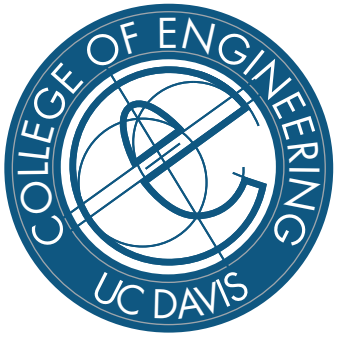


Measurement Board Control FPGA ERS

Version: 1.05

Authors: Jeremy W. Webb

Email: jwwebb@ece.ucdavis.edu

Contents

List of Figures

List of Tables

Table 1: Revision History

Revision	Date	Description	Initials
1.00	06/28/2008	First Draft	JWW
1.01	11/30/2008	Added initialization routine notes, added Data Path FPGA Configuration description, added Data Path FPGA Control description, updated Configuration Block Diagrams, and updated ICAP section.	JWW
1.02	03/15/2009	Added "Debug Commands" chapter.	JWW
1.03	04/18/2009	Updated EDK Peripherals chapter.	JWW
1.04	04/28/2009	Updated Data Path Control and Clock Generation EDK Peripherals.	JWW
1.05	05/26/2009	Updated Clock Generation EDK Peripherals.	JWW

1 Overview

The Control FPGA located on the Measurement Board is used to perform the following functions:

1. Receive and Decode High-Level Messages from the User Interface Board.
2. Initialize the Analog Devices AD9516 Clock PLL.
3. Configure the two AsAP Version 2 DSP Processors.
4. Monitor the temperature status of the board.
5. Control the two instrument fans.
6. Setup the Texas Instruments DAC5682Z 1 GS/s DAC.
7. Configure the Data Path FPGA.
8. Control the Data Path FPGA.
9. Display FFT results on an external display.
10. 10MHz Control.

The control functions listed above are implemented in GNU C code on a Xilinx MicroBlaze 32-bit Soft-Core Processor. The majority of the functions will operate autonomously.

2 Reference Documents

2.1 Schematics

The Measurement Board schematics are located at the following url:

http://www.ece.ucdavis.edu/vcl/vclpeople/jwwebb/measbd/docs/meas_main_board_bw.pdf

2.2 Datasheets and User Guides

2.2.1 Xilinx Spartan-3A

Manufacturer: Xilinx, Inc.
Manufactuers Part Number: XC3S1400A-4FGG484C

1. **Data Sheet**
   http://www.xilinx.com/support/documentation/data_sheets/ds529.pdf
2. **User Guide**
   http://www.xilinx.com/support/documentation/user_guides/ug331.pdf
3. **Configuration Guide**
   http://www.xilinx.com/support/documentation/user_guides/ug332.pdf

2.2.2 Texas Instruments High-Speed 16-bit 1GS/s D/A Converter (DAC)

1. **DAC5682ZIRGCT Data Sheet**
   http://focus.ti.com/lit/ds/symlink/dac5682z.pdf

2.2.3 Texas Instruments Temperature Sensors and Fan Controllers

1. **AMC6821 Data Sheet**
   http://focus.ti.com/lit/ds/symlink/amc6821.pdf
2. **TMP125 Data Sheet**
   http://focus.ti.com/lit/ds/symlink/tmp125.pdf

2.2.4 Analog Devices Clock PLL IC

1. **AD9516-3BCPZ Data Sheet**
   http://www.analog.com/UploadedFiles/Data_Sheets/AD9516_3.pdf

2.2.5 UMC 1GHz VCO

1. **UMX-244-B14 Data Sheet**
   http://www.vco1.com/SCDs/scd244-a.pdf

2.2.6 Vectron VTC2 TCXO

1. **Vectron International VTC2 Series TCXO**
   http://vectron.com/products/tcxo/VTC2.pdf

2.2.7 Silicon Laboratories CP2102

1. **CP2102 Product Brief**
   http://www.silabs.com/public/documents/marcom_doc/pbrief/Microcontrollers/Interface/en/CP2102_Brief.pdf
2. **CP2102 Data Short**
   http://www.silabs.com/public/documents/tpub_doc/dshort/Microcontrollers/Interface/en/CP2102_short.pdf
3. **CP2102 Data Sheet**
   http://www.silabs.com/public/documents/tpub_doc/dsheet/Microcontrollers/Interface/en/cp2102.pdf
4. **CP2102 Evaluation Kit User Guide**
   http://www.silabs.com/public/documents/tpub_doc/evbdsheet/Microcontrollers/Interface/en/CP2102-EK.pdf
5. **CP2102 Virtual COM Port Driver**
   http://www.silabs.com/tgwWebApp/public/web_content/products/Microcontrollers/USB/en/mcu_vcp.htm

2.2.8 Future Technology Devices International Ltd. (a.k.a, FTDI Chip)

1. **FT245BL Data Sheet**
   http://www.ftdichip.com/Documents/DataSheets/DS_FT245BL.pdf
2. **FT245BM Designers Guide (DG245)**
   http://www.ftdichip.com/Documents/AppNotes/DG245_20.pdf
3. **FT245BM Bit-Bang Mode (AN232B-01)**
   http://www.ftdichip.com/Documents/AppNotes/AN232B-01_BitBang.pdf
4. **FT245BM Power Control and Pin States (AN232B-02)**
   http://www.ftdichip.com/Documents/AppNotes/AN232B-02_PinModes_20.pdf
5. **Optimizing D2XX Data Throughput (AN232B-03)**
   http://www.ftdichip.com/Documents/AppNotes/AN232B-03_D2XXDataThroughput.pdf
6. **Data Throughput, Latency & Handshaking (AN232B-04)**
   http://www.ftdichip.com/Documents/AppNotes/AN232B-04_DataLatencyFlow.pdf
7. **Debugging FT245BM Based Designs (AN232B-06)**
   http://www.ftdichip.com/Documents/AppNotes/AN232B-06_11.pdf
8. **Configuring FTDI's VCP Drivers to use Location IDs (AN232B-07)**
   http://www.ftdichip.com/Documents/AppNotes/AN232B-07_LocIDs.pdf
9. **Using the Modem Emulation Mode in FTDI's VCP Driver (AN232B-09)**
   http://www.ftdichip.com/Documents/AppNotes/AN232B-09_Modem_Emulation_Mode.pdf
10. **Advanced Driver Options (AN232B-10)**
    http://www.ftdichip.com/Documents/AppNotes/AN232B-09_Modem_Emulation_Mode.pdf
11. **Virtual COM Port Drivers**
    http://www.ftdichip.com/Drivers/VCP.htm
12. **D2XX Drivers**
    http://www.ftdichip.com/Drivers/D2XX.htm

3 Control FPGA Block Diagram

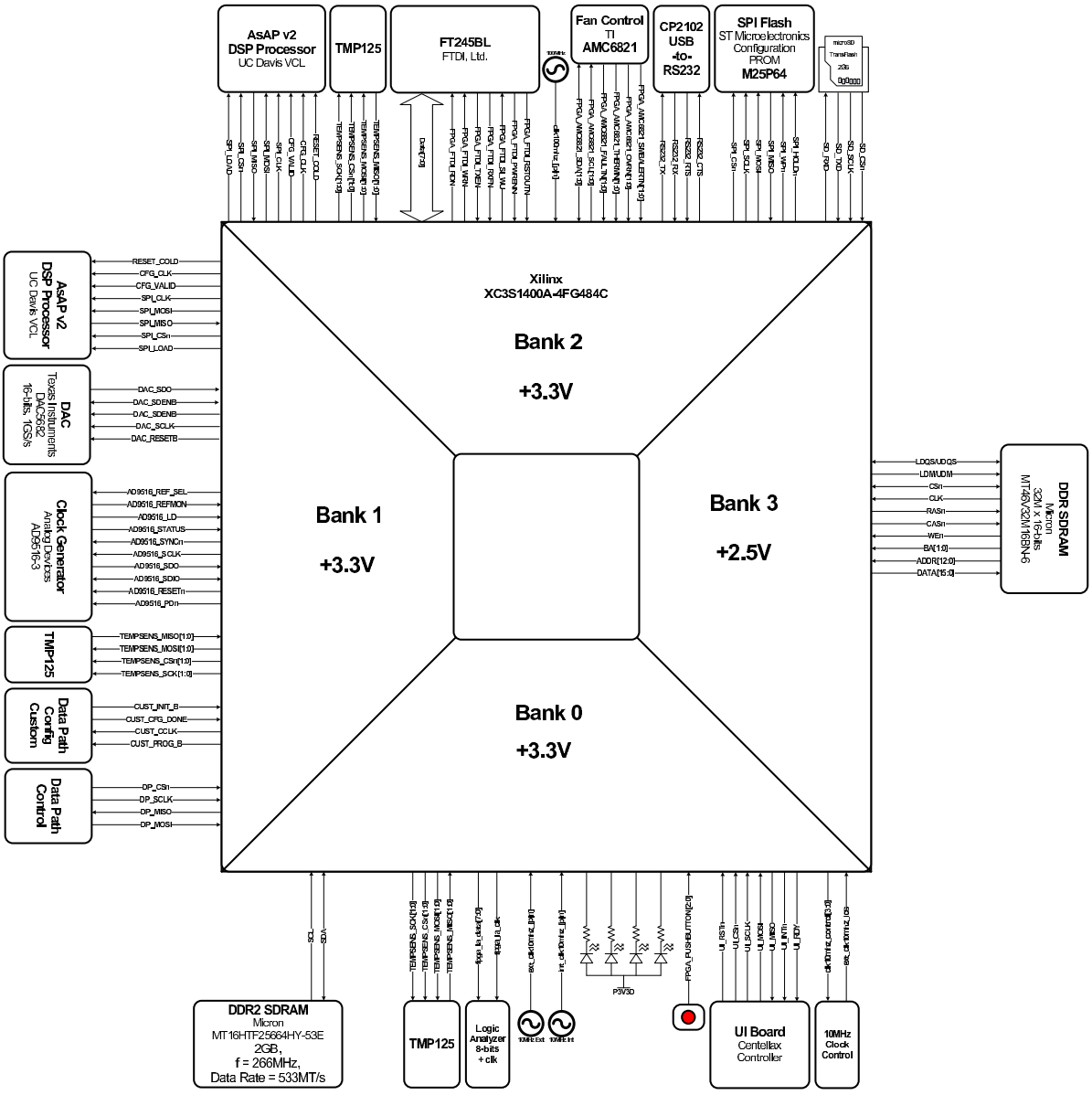


Figure 3.1: Control FPGA Block Diagram

4 EDK/MicroBlaze Peripherals

4.1 User Interface Board

The Measurement board can be controlled by the User Interface Board via a custom SPI bus.

The User Interface Board, shown in Figure 4.1, will provide 4 SPI ports for controlling Instrument boards. The SPI Ports will be used to control the Measurement board with an Xilinx FPGA. Table 4.1 shows a description of each signal function for the data signals on the SPI Port connector for both scenarios. Since the SPI interface consists of MOSI, MISO, SCLK, and CSn, it can still be used as a normal SPI port. The four SPI ports are meant to be used as point-to-point communications channels.

Signal Name	**Description**
MOSI	SPI Master-Out/Slave-In
MISO	SPI Master-In/Slave-Out
SCK	SPI Serial Clock
CSn	SPI Chip Select (Active Low) When CSn goes low the first 16-bits of MOSI will be the SPI command and the remaining n-bits will be data, where n = 8, 16, or 32.
INTn	Instrument Board Interrupt (Active Low)
RDY	Byte Ready Flag. 1 = ready, 0 = hold-off
RSTn	System Reset (Active Low)
PROG[0]	**FPGA**: *PROG_B*, The program pin, PROG_B, initiates the configuration process. The FPGA also automatically initiates configuration on power-up. The JTAG interface has a separate JTAG command to initiate configuration. The PROG_B pin also forces a master reset on the FPGA.
PROG[1]	**FPGA**: *INIT_B*, The INIT_B pins performs multiple functions. At the start of configuration, INIT_B goes Low indicating that the FPGA is clearing its internal configuration memory–a process called housecleaning. Later, when the FPGA is actively loading its configuration bitstream, INIT_B goes Low if the bitstream fails its CRC check. On Spartan-3A/3AN/3A DSP FPGAs, if so enabled in the FPGA application, the INIT_B pin also potentially signals a post-configuration CRC error.
PROG[2]	**FPGA**: *DONE*, The DONE pin, when High, indicates when the FPGA successfully completed loading its configuration data.

Table 4.1: User Interface Board SPI Port Signal Descriptions

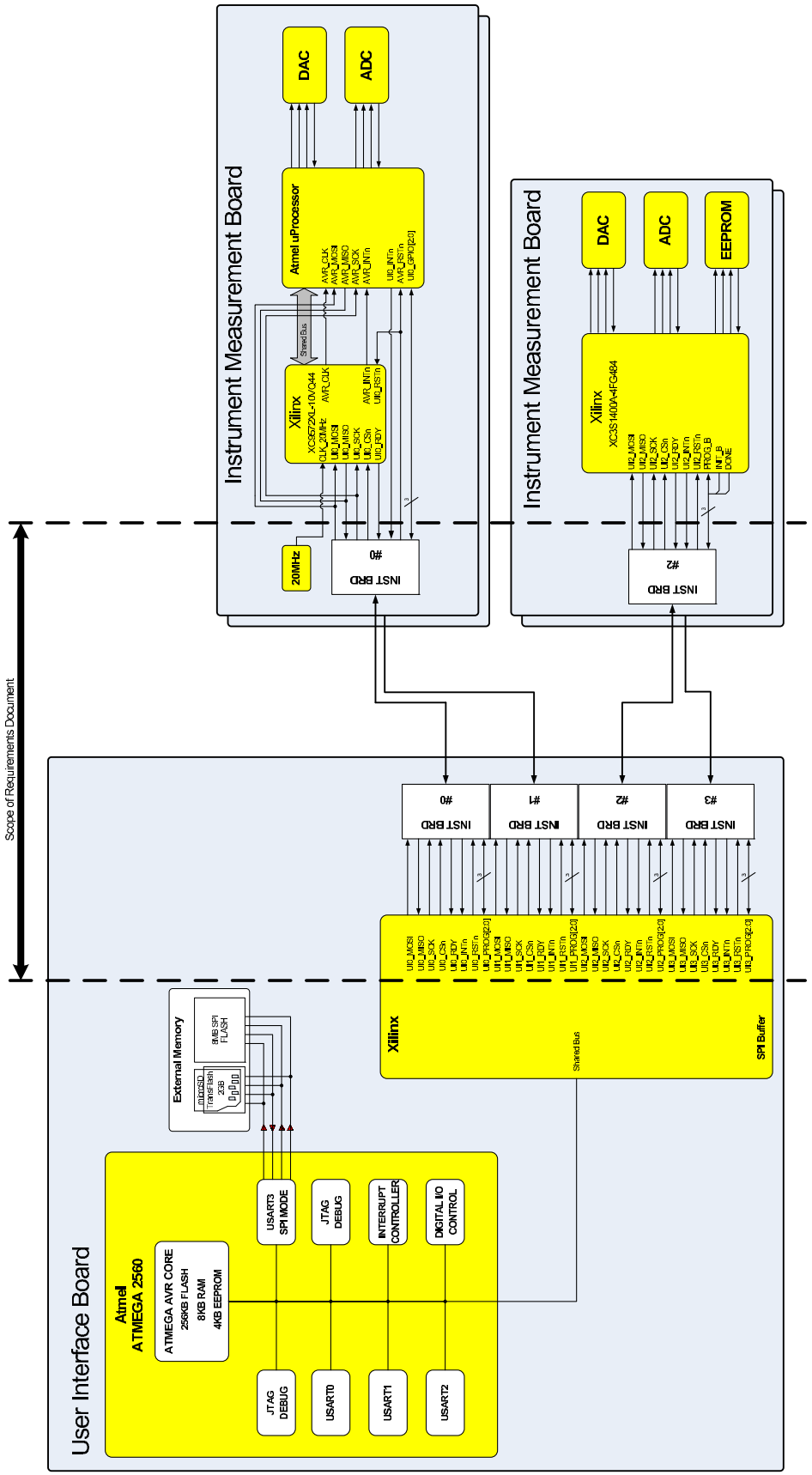


Figure 4.1: User Interface Board Communication Block Diagram

Table 4.1 shows the pinout of the SPI Port connector. A ground pin has been placed on three corners of the 14-pin 2mm Molex connector (Molex PN: 87832-1420), and an extra ground has been added to sandwich the SPI clock pin on the ribbon cable. The User Interface Board will use a 5.11 kΩ pull-up resistor on the **RDY** signal. Each Measurement Board will use a 5.11 kΩ pull-up resistor on both the **RDY** and **RSTn** signals.

Signal Name	Pin Number		Signal Name
	Odd	Even	
GND	1	2	GND
SCK	3	4	GND
MOSI	5	6	MISO
CSn	7	8	INTn
RDY	9	10	PROG[0]
PROG[1]	11	12	PROG[2]
RSTn	13	14	GND

Table 4.2: User Interface Board SPI Port Pinout

4.1.1 SPI_UI: User Interface Board SPI Controller

The User Interface board SPI interface connects to the Spartan-3A FPGA pins shown in Table 4.3.

Table 4.3: **SPI_UI EDK Peripheral I/O Descriptions**

ctx_uispi Pin Name	Pin Name	Dir	Pin	Description
MOSI	FPGA_UI_MOSI	I	B4	UI SPI Master-Out/Slave-In.
MISO	FPGA_UI_MISO	O	A6	UI SPI Master-In/Slave-Out.
SCK	FPGA_UI_SCK	I	D5	UI SPI Serial Clock.
SS	FPGA_UI_CSN	I	C6	UI SPI Chip Select (Active Low).

4.1.2 GPIO_UI_OUTS: User Interface Board GPIO Controller

The User Interface board non-SPI interface connects to the Spartan-3A FPGA pins shown in Table 4.4.

Table 4.4: **GPIO_UI_OUTS EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	FPGA_UI_INTN	O	B6	1	UI Interrupt (Active Low).
1	FPGA_UI_RDY	O	A5	1	UI Ready (Active Low).

4.1.3 GPIO_UI_INS: User Interface Board GPIO Controller

The User Interface board non-SPI interface connects to the Spartan-3A FPGA pins shown in Table 4.5.

Table 4.5: **GPIO_UI_INS EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	FPGA_UI_RSTN	I	C5	NA	UI Reset (Active Low).

4.2 Field Upgrade

The Measurement Board contains both a 64MB SPI Configuration Flash PROM and a 2GB microSD card for configuration file and calibration data storage. The bootloader application will always be contained in the block ram of each FPGA image, and will rarely need to change. The factory image will always reside at location 0x0 in the SPI Configuration Flash PROM to provide a reliable way of recovering from a configuration error.

When a Field Upgrade is initiated, the User Interface Board will transfer the new image to the Measurement Board where it will be stored in a new directory on the microSD card. The MicroBlaze 32-bit micro-controller will then perform a CRC/Checksum test to verify that the image is valid and move the upgrade image into the upgrade location in the SPI Configuration Flash PROM.

At system Power-Up or restart, the Spartan-3A Data Path FPGA will be configured from whichever image the ICAP peripheral is currently programmed to boot from in the SPI Configuration Flash PROM. The factory default is to boot from the factory image in the SPI Configuration Flash PROM. Once the Data Path FPGA is configured the bootloader will transfer the Control application data, as part of the Power-Up procedure, from the SPI Configuration Flash PROM to the DDR SDRAM memory.

4.2.1 hw_icap_registers_0: ICAP Software Register Peripheral

The *hw_icap_registers_v1_00_a* EDK Peripheral is used to control the Xilinx Spartan-3A ICAP Primitive (ICAP_SPARTAN3A). The ICAP_SPARTAN3A primitive works similar to the Slave Parallel (SelectMAP) configuration interface except it is available to the FPGA application using internal routing connections. Furthermore, the ICAP primitive has separate read and write data ports, as opposed to the bidirectional bus on the SelectMAP interface. ICAP allows the FPGA application to access configuration registers, readback configuration data, or to trigger a MultiBoot event after configuration successfully completes. The register map of the ICAP_SPARTAN3A primitive is shown in Table 4.6.

