyperLynx or similar simulators to obtain values that are representative of your board.

PCB traces can have skews between them that can cause timing margins to be reduced. Furthermore skews between different ranks can further reduce the timing margin in multi-rank topologies.

Maximum CK delay to DIMM/device:	0.03	ns
Maximum DQS delay to DIMM/device:	0.02	ns
Minimum delay difference between CK and DQS:	0.06	ns
Maximum delay difference between CK and DQS:	0.12	ns
Maximum skew within DQS group:	0.01	ns
Maximum skew between DQS groups:	0.06	ns
Average delay difference between DQ and DQS:	0.05	ns
Maximum skew within address and command bus:	0.02	ns
Average delay difference between address and command and CK:	0.01	ns

Info: **hps_0**: HPS Main PLL counter settings: n = 0 m = 63

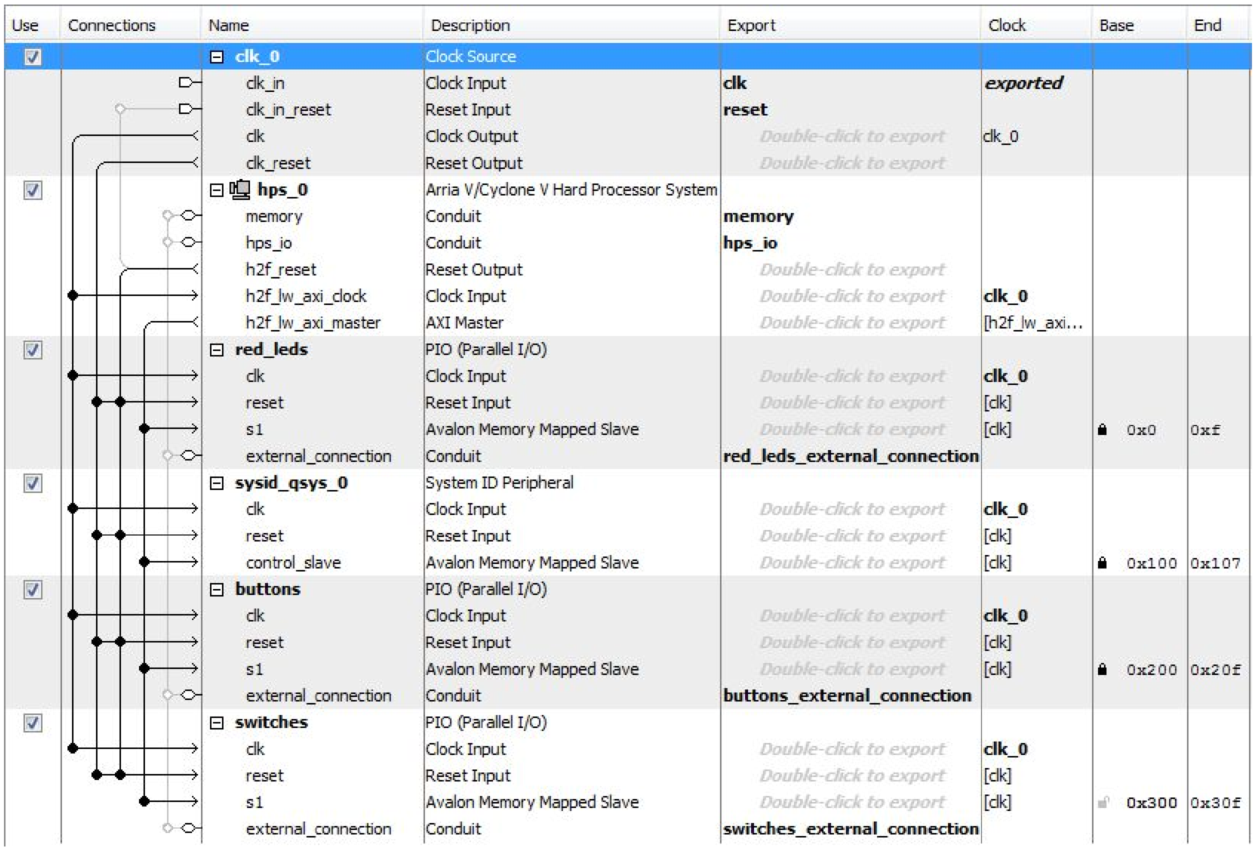
3. PIO (Parallel I/O) for red LEDs
   Set the width to 10 and the direction to output. Leave all other settings as default.

4. PIO (Parallel I/O) for buttons
   Set the width to 4 and the direction to input. Leave all other settings as default.

5. PIO (Parallel I/O) for switches
   Set the width to 10 and the direction to input. Leave all other settings as default.