Table 4.6: **ICAP_SPARTAN3A** Register Map

Address	Name	Default Value
1	sync_H	0xAA
2	sync_L	0x99
3	t1w_gen1_H	0x32
4	t1w_gen1_L	0x61
5	addr[15:8]	internal_addr[15:8]
6	addr[7:0]	internal_addr[7:0]
7	t1w_gen2_H	0x32
8	t1w_gen2_L	0x81
9	addr[31:24]	internal_addr[31:24]
10	addr[23:16]	internal_addr[23:16]
11	t1w_mode_H	0x32
12	t1w_mode_L	0xA1
13	reserved	{1'b0, internal_use, internal_mode[2:0], internal_vsel[2:0]}
14	user_mode	0x00
15	t1w_cmd_H	0x30
16	t1w_cmd_L	0xA1
17	reboot_H	0x00
18	reboot_L	0x0E
19	noop_H	0x20
20	noop_L	0x00

The variables shown in the *Default Value* column of Table 4.6 are described in Table 4.7.

Table 4.7: ICAP Variable Descriptions

Variable Name	# of Bits	Description
internal_addr	32	MultiBoot Address: SPI PROM Boot Location
internal_mode	3	BOOTMODE: M[2:0] Configuration Mode Pins
internal_vsel	3	VS[2:0] SPI Configuration Mode Pins
internal_use	1	NEW_MODE 0 = Sample M[2:0] and VS[2:0] pins 1 = Use BOOTMODE bits

The *hw_icap_registers_v1_00_a* EDK Peripheral's register map is shown in Table 4.8.

Table 4.8: **hw_icap_registers_v1_00_a EDK Peripheral Register Map**

Address	Name	R/W	Default Value	Description
0	{24'b0,icap_wr_reg[24:31]}	R/W	0x00000000	ICAP Write Byte Register.
1	{24'b0,icap_rd_reg[24:31]}	RO	0x00000000	ICAP Read Byte Register.
2	{30'b0,icap_ctrl[30:31]}	R/W	0x00000003	ICAP Control Register. [0:29]: reserved; [30]: icap_rnw; ICAP Read/nWrite Signal 1 = Read Operation. 0 = Write Operation. [31]: icap_ce; ICAP Clock Enable (Active Low)
3	{31'b0,icap_clk_r[31]}	R/W	0x00000001	ICAP Clock Register. [0:30]: reserved; [31]: icap_clk; ICAP Clock Toggle high then low to clock a data byte into or out of the ICAP module.
4	{31'b0,icap_busy}	RO	0x00000000	ICAP Busy/Ready output Register.
				[0:30]: reserved; [31]: icap_busy; ICAP Busy/Ready output (Active High). busy status. Only used in read operations. Busy remains Low during writes. Ops can be started when this bit goes high.

4.2.1.1 ICAP: Get Boot Address

To read the boot address from the ICAP module perform the following:

1. **First get lower 16-bits**
2. Set the icap_ctrl[30:31] register bits (icap_rnw bit and icap_ce) to a 1; or 0x3.
3. Set the icap_ctrl[30] register bit (icap_rnw bit) to a 0.
4. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
5. Write 0x20 (noop_H) to the icap_wr_reg register.
6. Set the icap_clk register bit to a 0.
7. Set the icap_clk register bit to a 1.
8. Write 0x00 (noop_L) to the icap_wr_reg register.
9. Set the icap_clk register bit to a 0.
10. Set the icap_clk register bit to a 1.
11. Write 0xAA (sync_H) to the icap_wr_reg register.
12. Set the icap_clk register bit to a 0.
13. Set the icap_clk register bit to a 1.
14. Write 0x99 (sync_L) to the icap_wr_reg register.
15. Set the icap_clk register bit to a 0.
16. Set the icap_clk register bit to a 1.
17. Write 0x2A (t1r_gen2_H) to the icap_wr_reg register.
18. Set the icap_clk register bit to a 0.
19. Set the icap_clk register bit to a 1.
20. Write 0x81 (t1r_gen2_L) to the icap_wr_reg register.
21. Set the icap_clk register bit to a 0.
22. Set the icap_clk register bit to a 1.
23. Write 0x20 (noop_H) to the icap_wr_reg register.
24. Set the icap_clk register bit to a 0.
25. Set the icap_clk register bit to a 1.
26. Write 0x00 (noop_L) to the icap_wr_reg register.
27. Set the icap_clk register bit to a 0.
28. Set the icap_clk register bit to a 1.
29. Set the icap_ctrl[31] register bit (icap_ce bit) to a 1.
30. Set the icap_ctrl[30] register bit (icap_rnw bit) to a 1.
31. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
32. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.

33. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
34. Read the value in the icap_rd_reg, store the value in the lower 8-bits of the Address variable (i.e., addr[7:0]).
35. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
36. Read the value in the icap_rd_reg, store the value in the next 8-bits of the Address variable (i.e., addr[15:8]).
37. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
38. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
39. Set the icap_ctrl[31] register bit (icap_ce bit) to a 1.
40. **Now get upper 16-bits**
41. Set the icap_ctrl[30:31] register bits (icap_rnw bit and icap_ce) to a 1; or 0x3.
42. Set the icap_ctrl[30] register bit (icap_rnw bit) to a 0.
43. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
44. Write 0x20 (noop_H) to the icap_wr_reg register.
45. Set the icap_clk register bit to a 0.
46. Set the icap_clk register bit to a 1.
47. Write 0x00 (noop_L) to the icap_wr_reg register.
48. Set the icap_clk register bit to a 0.
49. Set the icap_clk register bit to a 1.
50. Write 0xAA (sync_H) to the icap_wr_reg register.
51. Set the icap_clk register bit to a 0.
52. Set the icap_clk register bit to a 1.
53. Write 0x99 (sync_L) to the icap_wr_reg register.
54. Set the icap_clk register bit to a 0.
55. Set the icap_clk register bit to a 1.
56. Write 0x2A (t1r_gen1_H) to the icap_wr_reg register.
57. Set the icap_clk register bit to a 0.
58. Set the icap_clk register bit to a 1.
59. Write 0x61 (t1r_gen1_L) to the icap_wr_reg register.
60. Set the icap_clk register bit to a 0.
61. Set the icap_clk register bit to a 1.
62. Write 0x20 (noop_H) to the icap_wr_reg register.
63. Set the icap_clk register bit to a 0.
64. Set the icap_clk register bit to a 1.
65. Write 0x00 (noop_L) to the icap_wr_reg register.
66. Set the icap_clk register bit to a 0.

67. Set the icap_clk register bit to a 1.
68. Set the icap_ctrl[31] register bit (icap_ce bit) to a 1.
69. Set the icap_ctrl[30] register bit (icap_rnw bit) to a 1.
70. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
71. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
72. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
73. Read the value in the icap_rd_reg, store the value in the next 8-bits of the Address variable (i.e., addr[23:16]).
74. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
75. Read the value in the icap_rd_reg, store the value in the next 8-bits of the Address variable (i.e., addr[31:24]).
76. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
77. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
78. Set the icap_ctrl[31] register bit (icap_ce bit) to a 1.
79. Now we have the boot address stored in the Address variable.

4.2.1.2 ICAP: Reboot

To perform an FPGA reboot using the ICAP module perform the following:

1. Set the icap_ctrl[30:31] register bits (icap_rnw bit and icap_ce) to a 1; or 0x3.
2. Set the icap_ctrl[30] register bit (icap_rnw bit) to a 0.
3. Set the icap_ctrl[31] register bit (icap_ce bit) to a 0.
4. Write 0x20 (noop_H) to the icap_wr_reg register.
5. Set the icap_clk register bit to a 0.
6. Set the icap_clk register bit to a 1.
7. Write 0x00 (noop_L) to the icap_wr_reg register.
8. Set the icap_clk register bit to a 0.
9. Set the icap_clk register bit to a 1.
10. Write 0xAA (sync_H) to the icap_wr_reg register.
11. Set the icap_clk register bit to a 0.
12. Set the icap_clk register bit to a 1.
13. Write 0x99 (sync_L) to the icap_wr_reg register.
14. Set the icap_clk register bit to a 0.
15. Set the icap_clk register bit to a 1.
16. Write 0x32 (t1w_gen1_H) to the icap_wr_reg register.
17. Set the icap_clk register bit to a 0.
18. Set the icap_clk register bit to a 1.
19. Write 0x61 (t1w_gen1_L) to the icap_wr_reg register.
20. Set the icap_clk register bit to a 0.
21. Set the icap_clk register bit to a 1.
22. Write addr[15:8] to the icap_wr_reg register.
23. Set the icap_clk register bit to a 0.
24. Set the icap_clk register bit to a 1.
25. Write addr[7:0] to the icap_wr_reg register.
26. Set the icap_clk register bit to a 0.
27. Set the icap_clk register bit to a 1.
28. Write 0x32 (t1w_gen2_H) to the icap_wr_reg register.
29. Set the icap_clk register bit to a 0.
30. Set the icap_clk register bit to a 1.
31. Write 0x81 (t1w_gen2_L) to the icap_wr_reg register.
32. Set the icap_clk register bit to a 0.

33. Set the icap_clk register bit to a 1.
34. Write 0x03 (C7:0) to the icap_wr_reg register.
35. Set the icap_clk register bit to a 0.
36. Set the icap_clk register bit to a 1.
37. Write addr[23:16] to the icap_wr_reg register.
38. Set the icap_clk register bit to a 0.
39. Set the icap_clk register bit to a 1.
40. Write 0x32 (t1w_mode_H) to the icap_wr_reg register.
41. Set the icap_clk register bit to a 0.
42. Set the icap_clk register bit to a 1.
43. Write 0xA1 (t1w_mode_L) to the icap_wr_reg register.
44. Set the icap_clk register bit to a 0.
45. Set the icap_clk register bit to a 1.
46. Write 0x00 to the icap_wr_reg register.
47. Set the icap_clk register bit to a 0.
48. Set the icap_clk register bit to a 1.
49. Write 0x4D (sample M[2:0] and VS[2:0] pins) to the icap_wr_reg register.
50. Set the icap_clk register bit to a 0.
51. Set the icap_clk register bit to a 1.
52. Write 0x30 (t1w_cmd_H) to the icap_wr_reg register.
53. Set the icap_clk register bit to a 0.
54. Set the icap_clk register bit to a 1.
55. Write 0xA1 (t1w_cmd_L) to the icap_wr_reg register.
56. Set the icap_clk register bit to a 0.
57. Set the icap_clk register bit to a 1.
58. Write 0x00 (reboot_H) to the icap_wr_reg register.
59. Set the icap_clk register bit to a 0.
60. Set the icap_clk register bit to a 1.
61. Write 0x0E (reboot_L) to the icap_wr_reg register.
62. Set the icap_clk register bit to a 0.
63. Set the icap_clk register bit to a 1.
64. Write 0x20 (noop_H) to the icap_wr_reg register.
65. Set the icap_clk register bit to a 0.
66. Set the icap_clk register bit to a 1.

67. Write 0x00 (noop_L) to the icap_wr_reg register.
68. Set the icap_clk register bit to a 0.
69. Set the icap_clk register bit to a 1.
70. Write 0x20 (noop_H) to the icap_wr_reg register.
71. Set the icap_clk register bit to a 0.
72. Set the icap_clk register bit to a 1.
73. Write 0x00 (noop_L) to the icap_wr_reg register.
74. Set the icap_clk register bit to a 0.
75. Set the icap_clk register bit to a 1.
76. Set the icap_ctrl[31] register bit (icap_rnw bit) to a 1.
77. Set the icap_ctrl[30] register bit (icap_ce bit) to a 1.

4.3 Data Path FPGA Configuration

The Control FPGA is responsible for configuring the Data Path FPGA at power-up. The Control FPGA will configure the Data Path FPGA using the Slave Serial method. The Slave Serial interface consists of the following 5 signals:

1. CCLK: Serial Configuration Clock
2. DIN: Serial Data In
3. INIT_B: Initialization Flag (Active Low)
4. PROG_B: Configuration Reset (Active Low)
5. DONE: Configuration Done Flag (Active High)

The Data Path FPGA Configuration process consists of sending 2,730,704 bytes to the Data Path Slave Serial interface 32-bits at a time using the *vcl_dp_fpga_config_v1_00_a* EDK Peripheral. First the PROG_B signal is toggled low then high to initiate the Data Path FPGA Configuration process. The MicroBlaze Data Path FPGA Configuration application will wait until the INIT_B signal transitions from low to high. Once INIT_B goes high, the MicroBlaze Data Path FPGA Configuration application will begin to send 32-bit words one at a time, while waiting for the txfer_done signal of the *vcl_dp_fpga_config_v1_00_a* EDK Peripheral to go high between each 32-bit word. Once the last 32-bit configuration data packet has been sent via the Slave Serial interface, the MicroBlaze Data Path FPGA Configuration application will wait for the Configuration DONE flag to go high. This will signify the completion of the Data Path FPGA configuration process. If the Configuration DONE flag does not go high, then the process should be executed again.

4.3.1 vcl_dp_fpga_config_0: Data Path FPGA Control Peripheral

The *vcl_dp_fpga_config_v1_00_a* EDK Peripheral is used to configure the Data Path FPGA, and appears as a set of 4 software accessible registers to the applications running on the MicroBlaze. The *vcl_dp_fpga_config_v1_00_a* EDK Peripheral's register map is shown in Table 4.9.

Table 4.9: **vcl_dp_fpga_config_v1_00_a EDK Peripheral Register Map**

Address	Name	R/W	Default Value	Description
0	cfg_data[0:31]	R/W	0x00000000	32-bit Configuration Write Data Register.
1	{29'b0,prog_reg[29:31]}	R/W	0x00000001	Configuration Control Register. [0:28]: reserved; [29]: send_clks: Send 16 Clocks (Active High). Toggle high-then-low to iniate transaction. [30]: send_data: Send 32-bits of Data (Active High). Toggle high-then-low to iniate transaction. [31]: PROGB: Initiate FPGA Program Sequence. Toggle low-then-high to iniate transaction.
2	slv_reg2[0:31]	R/W	0x00000000	Reserved Register.
3	{29'b0,txfer_done,DONE,INITB}	RO	0x00000000	Configuration Transfer Done Register. [0:28]: reserved; [29]: txfer_done: Serial Transfer Done (Active High). [30]: DONE: Configuration Done (Active High). [31]: INITB: Initialization Flag (Active Low).

To initiate a write tranaction using the *vcl_dp_fpga_config_v1_00_a* EDK Peripheral first set the Read/Write Select to write (rnw_i = 0), 16-bit address (saddr_i) and 32-bit data (sdata_i), then toggle the start_spi_i bit low-high-low to initiate the transfer.

To initiate a read tranaction using the *vcl_dp_fpga_config_v1_00_a* EDK Peripheral first set the Read/Write Select to read (rnw_i = 1) and 16-bit address (saddr_i), then toggle the start_spi_i bit low-high-low to initiate the transfer.

The *vcl_dp_fpga_config_v1_00_a* connects to the Spartan-3A FPGA pins shown in Table 4.10.

Table 4.10: **vcl_dp_fpga_config_v1_00_a EDK Peripheral I/O Descriptions**

xps_spi Pin Name	Pin Name	Dir	Pin	Description
i_dp_fpga_done	CUST_CFG_DONE	I	B22	Configuration Done (Active High).
i_dp_fpga_initb	CUST_INIT_B	I	B21	Initialization Flag (Active Low).
o_dp_fpga_cclk	CUST_CCLK	O	C21	Serial Configuration Clock.
o_dp_fpga_din	S3A_SPI_DATA_TO_V5	O	AA15	Serial Data.
o_dp_fpga_progb	CUST_PROG_B	O	C22	Configuration Reset (Active Low).

4.4 Data Path FPGA Control

The Data Path FPGA is responsible for generating PRBS and RAM patterns as well as injecting errors. It is implemented in SystemVerilog, and targeted on a Xilinx Virtex-5 SX50T (XC5VSX50T-1FFG1136C). The Data Path FPGA contains hardware registers that need to be initialized at power-up and changed based on user input. The hardware registers are accessed via an Instrument Local Bus (ILB), which is based on the Serial Peripheral Interface (SPI) standard. The SPI bus consists of the following 5 signals:

1. SCK: Serial Clock
2. MOSI: Master-Out/Slave-In
3. MISO: Master-In/Slave-Out
4. ADDR_CSn: Address Chip Select (Active Low)
5. DATA_CSn: Data Chip Select (Active Low)
6. RNW: Read/Write Select; 0 = Write, 1 = Read

The Data Path FPGA is a slave, so it needs a means of requesting service from the master. An additional signal is provided for this purpose:

1. SRQn: Service Request (Active Low)

A single ILB transaction consists of two segments: Address and Data. The address value consists of 16-bits, and the data value consists of 32-bits. First the desired address shifted out over MOSI as the ADDR_CSn chip select is driven low. Once the address has been shifted out to the slave, the ADDR_CSn chip select is driven high. Then two SCK pulses are generated before the DATA_CSn chip select is driven low while shifting the write data out over MOSI. Once the data has been shifted out to the slave, the DATA_CSn chip select is driven high with two SCK pulses following. The two SCK pulses that precede the DATA_CSn chip select allow for the addressed register contents to be shifted to the master over MISO. The two SCK pulses that follow the DATA_CSn chip select allow for the slave to perform any last minute tasks before the transaction is completed.

As the Data Path FPGA Control Interface is a non-standard SPI interface, a custom EDK peripheral called *vcl_dp_fpga_ctrl_v1_00_a* was designed to implement the SPI transactions.

4.4.1 vcl_dp_fpga_ctrl_0: Data Path FPGA Control Peripheral

The *vcl_dp_fpga_ctrl_v1_00_a* EDK Peripheral is used to control the Data Path FPGA, and appears as a set of 8 software accessible registers to the applications running on the MicroBlaze. The *vcl_dp_fpga_ctrl_v1_00_a* EDK Peripheral's register map is shown in Table 4.11.

Table 4.11: **vcl_dp_fpga_ctrl_v1_00_a EDK Peripheral Register Map**

Address	Name	R/W	Default Value	Description
0	{16'b0,saddr_i[16:31]}	R/W	0x00000000	16-bit SPI Address Register.
1	sdata_i[0:31]	R/W	0x00000000	32-bit SPI Write Data Register.
2	{30'b0,ilb_ctrl[30:31]}	R/W	0x00000002	ILB Control Register. [0:29]: reserved; [30]: rnw_i: Read/Write Select; 0 = Write, 1 = Read. [31]: start_spi_i: Start current SPI transaction. Toggle high-then-low to iniate transaction.
3	slv_reg3[0:31]	R/W	0x00000000	Reserved Register.
4	miso_read_o[0:31]	RO	0x00000000	SPI Read Data Register.
5	{31'b0,txfer_done}	RO	0x00000000	SPI Transfer Done Register. [0:30]: reserved; [31]: txfer_done: SPI Transfer Done (Active High).
6	slv_reg6[0:31]	RO	0xAAAAAAAA	Reserved Register.
7	slv_reg7[0:31]	RO	0x55555555	Reserved Register.

To initiate a write tranaction using the *vcl_dp_fpga_ctrl_v1_00_a* EDK Peripheral first set the Read/Write Select to write (rnw_i = 0), 16-bit address (saddr_i) and 32-bit data (sdata_i), then toggle the start_spi_i bit low-high-low to initiate the transfer.

To initiate a read tranaction using the *vcl_dp_fpga_ctrl_v1_00_a* EDK Peripheral first set the Read/Write Select to read (rnw_i = 1) and 16-bit address (saddr_i), then toggle the start_spi_i bit low-high-low to initiate the transfer.

The *vcl_dp_fpga_ctrl_v1_00_a* connects to the Spartan-3A FPGA pins shown in Table 4.12.

Table 4.12: **vcl_dp_fpga_ctrl_v1_00_a EDK Peripheral I/O Descriptions**

xps_spi Pin Name	Pin Name	Dir	Pin	Description
i_ilb_srqn	FPGA_DP_CTRL_INTN	I	G18	Service Request (Active Low).
i_ilb_miso	FPGA_DP_CTRL_MISO	I	F18	Master-In/Slave-Out Serial Data.
o_ilb_sck	FPGA_DP_CTRL_SCK	O	F20	Serial Clock.
o_ilb_mosi	FPGA_DP_CTRL_MOSI	O	F22	Master-Out/Slave-In Serial Data.
o_ilb_addr_csn	FPGA_DP_CTRL_CSN	O	F21	Address Chip Select (Active Low).
o_ilb_data_csn	FPGA_DP_CTRL_GPIO4	O	E22	Data Chip Select (Active Low).
o_ilb_rnw	FPGA_DP_CTRL_GPIO3	O	D22	Read/nWrite Select.
o_ilb_rstn	FPGA_DP_CTRL_RSTN	O	F19	Data Path FPGA Reset.

4.5 SPI Configuration Flash PROM Organization

The SPI Configuration Flash PROM will be organized as shown in Figure 7.4. Figure 7.4 shows the images abutted next to each other, but in the final implementation the images will be re-aligned to the beginning of a sector.

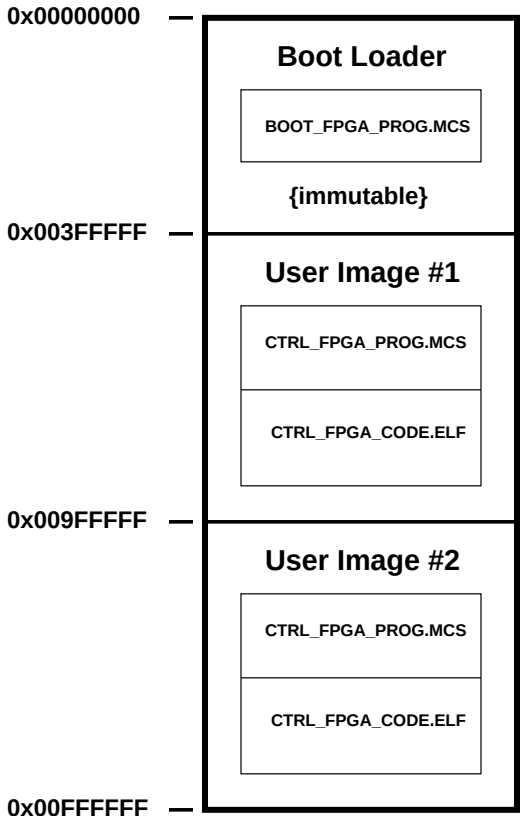


Figure 4.2: SPI Coniguration Flash PROM Organization

4.5.1 SPI_FLASH: ST Microelctronics M25P64 Flash Prom SPI Controller

The *SPI_FLASH* EDK Peripheral is an instance of the *xps_spi* IP and is the main interface to the ST Microelectronics M25P64 Flash Prom. The *SPI_FLASH* connects to the Spartan-3A FPGA pins shown in Table 4.13.

Table 4.13: **SPI_FLASH EDK Peripheral I/O Descriptions**

xps_spi Pin Name	Pin Name	Dir	Pin	Description
SCK	S3A_CCLK	O	AA20	Spartan-3A Configuration Clock.
SS	S3A_SPI_CSO	O	Y4	Spartan-3A SPI Chip Select (Active Low).
MISO	S3A_SPI_MISO	I	AB20	Spartan-3A SPI Master-In-Slave-Out (MISO).
MOSI	S3A_SPI_MOSI	O	AB14	Spartan-3A SPI Master-Out-Slave-In (MOSI).

4.5.2 GPIO_FLASH: ST Microelctronics M25P64 Flash Prom GPIO Controller

The *GPIO_FLASH* EDK Peripheral is used to control the write protect and hold pins of the ST Microelectronics M25P64 Flash PROM. The *GPIO_FLASH* EDK Peripheral connects to the Spartan-3A FPGA pins shown in Table 4.14.

Table 4.14: **GPIO_FLASH EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	S3A_SPI_HOLDN	O	AB13	1	Flash PROM Hold Signal (Active Low).
1	S3A_SPI_WPN	O	AA14	1	Flash PROM Write Protect Signal (Active Low).

4.6 AsAP v2 Configuration

The Measurement Board board contains two AsAP v2 DSP Processors, which need to be configured at power-up or at receipt of AsAP configuration command from the User Interface board. For information on the AsAP SPI Interface see Chapter 8.

The configuration data files for each AsAP will be stored in the microSD card. Each configuration data file will be retrieved upon power-up, and shifted serially to the appropriate AsAP device. The configuration data file is uploaded via either the User Interface board or the USB interface connected directly to the Control FPGA, which will in turn store the file on the microSD card. See Section 4.7 for information on the microSD card directory structure.

The Control FPGA will configure each of the AsAP v2 DSP Processors using a custom SPI Interface. The custom SPI interface consists of the following 8 signals:

1. SCK: Serial Configuration Clock
2. CSn: Chip Select (Active Low)
3. MOSI: Master-Out/Slave-In Serial Data
4. MISO: Master-In/Slave-Out Serial Data
5. LOAD_EN: Parallel Data Load Enable
6. CFG_CLK: Configuration Clock Pulse
7. CFG_VLD: Configuration Valid Pulse
8. RESET_COLD: Perform a Cold Reset (Active Low)

The outgoing serial data is sent in 20-bit packets/symbols, and the parallel data is formated as follows:

Table 4.15: AsAP Configuration Packet Descriptions

par[19]	par[18]	par[17:0]	Description
0b0	0b0	upper_addr	Upper Address Word (config_addr[35:18]).
0b1	0b0	lower_addr	Lower Address Word (config_addr[17:0]).
0b0	0b1	upper_data	Upper Data Word (config_data[35:18]).
0b1	0b1	lower_data	Lower Data Word (config_data[17:0]).

The par[18] bit is used for determining if address or data is being sent by the SPIE to the AsAP DSP Processor. If par[18] is a 0, then address is being sent. If par[18] is a 1, then data is being sent.

The par[19] bit is used for determining which part of the word is being sent: upper or lower. If par[19] is a 0, then the upper part of the word is being sent. If par[19] is a 1, then the lower part of the word is being sent.

The configuration blocks inside the AsAP DSP Processor know how to interpret the data based on the status of the 2 MSB bits of each 20-bit packet. Each of the 4 serial transactions outlined above are sent with a configuration clock and a configuration valid signal to clock the parallel data into the AsAP processor configuration blocks.

4.6.1 asap_config_0: AsAP Configuration Peripheral

The *asap_config_v1_00_a* EDK Peripheral is used to configure a single VCL AsAP version 2 IC, and appears as a set of 8 software accessible registers to the applications running on the MicroBlaze. The *asap_config_v1_00_a* EDK Peripheral's register map is shown in Table 4.16.

Table 4.16: **asap_config_v1_00_a EDK Peripheral Register Map**

Address	Name	R/W	Default Value	Description
0	config_data[31:0]	R/W	0x00000000	Lower 32-bits of Configuration Data Register.
1	{24'b0,config_data[39:32]}	R/W	0x00000000	Upper 8-bits of Configuration Data Register.
2	config_addr[31:0]	R/W	0x00000000	Lower 32-bits of Configuration Address Register.
3	{24'b0,config_addr[39:32]}	R/W	0x00000000	Upper 8-bits of Configuration Address Register.
4	{29'b0,spie_cfg[2:0]}	R/W	0x00000006	SPIE Configuration Control Register. [0:28]: reserved; [29]: start_spi: Start SPIE Transaction (Active High). Toggle high-then-low to initiate transaction. [30]: reset_cold: AsAP Cold Reset (Active Low). Toggle low-then-high to initiate transaction. [31]: reset_sel: Reset Selection. 0: 20ns wide retimed pulse. 1: MicroBlaze pulse high-low-high.
5	slv_reg5[31:0]	R/W	0x00000000	Reserved Register.
6	{29'b0,spie_stat[2:0]}	RO	0x00000000	SPIE Status Register. [0:28]: reserved; [29]: miso_rdy: MISO Read Data Ready (Active High). [30]: send: Ready to Start New Transaction (Active High). [31]: hold: 0: receiving data/addr. 1: transaction in process.
7	{12'b0,miso_read[19:0]}	RO	0x00000000	SPIE MISO Read Data Register.

To initiate a write tranaction using the *asap_config_v1_00_a* EDK Peripheral first set the 40-bit address registers (config_addr[39:0]) and 40-bit data registers (config_data[39:0]), then toggle the start_spi bit high-low to initiate the transfer. For subsequent write transactions wait for the send bit to go high. Also, before the config_addr[39:0] and config_data[39:0] registers can be updated, the application must wait for the hold bit to go high. When hold is low, the SPIE state machine is writing the current config_addr[39:0] and config_data[39:0] registers in to the upper_addr, lower_addr, upper_data, and lower_data registers.

To initiate a read tranaction using the *asap_config_v1_00_a* EDK Peripheral wait for the miso_rdy bit to go high, then read the data from the miso_read[19:0] register.

The *asap_config_0* connects to the Spartan-3A FPGA pins shown in Table 4.17.

Table 4.17: **asap_config_0 EDK Peripheral I/O Descriptions**

asap_config_v1_00_a Pin Name	Pin Name	Dir	Pin	Description
i_spi_asap_miso	FPGA_ASAP1_MISO	I	AA22	Master-In/Slave-Out Serial Data.
o_spi_asap_cfg_clk	FPGA_ASAP1_CFG_CLK	O	W20	Configuration Clock Pulse.
o_spi_asap_cfg_vld	FPGA_ASAP1_CFG_VALID	O	W19	Configuration Valid Pulse.
o_spi_asap_sck	FPGA_ASAP1_SPI_CLK	O	Y21	Serial Configuration Clock.
o_spi_asap_csn	FPGA_ASAP1_SPI_CSN	O	W22	Chip Select (Active Low).
o_spi_asap_mosi	FPGA_ASAP1_SPI_MOSI	O	Y22	Master-Out/Slave-In Serial Data.
o_spi_asap_load_en	FPGA_ASAP1_SPI_LOAD	O	W21	Parallel Data Load Enable (Active High).
o_asap_reset_cold	FPGA_ASAP1_RESET_COLD	O	V19	Perform a Cold Reset (Active Low).
o_asap_rst_cntclk	FPGA_ASAP1_RST_CNTCLK	O	V20	Reset Counter Clock (Active Low).

The *asap_config_1* connects to the Spartan-3A FPGA pins shown in Table 4.18.

Table 4.18: **asap_config_1 EDK Peripheral I/O Descriptions**

asap_config_v1_00_a Pin Name	Pin Name	Dir	Pin	Description
i_spi_asap_miso	FPGA_ASAP2_MISO	I	W18	Master-In/Slave-Out Serial Data.
o_spi_asap_cfg_clk	FPGA_ASAP2_CFG_CLK	O	AA21	Configuration Clock Pulse.
o_spi_asap_cfg_vld	FPGA_ASAP2_CFG_VALID	O	AB21	Configuration Valid Pulse.
o_spi_asap_sck	FPGA_ASAP2_SPI_CLK	O	AB18	Serial Configuration Clock.
o_spi_asap_csn	FPGA_ASAP2_SPI_CSN	O	AA19	Chip Select (Active Low).
o_spi_asap_mosi	FPGA_ASAP2_SPI_MOSI	O	Y18	Master-Out/Slave-In Serial Data.
o_spi_asap_load_en	FPGA_ASAP2_SPI_LOAD	O	AB19	Parallel Data Load Enable (Active High).
o_asap_reset_cold	FPGA_ASAP2_RESET_COLD	O	AB17	Perform a Cold Reset (Active Low).
o_asap_rst_cntclk	FPGA_ASAP2_RST_CNTCLK	O	AA17	Reset Counter Clock (Active Low).

4.7 microSD Card Organization

The microSD card will be used to store both configuration and calibration data. The microSD card is accessed and controlled using both an SPI interface and a FAT File System, and as such, must abide by the file name, size, and path restrictions. The current microSD card contains 2GB of storage capacity, which means we will be implementing a FAT16 File System in the MicroBlaze 32-bit micro-controller residing in the Spartan 3A FPGA. The maximum filename length is 8.3, therefore the descriptor can be at most 8 characters and the extension can be at most 3 characters; there is no limit defined on the maximum pathname length. A proposed directory structure and file naming convention is shown below:

```
ASAP/
        PROG1/
                ASAP1.BIN
                ASAP2.BIN
                RUN.BIN
        PROG2/
                ASAP1.BIN
                ASAP2.BIN
                RUN.BIN
        PROG3/
                ASAP1.BIN
                ASAP2.BIN
                RUN.BIN
CONFIG/
        FACTORY/
                CPIMAGE.MCS
                DPIMAGE.BIN
        IMAGE1/
                CPIMAGE.MCS
                DPIMAGE.BIN
        IMAGE2/
                CPIMAGE.MCS
                DPIMAGE.BIN
```

4.7.1 SPI_SD: microSD Card SPI Controller

The *SPI_SD* EDK Peripheral is an instance of the *vcl_spi* IP and is the main interface to the microSD card. The *SPI_SD* connects to the Spartan-3A FPGA pins shown in Table 4.19.

Table 4.19: **SPI_SD EDK Peripheral I/O Descriptions**

vcl_spi Pin Name	Pin Name	Dir	Pin	Description
MOSI	FPGA_SD_TX	O	AB2	microSD Card Receive Data Input.
SS	FPGA_SD_CARD_DETECT	I/O	AA4	microSD Card Chip Select (Active Low).
SCK	FPGA_SD_CLK	O	AA3	microSD Card Serial Clock.
MISO	FPGA_SD_RX	I	AB3	microSD Card Transmit Data Output.

4.7.2 GPIO_SD: microSD Card GPIO Controller

The *GPIO_SD* EDK Peripheral is used to control the busy LED placed next to the microSD card. The Control Application will blink the LED as a warning while the microSD card is being accessed in order to avoid accidental removal and data loss. The *GPIO_SD* EDK Peripheral connects to the Spartan-3A FPGA pins shown in Table 4.20.

Table 4.20: **GPIO_SD EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	FPGA_SD_BUSY_LED	O	AB4	1	microSD Card Busy LED (Active Low).

4.8 Digital-to-Analog Converter

For most of the instrument modes dynamic range is very important; however, both high resolution and high sample rate in a single Digital-to-Analog Converter (DAC) are hard to find. A trade-off must be made between sample rate and resolution. Several D/A converters are currently being considered for the measurement board:

- Texas Instruments 16-Bit, 1.0 GSPS 2x-4x Interpolating Dual-Channel DAC (DAC5682Z)
  - http://focus.ti.com/lit/ds/symlink/dac5682z.pdf

Table 4.21: TI DAC5682Z DAC Register Settings

TI DAC5682Z DAC Register Settings			
Register Name	**Register Address**	**Value**	**Description**
STATUS0	0x00	0x03	PLL/DLL Locked Status.
CONFIG1	0x01	0x00	DAC Delay, FIR Enable, Self Test, and FIFO Offset Control.
CONFIG2	0x02	0x80	Two's Comp Enable, Dual/Single DAC Mode, and FIR Mode Control.
CONFIG3	0x03	0x00	DAC Offset Enable, A/B Swap, A equal B, SW Sync, and SW Sync Enable.
STATUS4	0x04	0x00	Self Test Error, FIFO Error, and Pattern Error Status.
CONFIG5	0x05	0x92	SPI Setup, Clock Divider, FIFO Offset Sync, and PLL/DLL Bypass Control.
CONFIG6	0x06	0xae	Hold Sync, Sleep, Bias and PLL/DLL Sleep Control.
CONFIG7	0x07	0xff	Channel A and B Gain Control.
CONFIG8	0x08	0x00	DLL Restart Control. Toggle bit 2 high then low.
CONFIG9	0x09	0x00	PLL M/N Control.
CONFIG10	0x0a	0x00	DLL Delay and Clock Control.
CONFIG11	0x0b	0x00	PLL Loop Filter, VCO Divider, PLL Gain and Range Control.
CONFIG12	0x0c	0x00	Offset Sync and Channel A Offset (MSB) Control.
CONFIG13	0x0d	0x00	Channel A Offset (LSB) Control.
CONFIG14	0x0e	0x00	SPI Serial Data Out and Channel B Offset (MSB) Control.
CONFIG15	0x0f	0x00	Channel B Offset (LSB) Control.

4.8.1 SPI_DAC5682Z: TI DAC5682Z High-Speed DAC IC SPI Controller

The *SPI_DAC5682Z* EDK Peripheral is an instance of the *xps_spi* IP and is the main interface to the DAC5682Z High-Speed DAC IC [1]. The *SPI_DAC5682Z* connects to the Spartan-3A FPGA pins shown in Table 4.22.

Table 4.22: **SPI_DAC5682Z EDK Peripheral I/O Descriptions**

xps_spi Pin Name	**Pin Name**	**Dir**	**Pin**	**Description**
SCK	FPGA_DAC5682_SCLK	O	V22	DAC5682Z Serial Clock.
MOSI	FPGA_DAC5682_SDIO	O	U22	DAC5682Z Serial Data Input.
MISO	FPGA_DAC5682_SDO	I	U19	DAC5682Z Serial Data Output.
SS	FPGA_DAC5682_SDENB	O	U21	DAC5682Z Chip Select (Active Low).

4.8.2 GPIO_DAC5682Z_OUTS: TI DAC5682Z High-Speed DAC IC GPIO Controller

The *GPIO_DAC5682Z_OUTS* EDK Peripheral is used to control the non-spi pins of the TI DAC5682Z High-Speed DAC IC. The *GPIO_DAC5682Z_OUTS* EDK Peripheral connects to the Spartan-3A FPGA pins shown in Table 4.23.

Table 4.23: **GPIO_DAC5682Z_OUTS EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	FPGA_DAC5682_RSTB	O	U20	1	DAC5682Z Reset (Active Low).

4.9 10MHz Control

The Measurement Board board contains a Vectron VTC2 10MHz voltage controlled temperature compensated crystal oscillator (TCXO) for use as a precision reference. The VTC2 is a ±0.5ppm quartz stabilized, CMOS squarewave, temperature compensated oscillator, operating off a 3.3V supply.

- http://vectron.com/products/tcxo/VTC2.pdf

The Internal 10MHz reference can be used as the reference to the AD9516-3 Clock Generator IC. If the user requires a more precise 10MHz reference clock, then they can provide the Measurement board with an external 10MHz reference via the rear-panel connector and enable its input.

The Internal 10MHz reference is also used to generate a common 300kHz sync clock for the Texas Instruments PTH08T2xxWAZ DC-DC Converters. A Xilinx CPLD (XC9572XL) is employed to perform the divide-by-32 and multi-phase clock generation.

Table 4.24: 10MHz Control Descriptions

Bit	Description
0	Interal/External 10MHz Reference Select; 0 = internal, 1 = external.
1	External 10MHz Enable; 0 = disabled, 1 = enabled.
2	External 10MHz Input Disable; 0 = enabled, 1 = disabled.
3	Internal 10MHz Enable; 0 = disabled, 1 = enabled.

Table 4.25: 10MHz Control Register

Bits				Clock Mode Description
3	2	1	0	
1	1	0	0	Internal 10MHz Clock Selected **(Default)**
0	0	1	1	External 10MHz Clock Selected
1	0	1	0	Internal 10MHz used as reference, External 10MHz pass to Control FPGA.

The *CLK10MHZ_CTRL_OUTS* EDK Peripheral connects to the Spartan-3A FPGA pins shown in Table 4.26.

Table 4.26: **CLK10MHZ_CTRL_OUTS EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	FPGA_CLK10MHZ_REF_CTRL[0]	O	B2	1	10MHz Ref. Select; 1 = int, 0 = ext.
1	FPGA_CLK10MHZ_REF_CTRL[1]	O	B3	1	10MHz Ref. Select; 1 = int, 0 = ext.
2	FPGA_CLK10MHZ_REF_CTRL[2]	O	A3	1	10MHz Ref. Select; 1 = int, 0 = ext.
3	FPGA_CLK10MHZ_REF_CTRL[3]	O	A4	1	10MHz Ref. Select; 1 = int, 0 = ext.

The *CLK10MHZ_CTRL_INS* EDK Peripheral connects to the Spartan-3A FPGA pins shown in Table 4.27.

Table 4.27: **CLK10MHZ_CTRL_INS EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	FPGA_EXT_CLK10MHZ_LOS	I	A2	NA	10MHz LOS; 1 = Loss of Signal, 0 = Signal Present.

4.10 Clock Generation

The Analog Devices AD9516 Clock Generator IC does not require programming during normal use, but does need to be initialized during the normal power-up routines. Once the initialization is complete, then the AD9516 can generate the appropriate clock frequencies. Table 4.28 shows the generated output frequencies of the active AD9516 Clock Generator IC outputs. Table 4.29 shows the default value for each of the AD9516 Clock Generator IC registers. Writing to the register at 0x232 applies all updates to the AD9516 Clock Generator IC. The data sheet for the AD9516 can be found here:

- Analog Devices AD9516-3 14-Output Clock Generator with Integrated 2.0 GHz VCO [2]
  - http://www.analog.com/UploadedFiles/Data_Sheets/AD9516_3.pdf

The Universal Microwave Corp UMX Series 1GHz VCO (UMX-244-B14) is used to drive the AD9516-3 External Clock Input.