6. System ID Peripheral
   Use the default configuration

Once these components are added, connect them according to the following image:



Use	Connections	Name	Description	Export	Clock	Base	End
✓		⊟ **clk_0**	Clock Source				
		clk_in	Clock Input	**clk**	*exported*		
		clk_in_reset	Reset Input	**reset**			
		clk	Clock Output	Double-click to export	clk_0		
		clk_reset	Reset Output	Double-click to export			
✓		⊟ **hps_0**	Arria V/Cyclone V Hard Processor System				
		memory	Conduit	**memory**			
		hps_io	Conduit	**hps_io**			
		h2f_reset	Reset Output	Double-click to export			
		h2f_lw_axi_clock	Clock Input	Double-click to export	**clk_0**		
		h2f_lw_axi_master	AXI Master	Double-click to export	[h2f_lw_axi...		
✓		⊟ **red_leds**	PIO (Parallel I/O)				
		clk	Clock Input	Double-click to export	**clk_0**		
		reset	Reset Input	Double-click to export	[clk]		
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	🔒 0x0	0xf
		external_connection	Conduit	**red_leds_external_connection**			
✓		⊟ **sysid_qsys_0**	System ID Peripheral				
		clk	Clock Input	Double-click to export	**clk_0**		
		reset	Reset Input	Double-click to export	[clk]		
		control_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	🔒 0x100	0x107
✓		⊟ **buttons**	PIO (Parallel I/O)				
		clk	Clock Input	Double-click to export	**clk_0**		
		reset	Reset Input	Double-click to export	[clk]		
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	🔒 0x200	0x20f
		external_connection	Conduit	**buttons_external_connection**			
✓		⊟ **switches**	PIO (Parallel I/O)				
		clk	Clock Input	Double-click to export	**clk_0**		
		reset	Reset Input	Double-click to export	[clk]		
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x300	0x30f
		external_connection	Conduit	**switches_external_connection**			

Note, the base addresses for the red leds and buttons are set at 0x000 and 0x200, respectively. You may change these to suit your design as long as they are updated in the button.h file (discussed later). The system ID peripheral and switches base addresses are 0x100 and 0x300; these may also be changed as needed.

Next, ensure your toplevel VHDL file in Quartus is connected as follows. A template is provided in buttons.vhd for reference.

A. Declare these pins in the entity (in addition to all of the hps signals):

```
-- FPGA side pins
        LEDR                                    : out std_logic_vector(9 downto 0);
-- Button / Key Pins
        KEY_N                                   : in std_logic_vector(3 downto 0);
  -- Switches
        SW                                      : in std_logic_vector(9 downto 0)
```

B. Add these ports in the architecture's component declaration

```
buttons_external_connection_export  : in    std_logic_vector(3 downto 0) := (others => 'X'); -- export
red_leds_external_connection_export : out   std_logic_vector(9 downto 0);                    -- export
switches_external_connection_export : in    std_logic_vector(9 downto 0) := (others => 'X') -- export
```

C. Port map the pins to the ports in the instantiation:

```
reset_reset_n                         => SW(9),
buttons_external_connection_export  => KEY_N,
red_leds_external_connection_export => LEDR,
switches_external_connection_export => SW
```

Note, the reset signal is now 'connected' to switch 9. This means that in order for the board to function properly, the reset signal must be asserted high upon flashing the board by physically turning the switch "on". With this implementation, switch 9 cannot be used for any other purpose and must be left in the "on" position to function properly; turning it "off" results in a reset of the board which requires reflashing.

Ensure that the .qip and .tcl scripts have also been added to the project before compiling. Also be sure to run the appropriate .tcl scripts beforehand. Then you may compile the design.

A zipped archive of a template project is included with this report for your reference.

Software Procedure

To ensure the software can properly communicate with the hardware, specify the base addresses in the button.h file, specifically for the red LEDs and buttons.

Since the buttons are active low, 0xf is constantly being written to the LEDs turning them on. Thus once a button is pressed, the value changes and can be handled accordingly.
To handle the button presses, a driver file button.c (button.h) has been provided to give an example. It is recommended that calls to key_pressed() are done in a loop to ensure quick response time. Alternatively, interrupts could be implemented by the end user to provide more reliable response time. An example is shown in the attached app.c file. The end user can add any additional functionality to handle button presses in the appropriately commented places in the app.c task. Also note, certain time delays have been added in button.c to allow for button debounce time. These times can be modified to satisfy user preference.

References

[1] University of Toronto. *Digital Embedded Systems Lab: Push Buttons.* [Online]. Available:http://www-ug.eecg.utoronto.ca/desl/nios_devices_SoC/dev_pushbuttons.html

Attached Files

app.c: Provides an example main.c and AppTask to detect button presses.
button.c: Provides button driver functions for reading button behavior.
button.h: Header file for the button driver functions.
buttons.qar: Archived Quartus/Qsys project example to implement peripheral buttons.
pin_assignment_DE1_SoC.tcl: Pin Assignments tcl script provided by Terasic