- http://www.vco1.com/SCDs/scd244-a.pdf

4.10.1 AD9516 Clock Generator Output Assignments

Table 4.28: AD9516 Clock Generator Outputs

Output Number	Description
0	ADC Sampling Clock (500MHz).
1	DAC Sampling Clock (500MHz).
2	FPGA AsAP Clock (250MHz).
3	FPGA DSP Clock (250MHz).
4	Unused Output.
5	Unused Output.
6	FPGA SRAM Clock (250MHz).
7	FPGA SDRAM Clock (250MHz).
8	FPGA IODELAY Reference Clock (200MHz).
9	Test Clock Output.

4.10.2 AD9516 Clock Generator PLL Calculations

To determine the appropriate values of PLL parameters we need to use Equations 4.1 and 4.2.

$$f_{vco} = \left(\frac{f_{ref}}{R}\right) \cdot ((P \cdot B) + A) = f_{ref} \cdot \left(\frac{N}{R}\right) \tag{4.1}$$

$$N = ((P \cdot B) + A) \tag{4.2}$$

The known variables of Equation 4.1 are:

- f_{vco} = 1 GHz
- f_{ref} = 10 MHz

Solving for N given R = 1, we see that N should be equal to 100. Now we can solve for P, B, and A in Equation 4.2. First assume that A = 0 and P = 4. Solving for B in Equation 4.2 yields 25. A summary of the calculated values is shown below:

- R = 1
- P = 4
- B = 25
- A = 0

4.10.3 AD9516 Clock Generator Registers

Table 4.29: AD9516 Clock Generator Register Settings

AD9516 Clock Generator Register Settings		
Register Address	**Value**	**Description**
Serial Port Configuration		
0x000	0xbd	Serial Port Configuration (Reset SPI Registers).
0x000	0x99	Serial Port Configuration (Normal Operation).
0x001	blank	Unused Register
0x002	reserved	Unused Register
0x003	reserved	Unused Register
0x004	0x01	Read Back Control
PLL		
0x010	0x4c	PFD and Charge Pump
0x011	0x01	R Counter
0x012	0x00	R Counter
0x013	0x00	A Counter
0x014	0x19	B Counter
0x015	0x00	B Counter
0x016	0x03	PLL Control 1
0x017	0x00	PLL Control 2
0x018	0x66	PLL Control 3
0x019	0x00	PLL Control 4
0x01a	0x00	PLL Control 5
0x01b	0x00	PLL Control 6
0x01c	0x07	PLL Control 7
0x01d	0x00	PLL Control 8
0x01e	0x00	PLL Control 9
0x01f	0x00	PLL Readback
0x020 - 0x04f	blank	Unused Registers
Fine Delay Adjust: OUT6 to OUT9		
0x0a0	0x01	OUT6 Delay Bypass
0x0a1	0x00	OUT6 Delay Full-Scale
0x0a2	0x00	OUT6 Delay Fraction
0x0a3	0x01	OUT7 Delay Bypass
0x0a4	0x00	OUT7 Delay Full-Scale
0x0a5	0x00	OUT7 Delay Fraction
0x0a6	0x01	OUT8 Delay Bypass
0x0a7	0x00	OUT8 Delay Full-Scale
0x0a8	0x00	OUT8 Delay Fraction
0x0a9	0x01	OUT9 Delay Bypass
0x0aa	0x00	OUT9 Delay Full-Scale
0x0ab	0x00	OUT9 Delay Fraction
0x0ac - 0x0ef	blank	Unused Registers
LVPECL Outputs		
0x0f0	0x0c	OUT0
0x0f1	0x0c	OUT1
0x0f2	0x08	OUT2
0x0f3	0x08	OUT3
0x0f4	0x03	OUT4
0x0f5	0x03	OUT5
		Continued on Next Page...

AD9516 Clock Generator Register Settings		
Register Address	**Value**	**Description**
0x0f6 - 0x13f	blank	Unused Registers
LVDS/CMOS Outputs		
0x140	0x02	OUT6
0x141	0x02	OUT7
0x142	0x01	OUT8
0x143	0x02	OUT9
0x144 - 0x18f	blank	Unused Registers
LVPECL Channel Dividers		
0x190	0x00	Divider 0 (PECL)
0x191	0x40	Divider 0 (PECL)
0x192	0x01	Divider 0 (PECL)
0x193	0x44	Divider 1 (PECL)
0x194	0x40	Divider 1 (PECL)
0x195	0x01	Divider 1 (PECL)
0x196	0x00	Divider 2 (PECL)
0x197	0x00	Divider 2 (PECL)
0x198	0x00	Divider 2 (PECL)
LVDS/CMOS Channel Dividers		
0x199	0x00	Divider 3 (LVDS/CMOS)
0x19a	0x00	Divider 3 (LVDS/CMOS)
0x19b	0x00	Divider 3 (LVDS/CMOS)
0x19c	0x08	Divider 3 (LVDS/CMOS)
0x19d	0x01	Divider 3 (LVDS/CMOS)
0x19e	0x32	Divider 4 (LVDS/CMOS)
0x19f	0x00	Divider 4 (LVDS/CMOS)
0x1a0	0x11	Divider 4 (LVDS/CMOS)
0x1a1	0x28	Divider 4 (LVDS/CMOS)
0x1a2	0x01	Divider 4 (LVDS/CMOS)
0x1a3	reserved	Unused Register
0x1a4 - 0x1df	blank	Unused Registers
VCO Divider and CLK Input		
0x1e0	0x02	VCO Divider
0x1e1	0x01	Input CLKs
0x1e2 - 0x22a	blank	Unused Registers
System		
0x230	0x04	Power Down and Sync
0x231	blank/reserved	Unused Register
Update All Registers		
0x232	0x01	Update All Registers

4.10.4 vcl_ad9516_spi_0: AD9516 Clock Generator Control Peripheral

The *vcl_ad9516_spi_v1_00_a* EDK Peripheral is used to control the AD9516 Clock Generator IC, and appears as a set of 8 software accessible registers to the applications running on the MicroBlaze. The *vcl_ad9516_spi_v1_00_a* EDK Peripheral's register map is shown in Table 4.30.

Table 4.30: **vcl_ad9516_spi_v1_00_a EDK Peripheral Register Map**

Address	Name	R/W	Default Value	Description
0	{24'b0,ad9516_wr_data[7:0]}	R/W	0x00000000	8-bit Write Data Register.
1	{22'b0,ad9516_addr[9:0]}	R/W	0x00000000	10-bit Address Register.
2	{28'b0,ad9516_control[3:0]}	R/W	0x0000000C	AD9516 Control Register. [0:27]: reserved; [28]: ad9516_ref_sel: Reference Select; [29]: ad9516_pdn: Power Down Enable (Active Low). [30]: ad9516_rnw: R/nW Select; 0 = Write, 1 = Read. [31]: fsm_start: Start current SPI transaction. Toggle high-then-low to iniate transaction.
3	{31'b0,periph_control[0]}	R/W	0x00000000	Peripheral Control Register. [0:30]: reserved; [31]: mrst: Peripheral Reset (Active High).
4	slv_reg4[0:31]	R/W	0x00000000	Reserved Register.
5	slv_reg5[0:31]	R/W	0x00000000	Reserved Register.
6	slv_reg6[0:31]	R/W	0x00000000	Reserved Register.
7	slv_reg7[0:31]	R/W	0x00000000	Reserved Register.
8	{31'b0,ad9516_done}	RO	0x00000000	SPI Transfer Done Register.
				[0:30]: reserved; [31]: ad9516_done: SPI Transfer Done (Active High).
9	{24'b0,ad9516_rd_data[7:0]}	RO	0x00000000	8-bit Read Data Register.
A	slv_reg10[0:31]	RO	0xAAAAAAAA	Reserved Register.
B	slv_reg11[0:31]	RO	0xAAAAAAAA	Reserved Register.
C	slv_reg12[0:31]	RO	0xAAAAAAAA	Reserved Register.
D	slv_reg13[0:31]	RO	0xAAAAAAAA	Reserved Register.
E	slv_reg14[0:31]	RO	0xAAAAAAAA	Reserved Register.
F	slv_reg15[0:31]	RO	0xAAAAAAAA	Reserved Register.

The *vcl_ad9516_spi_v1_00_a* connects to the Spartan-3A FPGA pins shown in Table 4.31.

Table 4.31: **vcl_ad9516_spi_v1_00_a EDK Peripheral I/O Descriptions**

xps_spi Pin Name	Pin Name	Dir	Pin	Description
i_fpga_ad9516_sdo	FPGA_AD9516_SDO	I	P22	Master-In/Slave-Out Serial Data.
o_fpga_ad9516_sdin	FPGA_AD9516_SDIO	O	N20	Master-Out/Slave-In Serial Data.
o_fpga_ad9516_sclk	FPGA_AD9516_SCLK	O	N18	Serial Clock.
o_fpga_ad9516_csn	FPGA_AD9516_CSB	O	N19	Chip Select (Active Low).
o_fpga_ad9516_rstn	FPGA_AD9516_RESET	O	N21	Reset (Active Low).
o_fpga_ad9516_ref_sel	FPGA_AD9516_REF_SEL	O	N22	Reference Select.
o_fpga_ad9516_pdn	FPGA_AD9516_PD	O	P20	Power Down Enable.

4.10.5 GPIO_AD9516_INS: ADI AD9516 GPIO Controller

The *GPIO_AD9516_INS* EDK Peripheral is used to control the non-spi pins of the Analog Devices AD9516 Clock Generator IC. The *GPIO_AD9516_OUTS* EDK Peripheral connects to the Spartan-3A FPGA pins shown in Table 4.32.

Table 4.32: **GPIO_AD9516_INS EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	FPGA_AD9516_LD	I	P19	1	AD9516 Lock Detect.
1	FPGA_AD9516_REFMON	I	M17	1	AD9516 Reference Monitor.
1	FPGA_AD9516_STATUS	O	N17	1	AD9516 Status.

4.11 Instrument Fan Control

The Measurement Board board contains two instrument fan controllers for maintaining proper cooling and airflow, while avoiding the generation of beat frequencies between the two fans. The Texas Instruments AMC6821 [3] is an intelligent temperature monitor and pulse-width modulation (PWM) fan controller. Using either a low-frequency or a high-frequency PWM signal, this device can simultaneously drive a fan, monitor remote sensor diode temperatures, and measure and control the fan speed so that it operates with minimal acoustic noise at the lowest possible speed.

The AMC6821 has three fan control modes:

1. Auto Temperature-Fan mode
2. Software-RPM mode
3. Software-DCY mode

The Measurement Board uses the AMC6821 in *Software-RPM mode*. The Software-RPM mode is a closed-loop control. In this mode, the AMC6821 adjusts the PWM output to maintain a consistent fan speed at a user-specified target value; that is, the device functions as a fan speed regulator. Software-RPM mode can also be used to allow the AMC6821 to operate as a stand-alone device.

The I^2C address of the two Fan Controllers is:

1. 0b1001100
2. 0b1001101

4.11.1 AMC6821 Fan Controller #1 Registers

Table 4.33: AMC6821 Fan Controller #1 Register Settings

AMC6821 Fan Controller #1 Register Settings			
Register Address	**Value**	**R/W**	**Description**
IDENTIFICATION REGISTERS			
0x3D	0x21	R	Device ID Register
0x3E	0x49	R	Company ID Register
CONFIGURATION REGISTERS			
0x00	0xb5	R/W	Configuration Register 1
0x01	0x3d	R/W	Configuration Register 2
0x3F	0x82	R/W	Configuration Register 3
0x04	0x88	R/W	Configuration Register 4
0x02	0x00	R	Status Register 1
0x03	0x00	R	Status Register 2
TEMPERATURE MONITORING			
0x06	0x00	R	Temp-DATA-LByte
0x0A	0x80	R	Local-Temp-DATA-HByte
0x0B	0x80	R	Remote-Temp-DATA-HByte
0x14	0x3c	R/W	Local-High-Temp-Limit
0x15	0x00	R/W	Local-Low-Temp-Limit
0x16	0x46	R/W	Local-THERM-Limit
0x18	0x50	R/W	Remote-High-Temp-Limit
0x19	0x00	R/W	Remote-Low-Temp-Limit
0x1a	0x64	R/W	Remote-THERM-Limit
0x1b	0x50	R/W	Local-Critical-Temp
0x1c	0x00	R/W	PSV-Temp
0x1d	0x69	R/W	Remote-Critical-Temp
PWM CONTROLLER			
0x20	0x1d	R/W	FAN-Characteristics
0x21	0x55	R/W	DCY-Low-Temp
0x22	0x55	R/W	DCY (Duty Cycle)
0x23	0x52	R/W	DCY-RAMP
0x24	0x41	R/W	Local Temp-Fan Control
0x25	0x61	R/W	Remote Temp-Fan Control
TACH (RPM) MEASUREMENT			
0x08	0x00	R	TACH-DATA-LByte
0x09	0x00	R	TACH-DATA-HByte
0x10	0xff	R/W	TACH-Low-Limit-LByte
0x11	0xff	R/W	TACH-Low-Limit-HByte
0x12	0x00	R/W	TACH-High-Limit-LByte
0x13	0x00	R/W	TACH-High-Limit-HByte
0x1E	0xff	R/W	TACH-SETTING-LByte
0x1F	0xff	R/W	TACH-SETTING-HByte
0x3A	0x00	R	Reserved
0x3B	0x00	R	Reserved

4.11.2 vcl_iic_ctrlr_0: I^{2}C Control Peripheral

The *vcl_iic_ctrlr_v1_00_a* EDK Peripheral is used to control the I^{2}C Fan Controller, and appears as a set of 16 software accessible registers to the applications running on the MicroBlaze. The *vcl_iic_ctrlr_v1_00_a* EDK Peripheral's register map is shown in Table 4.36.
The *vcl_iic_ctrlr_0* connects to the Spartan-3A FPGA pins shown in Table 4.34.

Table 4.34: **vcl_iic_ctrlr_0 EDK Peripheral I/O Descriptions**

vcl_iic_ctrlr_v1_00_a Pin Name	Pin Name	Dir	Pin	Description
io_fpga_iic_sda	FPGA_AMC6821_FAN1_SDA	IO	Y7	AMC6821 #1 I^{2}C Serial Data.
io_fpga_iic_scl	FPGA_AMC6821_FAN1_SCK	O	W7	AMC6821 #1 I^{2}C Serial Clock.

The *GPIO_FAN_CTRL1* connects to the Spartan-3A FPGA pins shown in Table 4.35.

Table 4.35: **GPIO_FAN_CTRL1 EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Description
0	FPGA_AMC6821_FAN1_OVRN	I	T8	AMC6821 #1 Over Temp Flag (Active Low).
1	FPGA_AMC6821_FAN1_THERMN	I	U7	AMC6821 #1 Thermal Flag (Active Low).
2	FPGA_AMC6821_FAN1_FAULTN	I	T7	AMC6821 #1 Fault Flag (Active Low).
3	FPGA_AMC6821_FAN1_SMBALERTN	I	V7	AMC6821 #1 SMB Alert Flag (Active Low).

Table 4.36: **vcl_iic_ctrlr_v1_00_a EDK Peripheral Register Map**

Address	Name	R/W	Default Value	Description
0	{16'b0,iic_prer[15:0]}	R/W	0x00000063	I^2C Clock Prescaler Register.
1	{30'b0,iic_ctrl[1:0]}	R/W	0x00000000	[0:15]: reserved; [16:31]: iic_prer[15:0]: I^2C Prescale.. I^2C Control Register.
2	{24'b0,iic_txr[7:1],iic_rnw}	R/W	0x00000000	[0:29]: reserved; [30]: iic_core_en: I^2C Module Enable. (Active High) [31]: iic_int_en: I^2C Interrupt Enable. (Active High) I^2C Transmit Data Register.
3	{26'b0,iic_cmd[5:0]}	R/W	0x00000000	[0:23]: reserved; [24:30]: iic_txr: I^2C Transmit Data. [31]: iic_rnw: I^2C Read/Write Bit, 0 = Write, 1 = Read. I^2C Command Register.
4	{31'b0,iic_rst}	R/W	0x00000000	[0:25]: reserved; [26]: iic_iack: Interrupt Acknowledge. (Active High) When set, clears a pending interrupt. [27]: iic_ack: When a reciever, sent ACK (iic_ack = '0') or NACK (iic_ack = '1'). [28]: iic_wr: Write to a Slave. (Active High) [29]: iic_rd: Read from a Slave. (Active High) [30]: iic_sto: Generate Stop Condition. (Active High) [31]: iic_sta: Generate Start Condition. (Active High) I^2C Reset Register. [0:30]: reserved; [31]: iic_rst: I^2C Module Reset. (Active High). Toggle high then low to reset vcl_iic_ctrlr module.
5	slv_reg5[0:31]	R/W	0x00000000	Reserved Register.
6	slv_reg6[0:31]	R/W	0x00000000	Reserved Register.
7	slv_reg7[0:31]	R/W	0x00000000	Reserved Register.
8	{24'b0,iic_rxr[7:0]}	RO	0x00000000	I^2C Receive Data Register.
9	{26'b0,iic_stat[4:0]}	RO	0x00000000	I^2C Status Register.
A	slv_reg10[0:31]	RO	0x00000000	[0:26]: reserved; [27]: rxack: Received Acknowledge from Slave. 1 = No Acknowdge Received. 0 = Acknowdge Received. [28]: iic_busy: I^2c Busy. 1 after START signal detected. 0 after STOP signal detected. [29]: al: Arbitration Lost. Arbitration is lost when: a STOP signal is detected, but not requested. The master drives SDA high, but SDA is low. [30]: tip: Transfer in Progress. 1 = Transferring Data. 0 = Transfer Complete. [31]: irq_flag: Interrupt Flag. (Active High) This bit is set when an interrupt is pending. Reserved Register.
B	slv_reg11[0:31]	RO	0x00000000	Reserved Register.
C	slv_reg12[0:31]	RO	0x00000000	Reserved Register.
D	slv_reg13[0:31]	RO	0x00000000	Reserved Register.
E	slv_reg14[0:31]	RO	0x00000000	Reserved Register.
F	slv_reg15[0:31]	RO	0x00000000	Reserved Register.

4.11.3 AMC6821 Fan Controller #2 Registers

Table 4.37: AMC6821 Fan Controller #2 Register Settings

AMC6821 Fan Controller #2 Register Settings			
Register Address	**Value**	**R/W**	**Description**
IDENTIFICATION REGISTERS			
0x3D	0x21	R	Device ID Register
0x3E	0x49	R	Company ID Register
CONFIGURATION REGISTERS			
0x00	0xb5	R/W	Configuration Register 1
0x01	0x3d	R/W	Configuration Register 2
0x3F	0x82	R/W	Configuration Register 3
0x04	0x88	R/W	Configuration Register 4
0x02	0x00	R	Status Register 1
0x03	0x00	R	Status Register 2
TEMPERATURE MONITORING			
0x06	0x00	R	Temp-DATA-LByte
0x0A	0x80	R	Local-Temp-DATA-HByte
0x0B	0x80	R	Remote-Temp-DATA-HByte
0x14	0x3c	R/W	Local-High-Temp-Limit
0x15	0x00	R/W	Local-Low-Temp-Limit
0x16	0x46	R/W	Local-THERM-Limit
0x18	0x50	R/W	Remote-High-Temp-Limit
0x19	0x00	R/W	Remote-Low-Temp-Limit
0x1a	0x64	R/W	Remote-THERM-Limit
0x1b	0x50	R/W	Local-Critical-Temp
0x1c	0x00	R/W	PSV-Temp
0x1d	0x69	R/W	Remote-Critical-Temp
PWM CONTROLLER			
0x20	0x25	R/W	FAN-Characteristics
0x21	0x55	R/W	DCY-Low-Temp
0x22	0x55	R/W	DCY (Duty Cycle)
0x23	0x52	R/W	DCY-RAMP
0x24	0x41	R/W	Local Temp-Fan Control
0x25	0x61	R/W	Remote Temp-Fan Control
TACH (RPM) MEASUREMENT			
0x08	0x00	R	TACH-DATA-LByte
0x09	0x00	R	TACH-DATA-HByte
0x10	0xff	R/W	TACH-Low-Limit-LByte
0x11	0xff	R/W	TACH-Low-Limit-HByte
0x12	0x00	R/W	TACH-High-Limit-LByte
0x13	0x00	R/W	TACH-High-Limit-HByte
0x1E	0xff	R/W	TACH-SETTING-LByte
0x1F	0xff	R/W	TACH-SETTING-HByte
0x3A	0x00	R	Reserved
0x3B	0x00	R	Reserved

4.11.4 vcl_iic_ctrlr_1: I^{2}C Control Peripheral

The *vcl_iic_ctrlr_v1_00_a* EDK Peripheral is used to control the I^{2}C Fan Controller, and appears as a set of 16 software accessible registers to the applications running on the MicroBlaze. The *vcl_iic_ctrlr_v1_00_a* EDK Peripheral's register map is shown in Table 4.40.
The *vcl_iic_ctrlr_1* connects to the Spartan-3A FPGA pins shown in Table 4.38.

Table 4.38: **vcl_iic_ctrlr_1 EDK Peripheral I/O Descriptions**

vcl_iic_ctrlr_v1_00_a Pin Name	Pin Name	Dir	Pin	Description
io_fpga_iic_sda	FPGA_AMC6821_FAN2_SDA	IO	Y9	AMC6821 #2 I^{2}C Serial Data.
io_fpga_iic_scl	FPGA_AMC6821_FAN2_SCK	O	AB9	AMC6821 #2 I^{2}C Serial Clock.

The *GPIO_FAN_CTRL2* connects to the Spartan-3A FPGA pins shown in Table 4.39.

Table 4.39: **GPIO_FAN_CTRL2 EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Description
0	FPGA_AMC6821_FAN2_OVRN	I	V9	AMC6821 #2 Over Temp Flag (Active Low).
1	FPGA_AMC6821_FAN2_THERMN	I	V8	AMC6821 #2 Thermal Flag (Active Low).
2	FPGA_AMC6821_FAN2_FAULTN	I	U8	AMC6821 #2 Fault Flag (Active Low).
3	FPGA_AMC6821_FAN2_SMBALERTN	I	T9	AMC6821 #2 SMB Alert Flag (Active Low).

Table 4.40: **vcl_iic_ctrlr_v1_00_a EDK Peripheral Register Map**

Address	Name	R/W	Default Value	Description
0	{16'b0,iic_prer[15:0]}	R/W	0x00000063	I^2C Clock Prescaler Register.
1	{30'b0,iic_ctrl[1:0]}	R/W	0x00000000	[0:15]: reserved; [16:31]: iic_prer[15:0]: I^2C Prescale.. I^2C Control Register.
2	{24'b0,iic_txr[7:1],iic_rnw}	R/W	0x00000000	[0:29]: reserved; [30]: iic_core_en: I^2C Module Enable. (Active High) [31]: iic_int_en: I^2C Interrupt Enable. (Active High) I^2C Transmit Data Register.
3	{26'b0,iic_cmd[5:0]}	R/W	0x00000000	[0:23]: reserved; [24:30]: iic_txr: I^2C Transmit Data. [31]: iic_rnw: I^2C Read/Write Bit, 0 = Write, 1 = Read. I^2C Command Register.
4	{31'b0,iic_rst}	R/W	0x00000000	[0:25]: reserved; [26]: iic_iack: Interrupt Acknowledge. (Active High) When set, clears a pending interrupt. [27]: iic_ack: When a reciever, sent ACK (iic_ack = '0') or NACK (iic_ack = '1'). [28]: iic_wr: Write to a Slave. (Active High) [29]: iic_rd: Read from a Slave. (Active High) [30]: iic_sto: Generate Stop Condition. (Active High) [31]: iic_sta: Generate Start Condition. (Active High) I^2C Reset Register. [0:30]: reserved; [31]: iic_rst: I^2C Module Reset. (Active High). Toggle high then low to reset vcl_iic_ctrlr module.
5	slv_reg5[0:31]	R/W	0x00000000	Reserved Register.
6	slv_reg6[0:31]	R/W	0x00000000	Reserved Register.
7	slv_reg7[0:31]	R/W	0x00000000	Reserved Register.
8	{24'b0,iic_rxr[7:0]}	RO	0x00000000	I^2C Receive Data Register.
9	{26'b0,iic_stat[4:0]}	RO	0x00000000	I^2C Status Register.
A	slv_reg10[0:31]	RO	0x00000000	[0:26]: reserved; [27]: rxack: Received Acknowledge from Slave. 1 = No Acknowdge Received. 0 = Acknowdge Received. [28]: iic_busy: I^2c Busy. 1 after START signal detected. 0 after STOP signal detected. [29]: al: Arbitration Lost. Arbitration is lost when: a STOP signal is detected, but not requested. The master drives SDA high, but SDA is low. [30]: tip: Transfer in Progress. 1 = Transferring Data. 0 = Transfer Complete. [31]: irq_flag: Interrupt Flag. (Active High) This bit is set when an interrupt is pending. Reserved Register.
B	slv_reg11[0:31]	RO	0x00000000	Reserved Register.
C	slv_reg12[0:31]	RO	0x00000000	Reserved Register.
D	slv_reg13[0:31]	RO	0x00000000	Reserved Register.
E	slv_reg14[0:31]	RO	0x00000000	Reserved Register.
F	slv_reg15[0:31]	RO	0x00000000	Reserved Register.

4.12 Temperature Sensing

The Measurement Board contains 6 digital temperature sensors in a 2x3 array. The Texas Instruments TMP125 [4] digital temperature sensor is accurate to $2^{\circ}\mathrm{C}$ over a temperature range of -25$^{\circ}\mathrm{C}$ to $+85^{\circ}\mathrm{C}$, and is controlled using a serial peripheral interface (SPI). The temperature measurement is made with a 10-bit resolution Delta-Σ Analog to Digital Converter, which translates to a temperature resolution of $0.25^{\circ}\mathrm{C}$. A block diagram of the TMP125 temperature sensor is shown in Figure 4.3.

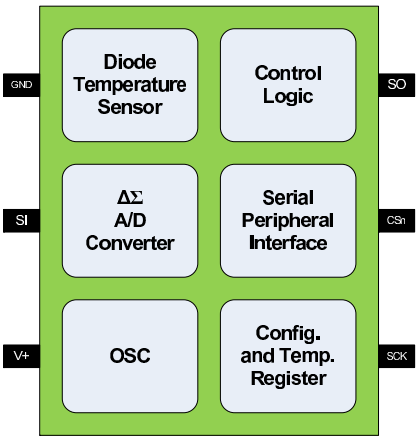


Figure 4.3: $2^{\circ}\mathrm{C}$ Accurate Digital Temperature Sensor with SPI Interface

An application running on the MicroBlaze 32-bit Soft-Core Processor will periodically sample the temperature of all nine sensors to determine if a temperature calibration is required for the various high-speed devices. Upon characterization of the Data Path board during normal operation within a PPG Chassis, a set of temperature limits will be defined that denote when a recalibration is necessary. It is possible that there may be only three temperature zones:

1. Too Cold!!
2. Normal Operation
3. Too Hot!!

However, more zones may be necessary depending on the environment and performance limitations of the high-speed devices.

The temperature register of the TMP125 is a 16-bit, read-only register that stores the output of the most recent conversion. However, temperature is represented by only 10-bits, which are in signed two's complement format. The first bit of the temperature register, D15, is a leading zero. Bits D14 to D5 are used to indicate temperature. Bits D4 to D0 are the same as D5 (see Table 4.41). Data format for temperature is summarized in Table 4.42. When calculating the signed two's complement temperature value, be sure to use only the 10 data bits.

Following power-up or reset, the temperature register will read $0^{\circ}\mathrm{C}$ until the first conversion is complete.

Table 4.41: TMP125 Temperature Register

D15	**D14**	**D13**	**D12**	**D11**	**D10**	**D9**	**D8**	**D7**	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**
0	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0	T0	T0	T0	T0	T0

Table 4.42: TMP125 Temperature Data Format

TMP125 Temperature Data Format	
Temperature ($^{\circ}\mathrm{C}$)	**DIGITAL OUTPUT (T9...T0)**
+127	0b01_1111_1100
+125	0b01_1111_0100
+100	0b01_1001_0000
+75	0b01_0010_1100
+50	0b00_1100_1000
+25	0b00_0110_0100
+10	0b00_0010_1000
+0.25	0b00_0000_0001
0	0b00_0000_0000
-0.25	0b11_1111_1111
-25	0b11_1001_1100
-50	0b11_0011_1000
-55	0b11_0010_0100

4.12.1 TMP_SENSE_1A: TI TMP125 Temperature Sensor SPI Controller

The *TMP_SENSE_1A* EDK Peripheral is an instance of the *xps_spi* IP and is the main interface to the Temperature Sensor #1(A). The *TMP_SENSE_1A* connects to the Spartan-3A FPGA pins shown in Table 4.43.

Table 4.43: **TMP_SENSE_1A EDK Peripheral I/O Descriptions**

xps_spi Pin Name	Pin Name	Direction	Pin	Description
SCK	FPGA_TMPSENS_1A_SCK	O	AB7	Temperature Sensor 1A Serial Clock.
MOSI	FPGA_TMPSENS_1A_MOSI	O	AB8	Temperature Sensor 1A Master-Out-Slave-In (MOSI).
MISO	FPGA_TMPSENS_1A_MISO	I	AB6	Temperature Sensor 1A Master-In-Slave-Out (MISO).
SS	FPGA_TMPSENS_1A_CSN	O	AA8	Temperature Sensor 1A Chip Select (Active Low).

4.12.2 TMP_SENSE_1B: TI TMP125 Temperature Sensor SPI Controller

The *TMP_SENSE_1B* EDK Peripheral is an instance of the *xps_spi* IP and is the main interface to the Temperature Sensor #1(B). The *TMP_SENSE_1B* connects to the Spartan-3A FPGA pins shown in Table 4.44.

Table 4.44: **TMP_SENSE_1B EDK Peripheral I/O Descriptions**

xps_spi Pin Name	Pin Name	Direction	Pin	Description
SCK	FPGA_TMPSENS_1B_SCK	O	F7	Temperature Sensor 1B Serial Clock.
MOSI	FPGA_TMPSENS_1B_MOSI	O	F8	Temperature Sensor 1B Master-Out-Slave-In (MOSI).
MISO	FPGA_TMPSENS_1B_MISO	I	E6	Temperature Sensor 1B Master-In-Slave-Out (MISO).
SS	FPGA_TMPSENS_1B_CSN	O	E7	Temperature Sensor 1B Chip Select (Active Low).

4.12.3 TMP_SENSE_1C: TI TMP125 Temperature Sensor SPI Controller

The *TMP_SENSE_1C* EDK Peripheral is an instance of the *xps_spi* IP and is the main interface to the Temperature Sensor #1(C). The *TMP_SENSE_1C* connects to the Spartan-3A FPGA pins shown in Table 4.45.

Table 4.45: **TMP_SENSE_1C EDK Peripheral I/O Descriptions**

xps_spi Pin Name	Pin Name	Direction	Pin	Description
SCK	FPGA_TMPSENS_1C_SCK	O	V17	Temperature Sensor 1C Serial Clock.
MOSI	FPGA_TMPSENS_1C_MOSI	O	W17	Temperature Sensor 1C Master-Out-Slave-In (MOSI).
MISO	FPGA_TMPSENS_1C_MISO	I	U16	Temperature Sensor 1C Master-In-Slave-Out (MISO).
SS	FPGA_TMPSENS_1C_CSN	O	Y17	Temperature Sensor 1C Chip Select (Active Low).

4.12.4 TMP_SENSE_2A: TI TMP125 Temperature Sensor SPI Controller

The *TMP_SENSE_2A* EDK Peripheral is an instance of the *xps_spi* IP and is the main interface to the Temperature Sensor #2(A). The *TMP_SENSE_2A* connects to the Spartan-3A FPGA pins shown in Table 4.46.

Table 4.46: **TMP_SENSE_2A EDK Peripheral I/O Descriptions**

xps_spi Pin Name	Pin Name	Direction	Pin	Description
SCK	FPGA_TMPSENS_2A_SCK	O	R19	Temperature Sensor 2A Serial Clock.
MOSI	FPGA_TMPSENS_2A_MOSI	O	R20	Temperature Sensor 2A Master-Out-Slave-In (MOSI).
MISO	FPGA_TMPSENS_2A_MISO	I	R18	Temperature Sensor 2A Master-In-Slave-Out (MISO).
SS	FPGA_TMPSENS_2A_CSN	O	R21	Temperature Sensor 2A Chip Select (Active Low).

4.12.5 TMP_SENSE_2B: TI TMP125 Temperature Sensor SPI Controller

The *TMP_SENSE_2B* EDK Peripheral is an instance of the *xps_spi* IP and is the main interface to the Temperature Sensor #2(B). The *TMP_SENSE_2B* connects to the Spartan-3A FPGA pins shown in Table 4.47.

Table 4.47: **TMP_SENSE_2B EDK Peripheral I/O Descriptions**

xps_spi Pin Name	Pin Name	Direction	Pin	Description
SCK	FPGA_TMPSENS_2B_SCK	O	J21	Temperature Sensor 2B Serial Clock.
MOSI	FPGA_TMPSENS_2B_MOSI	O	J22	Temperature Sensor 2B Master-Out-Slave-In (MOSI).
MISO	FPGA_TMPSENS_2B_MISO	I	J20	Temperature Sensor 2B Master-In-Slave-Out (MISO).
SS	FPGA_TMPSENS_2B_CSN	O	K22	Temperature Sensor 2B Chip Select (Active Low).

4.12.6 TMP_SENSE_2C: TI TMP125 Temperature Sensor SPI Controller

The *TMP_SENSE_2C* EDK Peripheral is an instance of the *xps_spi* IP and is the main interface to the Temperature Sensor #2(C). The *TMP_SENSE_2C* connects to the Spartan-3A FPGA pins shown in Table 4.48.

Table 4.48: **TMP_SENSE_2C EDK Peripheral I/O Descriptions**

xps_spi Pin Name	Pin Name	Direction	Pin	Description
SCK	FPGA_TMPSENS_2C_SCK	O	C17	Temperature Sensor 2C Serial Clock.
MOSI	FPGA_TMPSENS_2C_MOSI	O	B17	Temperature Sensor 2C Master-Out-Slave-In (MOSI).
MISO	FPGA_TMPSENS_2C_MISO	I	D17	Temperature Sensor 2C Master-In-Slave-Out (MISO).
SS	FPGA_TMPSENS_2C_CSN	O	A17	Temperature Sensor 2C Chip Select (Active Low).

4.13 DDR2 SDRAM SODIMM I^{2}C Memory Interface

The Measurement Board board contains a DDR2 SDRAM SODIMM connector for use with memory of a capacity up to 2GB. The SODIMM contains an I^{2}C memory for storing information related to the control and refresh of the DDR2 SDRAM.

4.13.1 vcl_iic_ctrlr_2: I^{2}C Control Peripheral

The *vcl_iic_ctrlr_v1_00_a* EDK Peripheral is used to control the DDR2 SDRAM SODIMM I^{2}C Interface, and appears as a set of 16 software accessible registers to the applications running on the MicroBlaze. The *vcl_iic_ctrlr_v1_00_a* EDK Peripheral's register map is shown in Table 4.50.
The *vcl_iic_ctrlr_2* connects to the Spartan-3A FPGA pins shown in Table 4.49.

Table 4.49: **vcl_iic_ctrlr_2 EDK Peripheral I/O Descriptions**

vcl_iic_ctrlr_v1_00_a Pin Name	Pin Name	Dir	Pin	Description
io_fpga_iic_sda	FPGA_DDR2_SDRAM_SDA	IO	B15	DDR2 SDRAM SODIMM I^{2}C Serial Data.
io_fpga_iic_scl	FPGA_DDR2_SDRAM_SCL	O	A15	DDR2 SDRAM SODIMM I^{2}C Serial Clock.

Table 4.50: **vcl_iic_ctrlr_v1_00_a EDK Peripheral Register Map**

Address	Name	R/W	Default Value	Description
0	{16'b0,iic_prer[15:0]}	R/W	0x00000063	I^2C Clock Prescaler Register.
1	{30'b0,iic_ctrl[1:0]}	R/W	0x00000000	[0:15]: reserved; [16:31]: iic_prer[15:0]: I^2C Prescale.. I^2C Control Register.
2	{24'b0,iic_txr[7:1],iic_rnw}	R/W	0x00000000	[0:29]: reserved; [30]: iic_core_en: I^2C Module Enable. (Active High) [31]: iic_int_en: I^2C Interrupt Enable. (Active High) I^2C Transmit Data Register.
3	{26'b0,iic_cmd[5:0]}	R/W	0x00000000	[0:23]: reserved; [24:30]: iic_txr: I^2C Transmit Data. [31]: iic_rnw: I^2C Read/Write Bit, 0 = Write, 1 = Read. I^2C Command Register.
4	{31'b0,iic_rst}	R/W	0x00000000	[0:25]: reserved; [26]: iic_iack: Interrupt Acknowledge. (Active High) When set, clears a pending interrupt. [27]: iic_ack: When a reciever, sent ACK (iic_ack = '0') or NACK (iic_ack = '1'). [28]: iic_wr: Write to a Slave. (Active High) [29]: iic_rd: Read from a Slave. (Active High) [30]: iic_sto: Generate Stop Condition. (Active High) [31]: iic_sta: Generate Start Condition. (Active High) I^2C Reset Register. [0:30]: reserved; [31]: iic_rst: I^2C Module Reset. (Active High). Toggle high then low to reset vcl_iic_ctrlr module.
5	slv_reg5[0:31]	R/W	0x00000000	Reserved Register.
6	slv_reg6[0:31]	R/W	0x00000000	Reserved Register.
7	slv_reg7[0:31]	R/W	0x00000000	Reserved Register.
8	{24'b0,iic_rxr[7:0]}	RO	0x00000000	I^2C Receive Data Register.
9	{26'b0,iic_stat[4:0]}	RO	0x00000000	I^2C Status Register.
A	slv_reg10[0:31]	RO	0x00000000	[0:26]: reserved; [27]: rxack: Received Acknowledge from Slave. 1 = No Acknowdge Received. 0 = Acknowdge Received. [28]: iic_busy: I^2c Busy. 1 after START signal detected. 0 after STOP signal detected. [29]: al: Arbitration Lost. Arbitration is lost when: a STOP signal is detected, but not requested. The master drives SDA high, but SDA is low. [30]: tip: Transfer in Progress. 1 = Transferring Data. 0 = Transfer Complete. [31]: irq_flag: Interrupt Flag. (Active High) This bit is set when an interrupt is pending. Reserved Register.
B	slv_reg11[0:31]	RO	0x00000000	Reserved Register.
C	slv_reg12[0:31]	RO	0x00000000	Reserved Register.
D	slv_reg13[0:31]	RO	0x00000000	Reserved Register.
E	slv_reg14[0:31]	RO	0x00000000	Reserved Register.
F	slv_reg15[0:31]	RO	0x00000000	Reserved Register.

4.14 Debug Peripherals

The Measurement Board contains several peripherals intended to be used during debug of both Hardware and Software.

- 3 Push-Buttons (Active Low)
- 4 LEDs (Active Low)
- Logic Analyzer Header
- CP2102 USB-to-UART Bridge
- FTDI FT245BL USB Interface
- FPGA Re-Program Push-Button

4.14.1 Buttons_3Bit: Push-Button GPIO Controller

The Measurement Board Control FPGA has three momentary push-button for debugging various applications and hardware. The push-buttons can be polled independently. A logic low indicates a push-button has been pressed, and the push-buttons idle high when not pressed. The push-buttons are controlled by the MicroBlaze via a GPIO peripheral.

Figure 4.4: Momentary Push-Button

The push-buttons connect to the Spartan-3A FPGA pins shown in Table 4.51.

Table 4.51: **Buttons_3Bit EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	FPGA_PUSHBUTTONS[0]	I	H9	NA	Push-Button 0 (Active Low).
1	FPGA_PUSHBUTTONS[1]	I	G8	NA	Push-Button 1 (Active Low).
2	FPGA_PUSHBUTTONS[2]	I	G7	NA	Push-Button 2 (Active Low).

4.14.2 LEDs_4Bit: LED GPIO Controller

The Measurement Board Control FPGA has four LEDs for debugging various applications and hardware. The LEDs can be controlled independently, and are turned on by setting the appropriate control bit to a logic low. The LEDs are controlled by the MicroBlaze via a GPIO peripheral.

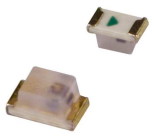


Figure 4.5: Light-Emitting Diode

The LEDs connect to the Spartan-3A FPGA pins shown in Table 4.52.

Table 4.52: **LEDs_4Bit EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	FPGA_LEDS[0]	O	D8	1	LED Bit 0 (Active Low).
1	FPGA_LEDS[1]	O	C7	0	LED Bit 1 (Active Low).
2	FPGA_LEDS[2]	O	C8	1	LED Bit 2 (Active Low).
3	FPGA_LEDS[3]	O	B8	0	LED Bit 3 (Active Low).

4.14.3 Logic Analyzer

The Measurement Board contains a header for probing internal FPGA signals with a Logic Analyzer. The Logic Analyzer header is a 18-pin header with the following pinout:

Table 4.53: Logic Analyzer Header Pinout

Odd Pins		Even Pins	
FPGA_LA_CLK	1	2	GND
FPGA_LA_DATA[7]	3	4	GND
FPGA_LA_DATA[6]	5	6	GND
FPGA_LA_DATA[5]	7	8	GND
FPGA_LA_DATA[4]	9	10	GND
FPGA_LA_DATA[3]	11	12	GND
FPGA_LA_DATA[2]	13	14	GND
FPGA_LA_DATA[1]	15	16	GND
FPGA_LA_DATA[0]	17	18	GND

The LEDs connect to the Spartan-3A FPGA pins shown in Table 4.54.

Table 4.54: **Logic Analyzer I/O Descriptions**

Pin Name	Dir	Pin	Default Value	Description
FPGA_LA_CLK	O	A14	NA	Logic Analyzer Clock.
FPGA_LA_DATA[7]	O	A13	NA	Logic Analyzer Data Bit 7.
FPGA_LA_DATA[6]	O	B13	NA	Logic Analyzer Data Bit 6.
FPGA_LA_DATA[5]	O	C13	NA	Logic Analyzer Data Bit 5.
FPGA_LA_DATA[4]	O	D15	NA	Logic Analyzer Data Bit 4.
FPGA_LA_DATA[3]	O	D13	NA	Logic Analyzer Data Bit 3.
FPGA_LA_DATA[2]	O	E13	NA	Logic Analyzer Data Bit 2.
FPGA_LA_DATA[1]	O	E14	NA	Logic Analyzer Data Bit 1.
FPGA_LA_DATA[0]	O	F13	NA	Logic Analyzer Data Bit 0.

4.14.4 CP2102 USB-to-UART Bridge

The Measurement Board contains a Silicon Laboratories CP2102 USB-to-UART Bridge for controlling the board in a development environment in lieu of the User Interface board. The CP2102 provides a COM Port Interface over USB, and allwos for an RS-232 interface between the Data Path FPGA and the CP2102. The Microblaze application will treat the CP2102 as an alternate data source similar to the User Interface board. This will allow for a common interface to debug the Measurement Board, because the same message decoding application will be used to intercept and decode incoming messages. The Microblaze design will use an EDK XPS UART Lite peripheral, which will be configured as follows:

C_BAUDRATE	115200
C_DATA_BITS	8
C_USE_PARITY	0
C_ODD_PARITY	0

The desired terminal settings would then be:

Bits per second:	115200
Data bits:	8
Parity:	None
Stop bits:	1
Flow Control:	None

The CP2102 connects to the Spartan-3A FPGA pins shown in Table 4.56.

Table 4.55: CP2102 I/O Descriptions

CP2102 Pin Name	FPGA Pin Name	Pin	Description
TXD	FPGA_RS232_RX	AB5	RS-232 Receive Data to FPGA.
RXD	FPGA_RS232_TX	Y5	RS-232 Transmit Data from FPGA.
RTS	FPGA_RS232_RTS	AA6	RS-232 Request to Send to FPGA.
CTS	FPGA_RS232_CTS	Y6	RS-232 Clear to Send from FPGA.

In order to use the CP2102, you may need it install a driver on your computer. The driver can be downloaded from the following Silicon Laboratories website:

- https://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx

4.14.4.1 RS232_CP2102: RS-232 UartLite Controller

The *RS232_CP2102* EDK Peripheral is an instance of the *xps_uartlite* and connects to the Spartan-3A FPGA pins shown in Table 4.56.

Table 4.56: **RS232_CP2102 EDK Peripheral I/O Descriptions**

xps_uartlite Pin #	FPGA Pin Name	Dir	Pin	Description
RX	FPGA_RS232_RX	I	AB6	RS-232 Receive Data to FPGA.
TX	FPGA_RS232_TX	O	Y5	RS-232 Transmit Data from FPGA.

4.14.4.2 CP2102_RTS: RS-232 UartLite Controller

The *CP2102_RTS* EDK Peripheral connects to the Spartan-3A FPGA pins shown in Table 4.57.

Table 4.57: **CP2102_RTS EDK Peripheral I/O Descriptions**

xps_gpio Pin #	FPGA Pin Name	Dir	Pin	Default Value	Description
0	FPGA_RS232_RTS	I	AA6	NA	RS-232 Request to Send to FPGA.

4.14.4.3 CP2102_CTS: RS-232 UartLite Controller

The *CP2102_CTS* EDK Peripheral connects to the Spartan-3A FPGA pins shown in Table 4.58.

Table 4.58: **CP2102_CTS EDK Peripheral I/O Descriptions**

xps_gpio Pin #	FPGA Pin Name	Dir	Pin	Default Value	Description
0	FPGA_RS232_CTS	O	Y6	0	RS-232 Clear to Send from FPGA.

4.14.5 FTDI FT245BL USB Interface

The Measurement Board Control FPGA has access to a Future Technology Devices International Ltd. (a.k.a., FTDI Chip) FT245BL USB FIFO IC. This USB FIFO can provide higher data throughput than the Silicon Laboratories CP2102 USB-to-UART Bridge IC, and may be used as the main communications interface for the Measurement Instrument if there are problems with either acquiring or using the User Interface Board.

In order to use the FT245BL USB FIFO IC, you will need to install a driver on your computer. FTDI Chip provides two kinds of drivers:

- **Virtual COM Port Drivers**
  http://www.ftdichip.com/Drivers/VCP.htm
- **D2XX Drivers**
  http://www.ftdichip.com/Drivers/D2XX.htm

For higher throughput the D2XX Drivers should be used. If the Virtual COM Port driver is used, then similar data throughput to the CP2102 will be achieved.

The FT245BL USB FIFO IC connects to the Spartan-3A FPGA pins shown in Table 4.59.

Table 4.59: **FT245BL_GPIO EDK Peripheral I/O Descriptions**

xps_gpio Pin #	Pin Name	Dir	Pin	Default Value	Description
0	FPGA_FTDI_DATA[0]	I/O	V15	1	FTDI Data Bit 0.
1	FPGA_FTDI_DATA[1]	I/O	V16	0	FTDI Data Bit 1.
2	FPGA_FTDI_DATA[2]	I/O	W15	1	FTDI Data Bit 2.
3	FPGA_FTDI_DATA[3]	I/O	W16	0	FTDI Data Bit 3.
4	FPGA_FTDI_DATA[4]	I/O	Y15	1	FTDI Data Bit 4.
5	FPGA_FTDI_DATA[5]	I/O	Y16	0	FTDI Data Bit 5.
6	FPGA_FTDI_DATA[6]	I/O	AB15	1	FTDI Data Bit 6.
7	FPGA_FTDI_DATA[7]	I/O	AB16	0	FTDI Data Bit 7.
8	FPGA_FTDI_TXEN	I	P12	0	FTDI TX Data Enable.
9	FPGA_FTDI_RXFN	I	R12	0	FTDI RX Data Valid.
10	FPGA_FTDI_PWRENN	I	R13	0	FTDI Power Enable.
11	FPGA_FTDI_RSTOUTN	I	R14	0	FTDI Reset Output.
12	FPGA_FTDI_RDN	O	U13	0	FTDI Read Enable.
13	FPGA_FTDI_WRN	O	V14	0	FTDI Write Enable.
14	FPGA_FTDI_SI_WU	O	W13	0	FTDI Send Immediate / Wake Up Signal.

4.14.6 FPGA Re-Program Push-Button

The Measurement Board has the ability to precisely control the re-configuration process for the Control FPGA. Figure 4.6 shows the circuit used on the Measurement Board. Subsection **??** describes how the TPS3823-33DBV device works at power-up along with the manual reset input. Unlike the Board Reset circuit whose Logical AND gate *A* input is driven by *Config Done*, the PROG_B circuit's *A* input is driven by the User Interface board. This input provides the User Interface Board with the capability to initiate an FPGA re-configuration during either the Measurement Instrument power-up routine or a firmware upgrade.

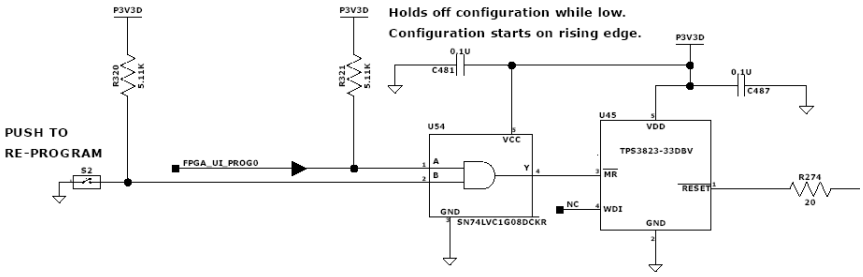


Figure 4.6: Measurement Board PROG_B Circuit

4.14.7 Reach Technologies Display

The Measurement Board contains an RS-232 interface to a Reach Technologies Display for displaying waveforms and configuring the instrument. The Microblaze design will use an EDK XPS UART Lite peripheral, which will be configured as follows:

C_BAUDRATE	115200
C_DATA_BITS	8
C_USE_PARITY	0
C_ODD_PARITY	0

The desired terminal settings would then be:

Bits per second:	115200
Data bits:	8
Parity:	None
Stop bits:	1
Flow Control:	None

The Reach Technologies Display connects to the Spartan-3A FPGA pins shown in Table 4.60.

Table 4.60: CP2102 I/O Descriptions

CP2102 Pin Name	FPGA Pin Name	Pin	Description
TXD	FPGA_RS232_RX	V12	RS-232 Receive Data to FPGA.
RXD	FPGA_RS232_TX	U12	RS-232 Transmit Data from FPGA.

4.15 Board Reset Push-Button

There are several methods to reset the Measurement Board. A Texas Instruments Processor Supervisory Circuit (TSP3823-33DBV) is used to provide both a power-up and manual reset. During power-on, $\overline{RESET}$ is asserted when supply voltage +3.3V becomes higher than 1.1V. Thereafter, the supply voltage supervisor monitors +3.3V and keeps $\overline{RESET}$ active as long as +3.3V remains below the threshold V_{IT}. An internal timer delays the return of the output to the inactive state (high) to ensure proper system reset. The delay time, t_d, starts after +3.3V has risen above the threshold voltage V_{IT}. When the supply voltage drops below the threshold voltage V_{IT}, the output becomes active (low) again. The threshold voltage of the TPS3823-33DBV is 2.93V.

The TPS3823-33DBV device incorporates a manual reset input, $\overline{MR}$. A low level at $\overline{MR}$ causes $\overline{RESET}$ to become active. The Measurement board does not take advantage of the Watch Dog Circuit available in the TPS3823-33DBV. A truth table for the TPS3823-33DBV is shown in Table 4.61.

Table 4.61: TPS3823-33DBV Truth Table

TPS3823-33DBV Truth Table		
INPUTS		**OUTPUTS**
$\overline{MR}$	$\mathbf{V}_{DD} > \mathbf{V}_{IT}$	$\overline{RESET}$
0	0	0
0	1	0
1	0	0
1	1	1

As +3.3V powers-on the TPS3823 will initiate a power-on reset to the Control FPGA, but the FPGA will most likely be busy in the configuration process. Therefore, the reset will be missed by the FPGA. However, an external logic AND gate is provided to logically *AND* an active-low push-button and the FPGA Configuration Done (Active High) signal (see Table 4.62), so that the $\overline{MR}$ pin is toggled low then high. This results in $\overline{RESET}$ being toggled low for 1 μs, which will provide plenty of time for the FPGA to properly reset after the configuration process completes. After the board has powered up and the FPGA is configured, the user can press the push-button to initiate an FPGA reset whenever desired.

Table 4.62: Local Reset Truth Table

Local Reset Truth Table		
Config Done	**Push-Button**	**Local Reset**
0	0	0
0	1	0
1	0	0
1	1	1

During normal operation, the User Interface board can also initiate a board reset by toggling the UI Reset pin low then high. The truth table for the Board Reset is shown in Table 4.63.

Table 4.63: Board Reset Truth Table

Board Reset Truth Table		
Local Reset	**UI Reset**	**Board Reset**
0	0	0
0	1	0
1	0	0
1	1	1

The Board Reset signal is fanned out to 3 devices on the Measurement Board:

- Control FPGA
- Power Control CPLD
- Data Path FPGA

The circuit shown in Figure 4.7 allows for all three programmable devices to be simultaneously reset.

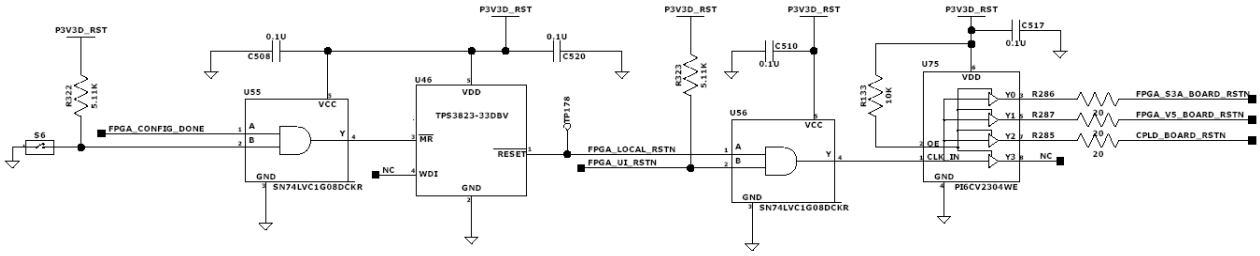


Figure 4.7: Measurement Board Reset Circuit

The reset signals connect to the Spartan-3A FPGA pins shown in Table 4.64.

Table 4.64: Reset Signal Descriptions

Pin Name	Dir	Pin	Description
FPGA_S3A_BOARD_RSTN	I	D7	Measurement Board Main Reset (Active Low).
FPGA_LOCAL_RSTN	I	D6	Measurement Board Local Reset (Active Low).

4.15.1 proc_sys_reset_0: Processor System Reset Controller

The *proc_sys_reset* EDK Peripheral is in charge of receiving all reset and DCM locked signals in order to determine when the processor reset should be driven.

```
BEGIN proc_sys_reset
 PARAMETER INSTANCE = proc_sys_reset_0
 PARAMETER HW_VER = 2.00.a
 PARAMETER C_EXT_RESET_HIGH = 0
 PORT Slowest_sync_clk = sys_clk_s
 PORT Dcm_locked = dcm0_locked
 PORT Ext_Reset_In = sys_rst_s
 PORT MB_Reset = mb_reset
 PORT Bus_Struct_Reset = sys_bus_reset
 PORT MB_Debug_Sys_Rst = Debug_SYS_Rst
END
```

The *proc_sys_reset* EDK Peripheral connects to the Spartan-3A FPGA pins shown in Table 4.65.

Table 4.65: **proc_sys_reset_0 EDK Peripheral I/O Descriptions**

Peripheral Name	Pin Name	Dir	Pin	Default Value	Description
Ext_Reset_In	FPGA_S3A_BOARD_RSTN	I	D7	1	Board Main Reset (Active Low).

4.16 CLK_100MHZ_INPUT: Differential Clock Input Buffer

The *CLK_100MHZ_INPUT* EDK Peripheral is an instance of the *util_ds_buf* IP and connects to the Spartan-3A FPGA pins shown in Table 4.66. The *util_ds_buf* IP is essentially an *IBUFGDS* Spartan-3A input clock buffer primitive.

Table 4.66: **CLK_100MHZ_INPUT EDK Peripheral I/O Descriptions**

util_ds_buf Pin #	FPGA Pin Name	Dir	Pin	Description
IBUF_DS_P	FPGA_CLK100MHZ_P	I	AA12	100MHz Clock (Positive Differential).
IBUF_DS_P	FPGA_CLK100MHZ_N	I	AB12	100MHz Clock (Negative Differential).
IBUF_OUT	dcm_clk_s	O	NA	100MHz Clock.

```
BEGIN util_ds_buf
 PARAMETER INSTANCE = CLK_100MHZ_INPUT
 PARAMETER HW_VER = 1.00.a
 PARAMETER C_BUF_TYPE = IBUFGDS
 PORT IBUF_DS_P = fpga_0_CLK_100_P
 PORT IBUF_DS_N = fpga_0_CLK_100_N
 PORT IBUF_OUT = dcm_clk_s
END
```

4.17 dcm_module_0: Digital Clock Module

The *dcm_module_0* EDK Peripheral is an instance of the *dcm_module* IP and is used to generate the clocks shown in Table 4.67. The *util_ds_buf* IP is essentially a Spartan-3A Digital Clock Manager (*DCM*).

Table 4.67: **dcm_module_0 EDK Peripheral I/O Descriptions**

util_ds_buf Pin #	FPGA Pin Name	Dir	Description
CLKIN	dcm_clk_s	I	100MHz Clock.
CLK0	DDR_SDRAM_mpmc_clk_s	O	DDR SDRAM 100MHz Clock, 0 degrees.
CLK90	DDR_SDRAM_mpmc_clk_90_s	O	DDR SDRAM 100MHz Clock, 90 degrees.
CLKDV	sys_clk_s	O	MicroBlaze 50MHz System Clock.
CLKFB	DDR_SDRAM_mpmc_clk_s	O	DCM Feedback CLock Clock.
LOCKED	dcm0_locked	O	DCM Locked (Active High).

```
BEGIN dcm_module
 PARAMETER INSTANCE = dcm_module_0
 PARAMETER HW_VER = 1.00.c
 PARAMETER C_CLKDV_DIVIDE = 2.0
 PARAMETER C_CLKIN_PERIOD = 10.0
 PARAMETER C_EXT_RESET_HIGH = 0
 PARAMETER C_CLKIN_BUF = FALSE
 PARAMETER C_CLKFB_BUF = FALSE
 PARAMETER C_CLK0_BUF = TRUE
 PARAMETER C_CLK90_BUF = TRUE
 PARAMETER C_CLKDV_BUF = TRUE
 PORT RST = net_vcc
 PORT CLKIN = dcm_clk_s
 PORT CLK0 = DDR_SDRAM_mpmc_clk_s
 PORT CLK90 = DDR_SDRAM_mpmc_clk_90_s
 PORT CLKFB = DDR_SDRAM_mpmc_clk_s
 PORT CLKDV = sys_clk_s
 PORT LOCKED = dcm0_locked
END
```

4.18 INT_CLK_10MHZ_INPUT: Differential Internal 10MHz Clock Input Buffer

The *INT_CLK_10MHZ_INPUT* EDK Peripheral is an instance of the *util_ds_buf* IP and connects to the Spartan-3A FPGA pins shown in Table 4.69. The *util_ds_buf* IP is essentially an *IBUFGDS* Spartan-3A input clock buffer primitive.

Table 4.68: **CLK_100MHZ_INPUT EDK Peripheral I/O Descriptions**

util_ds_buf Pin #	FPGA Pin Name	Dir	Pin	Description
IBUF_DS_P	FPGA_INT_CLK10MHZ_REF_P	I	A12	Internal 10MHz Reference Clock (Positive Differential).
IBUF_DS_P	FPGA_INT_CLK10MHZ_REF_N	I	A11	Internal 10MHz Reference Clock (Negative Differential).
IBUF_OUT	int_clk10mhz_ref	O	NA	Internal 10MHz Reference Clock.

```
BEGIN util_ds_buf
 PARAMETER INSTANCE = INT_CLK_10MHZ_INPUT
 PARAMETER HW_VER = 1.00.a
 PARAMETER C_BUF_TYPE = IBUFGDS
 PORT IBUF_DS_P = FPGA_INT_CLK10MHZ_REF_P
 PORT IBUF_DS_N = FPGA_INT_CLK10MHZ_REF_N
 PORT IBUF_OUT = int_clk10mhz_ref
END
```

4.19 EXT_CLK_10MHZ_INPUT: Differential External 10MHz Clock Input Buffer

The *EXT_CLK_10MHZ_INPUT* EDK Peripheral is an instance of the *util_ds_buf* IP and connects to the Spartan-3A FPGA pins shown in Table 4.69. The *util_ds_buf* IP is essentially an *IBUFGDS* Spartan-3A input clock buffer primitive.

Table 4.69: **CLK_100MHZ_INPUT EDK Peripheral I/O Descriptions**

util_ds_buf Pin #	FPGA Pin Name	Dir	Pin	Description
IBUF_DS_P	FPGA_EXT_CLK10MHZ_REF_P	I	A12	External 10MHz Reference Clock (Positive Differential).
IBUF_DS_P	FPGA_EXT_CLK10MHZ_REF_N	I	A11	External 10MHz Reference Clock (Negative Differential).
IBUF_OUT	ext_clk10mhz_ref	O	NA	External 10MHz Reference Clock.

```
BEGIN util_ds_buf
 PARAMETER INSTANCE = EXT_CLK_10MHZ_INPUT
 PARAMETER HW_VER = 1.00.a
 PARAMETER C_BUF_TYPE = IBUFGDS
 PORT IBUF_DS_P = FPGA_EXT_CLK10MHZ_REF_P
 PORT IBUF_DS_N = FPGA_EXT_CLK10MHZ_REF_N
 PORT IBUF_OUT = ext_clk10mhz_ref
END
```

5 MicroBlaze Design Information

5.1 Initialization Routine

The Control FPGA located on the Measurement Board will need to perform the following system initialization in the order shown:

1. Setup the two instrument fan controllers.
2. Initialize the Analog Devices AD9516 Clock PLL.
3. Configure the Data Path FPGA.
4. Configure the two AsAP Version 2 DSP Processors.
5. Setup the Texas Instruments DAC5682Z 1 GS/s DAC.
6. Perform an FPGA reset on the Data Path FPGA.

6 Debug Commands

6.1 Instrument Sub-System Control

6.1.1 Firmware Revision

The **get_fwrev** command is used to get the Firmware Revision.

- **get_fwrev [mode]**

The **get_fwrev** command will retreive the Firmware Revision from the application and display the revision on the terminal.

6.1.2 Install Firmware Image

The **load_image** command is used to install either a factory or upgrade image.

- **load_image [u/f] [file]**

The **load_image** command will set the install mode; either factory (f) or upgrade (u), and provide the path to the firmware image. Once the file has been transfered to the SPI Flash PROM, then the application will issue a factory reboot to the appropriate location via ICAP.

6.1.3 Program Data Path FPGA

The **dp_prog** command is used to program the Data Path FPGA.

- **dp_prog [f/n]**

The **dp_prog** command will check the DONE pin of the Data Path FPGA if the "n" flag is set, and if it is low it will program the Data Path FPGA. However, if the DONE pin is high, then the Data Path will not be programmed and a message will be displayed on the terminal indicating the status. If the "f" flag is set, then the Data Path FPGA will be forced to re-program using the file "DPIMAGE.BIN" stored on the microSD card.

6.1.4 Help

The **help** command is used to provide a list of terminal debug commands.

- **help [cmd]**

The **help** command will retreive the terminal debug commands, and if a [cmd] argument is provided display detailed command instructions where necessary.

6.2 Instrument Sub-System Analog Control

6.2.1 Temperature Status

The **get_temp** command is used to query one of six temperature sensors for their respective temperature measurements.

- **get_temp [temp]**

The **get_temp** command will return a temperature reading in degress Celsius. For more information see Section 4.12.

6.2.2 AD9516 Clock Generator Control

The **ad9516** command is used to set or read the internal registers of the AD9516 Clock Generator.

- **ad9516 [r/w] [addr] [data]**

The **ad9516** command will receive a read/write bit, a 13-bit address and an 8-bit data value. The application will send the address and data in 3-bytes to the AD9516, and immediately after will write a 0x1 to address 0x232 to update the registers within the AD9516. The r/w bit of the command will be used to determine if data needs to be read from the AD9516. The application will also print out address and data written to or read from the AD9516. For example,

```
AD9516 Write: 0x0A written to address 0x101

                -or-

AD9516 Read: 0xA0 read from address 0x1A1
```

6.2.3 AMC6821 Fan Control

The **fan** command is used to set or read the internal registers of the AMC6821 Fan Controller(s).

- **fan [r/w] [addr] [data] [chan]**

The **fan** command will receive a read/write bit, an 8-bit address, an 8-bit data value, and a channel identifier. The application will send the address and data in 2-bytes to the desired AMC6821 fan controller (i.e., 0 or 1). The r/w bit of the command will be used to determine if data needs to be read from the AMC6821 fan controller. The application will also print out address and data written to or read from the AMC6821. For example,

```
AMC6821 Write: 0x0A written to address 0x01

                -or-

AMC6821 Read: 0xA0 read from address 0xA1
```

6.2.4 AMC6821 Fan Status

The **fan_stat** command is used to read the status of the AMC6821 Fan Controller(s).

- **fan_stat [chan]**

The **fan_stat** command will receive a channel identifier for the AMC6821 whose status will be displayed. The application will sample the 4 status bits of the AMC6821 connected to the FPGA and print the results to the terminal. For example,

```
AMC6821 Ch 0 Status: 0xA
FAULTN:      1
THERMN:      0
OVRN:        1
SMBALERTN:   0
```

6.2.5 DAC5682 High-Speed DAC Control

The **dac5682** command is used to set or read the internal registers of the DAC5682 High-Speed DAC.

- **dac5682 [r/w] [addr] [data]**

The **dac5682** command will receive a read/write bit, an 5-bit address and an 8-bit data value. The application will send the address and data in 2-bytes to the DAC5682. The r/w bit of the command will be used to determine if data needs to be read from the DAC5682. The application will also print out address and data written to or read from the DAC5682. For example,

```
DAC5682 Write: 0x0A written to address 0x01

                    -or-

DAC5682 Read: 0xA0 read from address 0xA1
```

6.2.6 10MHz Clock Reference Control

The **ref_clk** command is used to read the status of the AMC6821 Fan Controller(s).

- **ref_clk [ctrl]**

The **ref_clk** command will receive a 4-bit control word to set the desired status of the 10MHz reference. The application will write the 4 bits of the 10MHz reference control pins connected to the FPGA and print the results to the terminal. For example,

```
10MHz Reference Write: 0xA
```

6.2.7 External 10MHz Clock Reference Status

The **ext_ref** command is used to read the status of the AMC6821 Fan Controller(s).

- **ext_ref**

The **ext_ref** command will receive a 4-bit control word to set the desired status of the 10MHz reference. The application will sample the 1 status bit of the External 10MHz reference clock connected to the FPGA and print the results to the terminal. If the loss-of-signal (LOS) bit is a logic high, then the external reference clock is not present. For example,

```
External 10MHz Reference Not Present

                    -or-

External 10MHz Reference Present
```

6.3 Instrument Sub-System AsAP Control

6.3.1 AsAP Configuration

The **asap_cfg** command is used to set or read the internal registers of the AD9516 Clock Generator.

- **asap_cfg [chan] [file]**

The **asap_cfg** command will receive a channel selection bit, and the path to an AsAP configuration file. The application will configure the desired AsAP DSP with the provided configuration file. The application will also print out the status of the AsAP configuration. For example,

```
AsAP Ch 0: Configuration Started
AsAP Ch 0: . . . . . .
AsAP Ch 0: Configuration Completed

                -or-

AsAP Ch 1: Configuration Started
AsAP Ch 1: . . . . . .
AsAP Ch 1: Configuration Completed
```

6.4 Instrument Sub-System Data Path Control

6.4.1 Data Path FPGA Control

The **dp** command is used to set or read the internal registers of the Data Path FPGA.

- **dp [r/w] [addr] [data]**

The **dp** command will receive a read/write bit, an 16-bit address and an 32-bit data value. The application will send the address and data to the Data Path FPGA. The r/w bit of the command will be used to determine if data needs to be read from the Data Path FPGA. The application will also print out address and data written to or read from the DAC5682. For example,

```
DP Read: 0x0000000F from 0x000E

                -or-

DP Write: 0x00000048 to 0x000E
```

6.4.2 Spectrum Analyzer Control

The **sample** command is used to initiate an FFT of the input ADC samples.

- **sample [num_samples]**

The **sample** command will receive the number of desired samples to capture in SRAM. The application will send the number of samples (a 21-bit value) to the Data Path FPGA. The application will also print out desired number of samples in decimal and hexadecimal. For example,

```
Spectrum Analyzer Samples: 15 (0x00000F)
```

6.4.3 Spectrum Analyzer FFT Control

The **fft** command is used to initiate an FFT of the input ADC samples.

- **fft [on/off]**

The **fft** command will initiate or end the FFT operation for the Spectrum Analyzer. The application will set the FFT on/off bit high in the Data Path FPGA to begin FFT processing, or set the FFT on/off bit low to end the FFT processing. The application will also print out desired FFT status. For example,

```
Spectrum Analyzer FFT: On

                -or-

Spectrum Analyzer FFT: Off
```

6.4.4 Decimation Low-Pass FIR Filter Coefficient Load Control

The **dlpfir_load** command is used to load the FIR filter coefficients.

- **dlpfir_load [file]**

The **dlpfir_load** command will receive the file path to the Decimation Low-Pass FIR Filter Coefficient file. The application will send the 16-bit coefficient values one-by-one to the Data Path FPGA. The coefficient file will contain 200 16-bit coefficients. The application will also print out the total number of words being sent to the Data Path FPGA. For example,

```
Sending 200 coefficients.
```

6.4.5 Spectrum Analyzer Get Samples

The **get_wave** command is used to retrieve the Spectrum Analyzer FFT results.

- **get_wave [num_samples] [file]**

The **get_wave** command will receive the number of samples to retrieve and the file path to the destination file for the FFT results. The application will retrieve eight 16-bit samples from SRAM and store the data into the destination file. The destination file will contain up to 2^{21} 16-bit values. The application will also print out the status of data retrieval from the Data Path FPGA. For example,

```
Retrieving Spectrum Analyzer Samples: 16 (0x000010)
```

6.4.6 Signal Source Output Control

The **src** command is used to start/stop the Signal Source waveform generation.

- **src [on/off]**

The **src** command will start or stop the Signal Source waveform generation. The application will set the Signal Source on/off bit high in the Data Path FPGA to begin waveform generation, or set the Signal Source on/off bit low to end the waveform generation. The application will also print out the current waveform type being generated or when waveform generation is stopped. For example,

```
Signal Source: Sinusoid, 200MHz, On

                -or-

Signal Source: Sinusoid, 200MHz, Off
```

6.4.7 Interpolation Low-Pass FIR Filter Coefficient Load Control

The **ilpfir_load** command is used to load the FIR filter coefficients.

- **ilpfir_load [file]**

The **ilpfir_load** command will receive the file path to the Interpolation Low-Pass FIR Filter Coefficient file. The application will send the 16-bit coefficient values one-by-one to the Data Path FPGA. The coefficient file will contain 200 16-bit coefficients. The application will also print out the total number of words being sent to the Data Path FPGA. For example,

```
Sending 200 coefficients.
```

6.4.8 Signal Source Waveform Load Control

The **src_load** command is used to load the Signal Source Waveform from a file.

- **src_load [file]**

The **src_load** command will receive the file path to the Signal Source waveform file. The application will send the 32-bit waveform values in groups of four to the Data Path FPGA. The coefficient file will contain up to 2^{20} values. The application will also print out the waveform description from the file header. For example,

```
Loading Waveform: Sinusoid, 200MHz
```

6.4.9 Auxiliary Input Mode Control

The **set_auxin** command is used to set the Auxiliary Input Mode.

- **set_auxin [mode]**

The **set_auxin** command will set the Auxiliary Input Mode register in the Data Path FPGA.

6.4.10 Trigger Input Mode Control

The **set_trigin** command is used to set the Trigger Input Mode.

- **set_trigin [mode]**

The **set_trigin** command will set the Trigger Input Mode register in the Data Path FPGA.

6.4.11 Trigger Output Mode Control

The **set_trigout** command is used to set the Trigger Output Mode.

- **set_trigout [mode]**

The **set_trigout** command will set the Trigger Output Mode register in the Data Path FPGA.

7 Measurement Board Firmware Field Upgrade Process

One of the goals of the Measurement Board project is to have the capability of upgrading all programmable parts in the field. This includes the following:

1. Xilinx MicroBlaze Soft-Core Embedded Micro-Controller Code residing in a Spartan-3A (Control FPGA)
2. Xilinx Spartan-3A FPGA Code (Control FPGA)
3. Xilinx Virtex-5 SX50T FPGA Code (Data Path FPGA)

The FPGA devices will be upgraded via the User Interface Board SPI Port interface in conjunction with the Xilinx MicroBlaze Processor. The high-level process is:

The user puts the Measurement Board into the Field Upgrade Mode by connecting a USB cable to the B-type USB connector on the rear panel and initiating the upgrade via a PC upgrade application. The connected PC will contain a new firmware release, and enable the upgrade process. Alternatively, there may be a method of enabling the update process in a front panel display menu called the "**system setup menu**". The User Interface Board's Atmel AVR 8-bit Micro-Controller will download the new firmware release to its local memory, and then decompress the file into individual configuration files:

1. Atmel AVR 8-bit Micro-Controller configuration data
2. Measurement board configuration data

Upon completion of the Atmel AVR 8-bit Micro-Controller update, it will then start sending data to the Xilinx MicroBlaze processor in the Spartan-3A FPGA on the Measurement board. The configuration data is then re-formatted and sent to the external 2GB microSD card residing on the Measurement Board. The new FPGA programming files will be read out of the microSD card and then written into the normal operation sector of the SPI Configuration Flash PROM. Upon completion of the data transfer the check-sum will be calculated by the MicroBlaze processor to verify that the programming file was successfully stored in memory. The MicroBlaze will then initiate an FPGA re-programming by toggling the program bit low on the FPGA configuration bus. If an error exists during re-configuration, the Atmel AVR 8-bit Micro-Controller will either re-transfer the programming file to the MicroBlaze or pulse the PROG_B pin of the FPGAs. The Atmel AVR 8-bit Micro-Controller will verify that the FPGA configuration was successful by monitoring the FPGA Done and INIT_B pins. If the Done pin goes high, then the Atmel AVR 8-bit Micro-Controller will read the FPGA version register. The successful read will indicate that there were no configuration errors. If the Done pin is not driven high, and the INIT_B pin is low then the Atmel AVR 8-bit Micro-Controller will re-initiate the FPGA configuration process by strobing the PROG_B pin of the FPGA.

7.1 FPGA Configuration Options

The Measurement Board has 3 Xilinx FPGAs that need to be configured either at power-up or during a field upgrade. Subsections 7.1.1, 7.1.2, and 7.1.3 describe the three possible configuration options we could employ.

7.1.1 Option #1: SPI Serial Daisy Chain

The Measurement Board has an SPI serial daisy chain that consists of 3 Xilinx FPGAs, and 1 SPI Configuration Flash PROM. Figure 7.1 shows a block diagram of the SPI serial daisy chain. The Xilinx FPGAs have a fixed number of configuration bits that need to be transferred from the SPI Configuration Flash PROM at power up. Table 7.1 shows the required number of configuration bits for the Xilinx FPGAs used on the Measurement Board Measurement Board. The total number of configuration bits required by the SPI serial daisy chain is 31,356,224 bits. In addition to the configuration bits, the MicroBlaze assembled application data will also be a part of the data stored in the SPI Configuration Flash PROM. At the moment, it is unknown how many bits the MicroBlaze assembled application data will require.

The current plan is to generate an SPI Configuration Flash PROM file (.MCS) for each complete Measurement Board image. These images will be stored in a single SPI Configuration Flash PROM. Since the total SPI serial daisy chain will require at least 31,356,224 bits, and we would like to store 2 operational images and one bootloader image, we either need an SPI Configuration Flash PROM with at least 128MB of storage capacity or we need to use the microSD card to store the images for the data path FPGA.

Table 7.1: Xilinx FPGA Bitstream Length

Xilinx FPGA Family	**FPGA Part Number**	**Number of Configuration Bits**
Spartan-3A/3AN FPGA	XC3S1400A	4,755,296
Virtex-5	XC5VSX50T	21,845,632

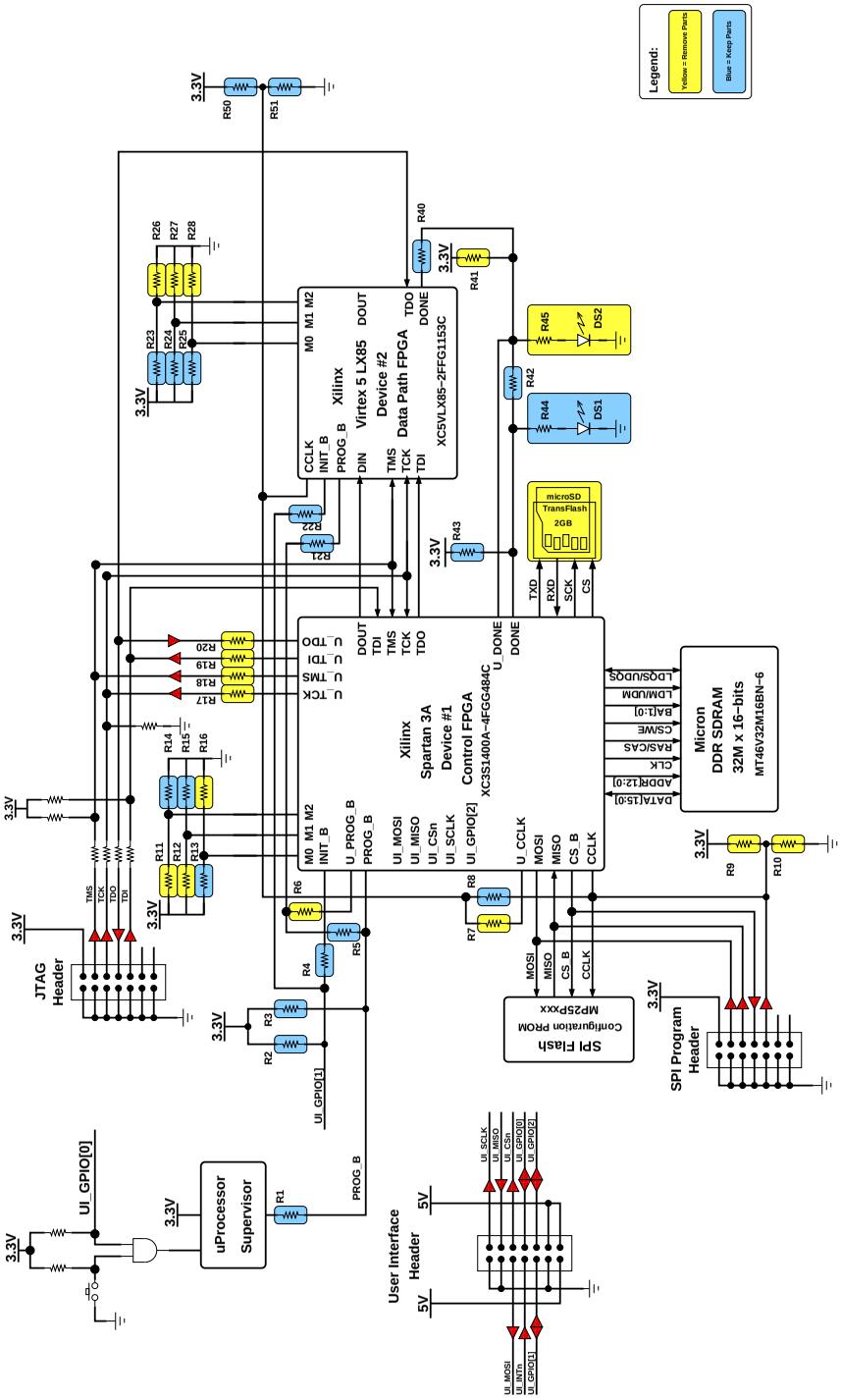


Figure 7.1: SPI Configuration of FPGA Chain Block Diagram

7.1.2 Option #2: SPI Configuration and Slave Serial Daisy Chain

The Measurement Board Measurement Board has 3 Xilinx FPGAs and 1 SPI Configuration Flash PROM that need to be configured during power-up and a field upgrade. The main Control FPGA (Spartan 3A 1400A) is connected directly to the SPI Configuration Flash PROM, where it's configuration and application data reside. At power-up the SPI Configuration Flash PROM will download the bootloader image into the Control FPGA, which contains a MicroBlaze 32-bit microcontroller. Once the Control FPGA is configured, it will transfer the application code from the SPI Configuration Flash PROM to the 512Mb DDR SDRAM, and initiate a reprogramming of itself from the sector containing the operational image. After the Control FPGA is up and running the MicroBlaze application will grab an FPGA configuration file from the 2GB microSD card, and initiate a configuration of the remaining 2 FPGAs using the slave serial process. As shown in Figure 7.2, the Control FPGA will drive the *PROG_B* pin of the slave serial chain, and then send data to the *DIN* pin of the 1^{st} device in the slave serial chain. During this process, the Control FPGA is monitoring the both the *DONE* and *INIT_B* pin of the slave serial chain. Once the *DONE* pin is driven high; signaling a successful configuration, all FPGAs will begin their initialization routines and wait for direction from the User Interface Board.

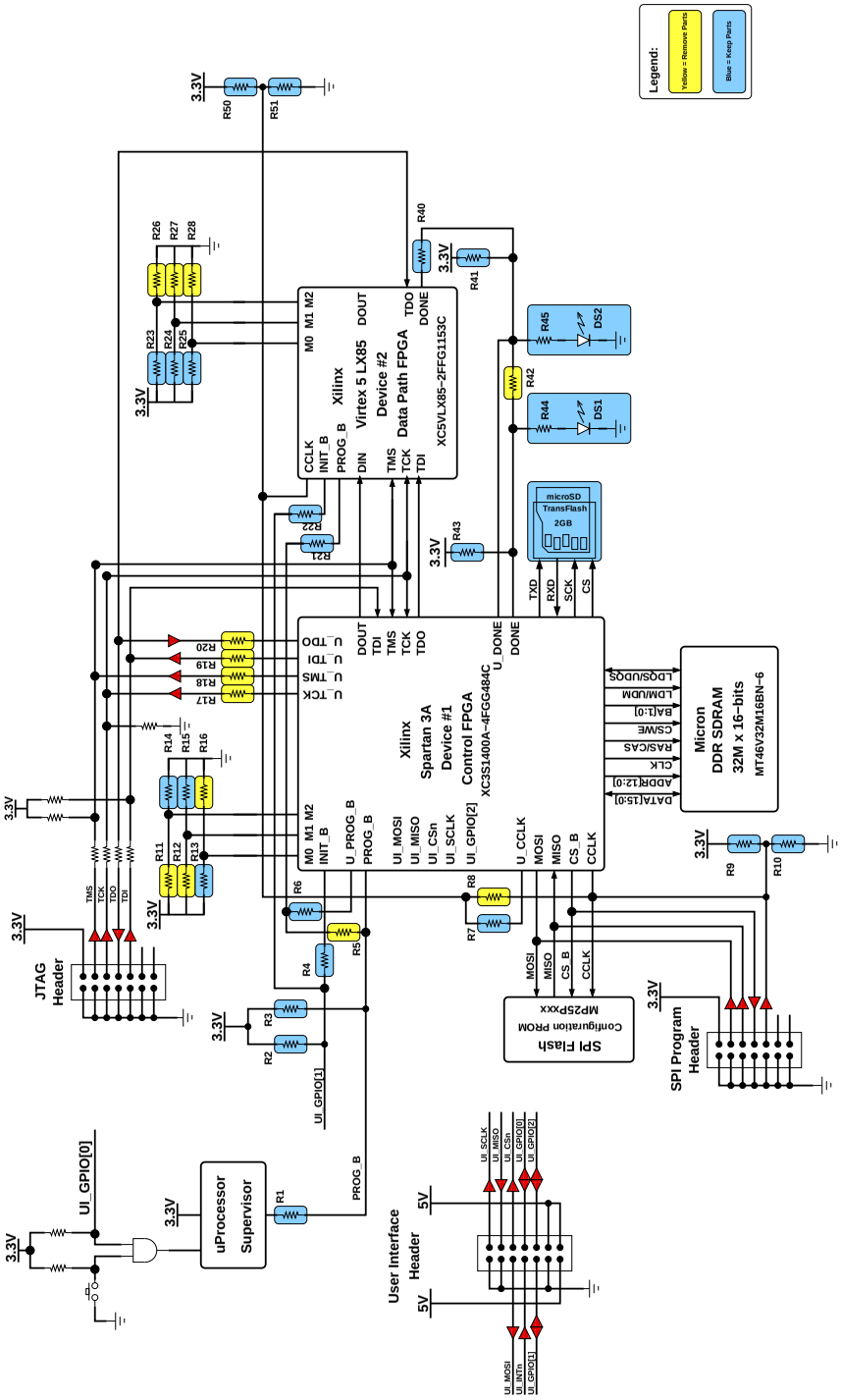


Figure 7.2: Slave Serial Configuration of FPGA Chain Block Diagram

7.1.3 Option #3: SPI Configuration and JTAG Daisy Chain

The Measurement Board Measurement Board has 3 Xilinx FPGAs and 1 SPI Configuration Flash PROM that need to be configured during power-up and a field upgrade. The main Control FPGA (Spartan 3A 1400A) is connected directly to the SPI Configuration Flash PROM, where it's configuration and application data reside. At power-up the SPI Configuration Flash PROM will download the bootloader image into the Control FPGA, which contains a MicroBlaze 32-bit microcontroller. Once the Control FPGA is configured, it will transfer the application code from the SPI Configuration Flash PROM to the 512Mb DDR SDRAM, and initiate a reprogramming of itself from the sector containing the operational image. After the Control FPGA is up and running the MicroBlaze application will grab an FPGA configuration file from the 2GB microSD card, and initiate a configuration of the remaining 2 FPGAs using the JTAG process via an XSVF player. As shown in Figure 7.3, the Control FPGA will drive the *TMS* and *TCK* pins of the JTAG chain, and then send data to the *TDI* pin of the 1^{st} device in the JTAG chain. During this process, the Control FPGA's XSVF player is monitoring the number of bits sent and will signal the completion of the JTAG configuration. Once the configuration process is completed, all FPGAs will begin their initialization routines and wait for direction from the User Interface Board.

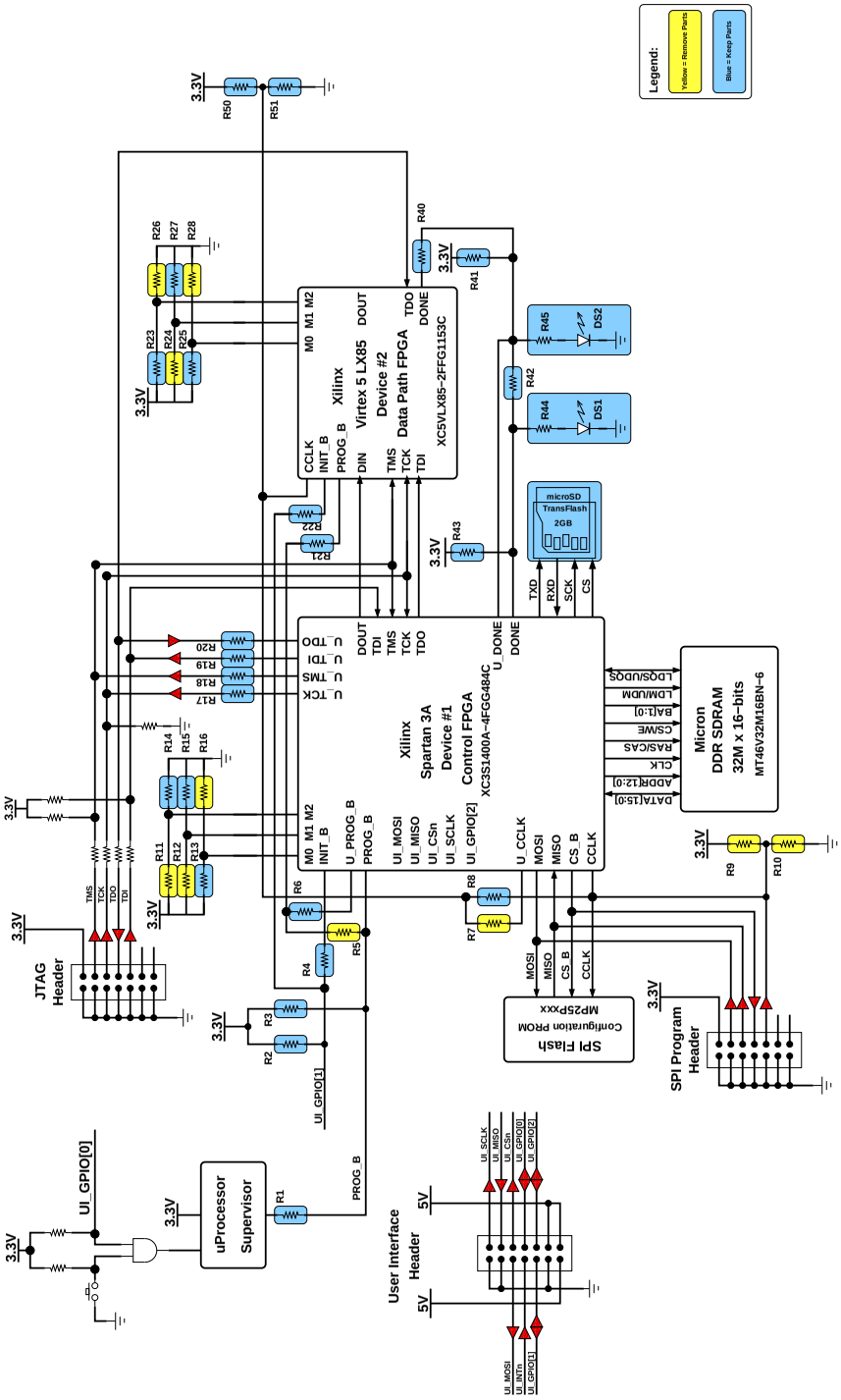


Figure 7.3: JTAG Configuration of FPGA Chain Block Diagram

7.1.4 Configuration Application Notes

For more information on Xilnx FPGA configuration see the following application notes:

- http://direct.xilinx.com/bvdocs/userguides/ug332.pdf.
- http://direct.xilinx.com/bvdocs/appnotes/xapp951.pdf.
- http://direct.xilinx.com/bvdocs/appnotes/xapp058.pdf.

  The files for the XAPP #: 058 can be found here: ftp://ftp.xilinx.com/pub/swhelp/cpld/eisp_pc.zip.

7.2 FPGA Configuration Storage Requirements

At the moment, the 128MB SPI Configuration Flash PROM is difficult to procure, so it is not a viable option. Since we have a 2GB microSD card on the Measurement Board Measurement Board for Pattern and Calibration data storage, we could store each FPGA image on the microSD card. The bootloader image will always reside at location 0x0 in the SPI Configuration Flash PROM to provide a reliable way of recovering from a configuration error.

When a Field Upgrade is initiated, the User Interface Board will transfer the new image to the Measurement Board where it will be stored in a new directory on the microSD card. The MicroBlaze 32-bit micro-controller will then perform a CRC/Checksum test to verify that the image is valid and move the operational image into one of two locations in the SPI Configuration Flash PROM with the new image.

At system Power-Up or restart, the Spartan 3A Control FPGA will be configured from the bootloader image stored at location 0x0 in the SPI Configuration Flash PROM. If the bootloader image was small enough to run out of block RAM, the MicroBlaze would transfer the application data, as part of the Power-Up procedure, from the SPI Configuration Flash PROM to the DDR SDRAM memory; an SPI serial daisy chain would be configured from the operational image.

This scheme has the potential of simplifying the Firmware Upgrade and Power-Up procedures executed by the MicroBlaze 32-bit micro-controller, because the MicroBlaze will not have to determine which operational image is the most recent.

7.3 SPI Configuration Flash PROM Organization

The SPI Configuration Flash PROM will be organized as shown in Figure 7.4. Figure 7.4 shows the images abutted next to each other, but in the final implementation the images will be re-aligned to the beginning of a sector.

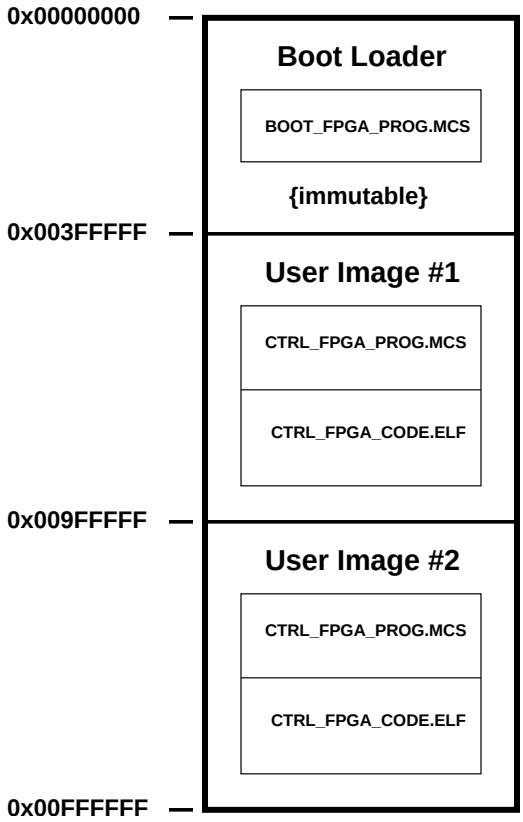


Figure 7.4: SPI Coniguration Flash PROM Organization

7.4 JTAG Mode Configuration

During HW debug and manufacturing, we will use JTAG mode to configure the FPGA chain. Figure 7.5 shows an example of how the Xilinx Platform Cable USB JTAG programming cable is connected to the Measurement Board.

Figure 7.5: Xilinx Platform Cable USB

8 AsAP Serial Bus Data Link Layer

8.1 Basic Assembly Control Description

AsAP chip assemblies on the AsAP Serial Bus are addressed by enabling the devices via a unique encoding of the chip select and load enable signals. Data is written and read from a device that has been selected.

8.2 AsAP Configuration Hardware Address Map

8.2.1 Configuration Interface Address Map

Table 8.1: Configuration Truth Table

Serial / Parallel Data			**Description**
data[19]	**data[18]**	**data[17:0]**	
Hi: [19] = 0 Low: [19] = 1	**~[18] = Addr [18] = Data**	**Address or Data**	
0b	0b	Upper Address	Upper Address word: config_addr[20:18]
1b	0b	Lower Address	Lower Address word: config_addr[17:0]
0b	1b	Upper Data	Upper Data word: config_data[34:18]
1b	1b	Lower Data	Lower Data word: config_data[17:0]
NOTE: The upper address word is sent with 18-bits, but the upper 15-bits are set to zero by the host sending the data over the proprietary serial interface. This allows us to use a single standard width shift register.			

8.3 AsAP Serial Bus Signals

The AsAP Serial Bus is a proprietary interface used to communicate with AsAP device on a printed circuit board assembly. The AsAP Serial Bus is based on the SPI bus. The AsAP Serial Bus is a 5 wire serial interface consisting of the following lines:

Table 8.2: Serial Bus Signal Descriptions

Signal Name	Description
SCK	Serial Clock. The serial clock is active when one of the strobe lines is low. Data is valid on the rising edge of SCK. All assemblies must be able to handle the enumeration clock rate when ADDRn is low. When not addressed, assemblies must be able to ignore SCK clock rates up to the maximum rate of the system they are installed in.
CSn	Chip Select, active low. This line idles in a high state. The serial address/data is sent in on MOSI while this strobe is low. The serial address/data is latched on the rising edge of CSn.
LOAD_EN	Load enable, active high. This line idles in a low state. The data sent in on MOSI is latched on the rising edge of LOAD_EN.
MISO	Master In/ Slave Out Data Line. This is data from the addressed assembly.
config_clk	Configuration Clock. This clocks in the config_valid signal into the configuration block of each AsAP processor in the processor array.
config_valid	Configuration Valid, active high. This line idles in a low state. This signal is toggled high, when the parallel data is valid. Toggles high two clocks after LOAD_EN toggles high.

The AsAP Serial Bus implementation inside the AsAP chip is considered a slave serial bus. The master serial bus is hosted by either a CPLD or an FPGA. The master can drive multiple AsAP chips by uniquely controlling the CSn and LOAD_EN signals. A complete description of this implementation can be found in the AsAP Serial Bus Engine document. The master sources the SCK, CSn, LOAD_EN, and MOSI lines. Only the MISO line is sourced from AsAP chip assemblies on the bus. The two additional signals: config_clk and config_valid, are not used by the AsAP Serial Bus inside of the AsAP chip assemblies, but they are used by the local configuration blocks within each processor in the processor array.

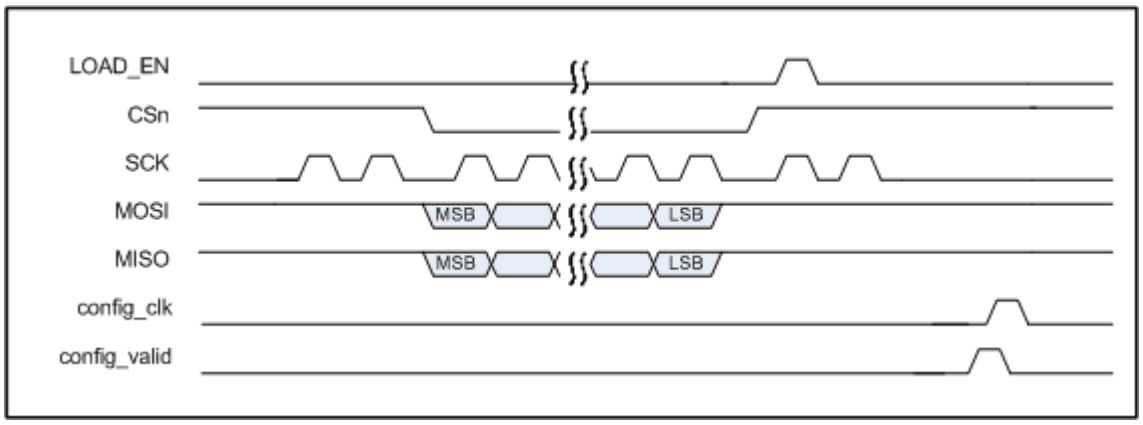
8.4 Bus Transactions

It typically takes four bus transactions to communicate with an AsAP chip assembly on the AsAP Serial Bus. The four transactions are the upper address transaction followed by a lower address transaction, an upper data transaction and a lower data transaction. The first transaction provides the upper 3-bits of the configuration address; the upper 15-bits are padded with zeros. The second transaction provides the lower 18-bits of the configuration address. The third transaction provides the upper 17-bits of configuration data. Finally, the fourth transaction provides the lower 18-bits of configuration data. In this transaction the actual data is written. The shift register data stays valid until another transaction occurs, and overwrites the register contents. All of the transactions are shifted in identically, since each transaction consists of 20-bits. The local configuration block of each processor in the array will pack the data appropriately. At the end of the fourth transaction a valid signal and configuration clock are toggled indicating to the local configuration block that it can perform the configuration writes to its local registers.

8.4.1 Single AsAP Serial Bus Transaction

1. SCK pulses are sent, and then CSn line goes low.
2. The bus master shifts sixteen bits out on the bus which are latched into the shift register of the AsAP when CSn goes high. Below is a drawing of a Single Transaction.
3. One SCK after CSn goes high, LOAD_EN toggles high then low to latch the parallel data.
4. Two SCKs after LOAD_EN toggles, config_clk and config_valid will be driven. This allows enough settling time of the latched data.

Figure 8.1: Single Bus Transaction



8.4.2 A Complete Serial Bus Transfer

Below all four transactions are shown together as in an actual control sequence of an AsAP chip assembly. A description of the complete serial bus transfer is listed below.

1. Two SCK pulses are sent, and then CSn line goes low.
2. The bus master shifts sixteen bits out on the bus which are latched into the shift register of the AsAP when CSn goes high. Below is a drawing of a Single Transaction.
3. One SCK after CSn goes high, LOAD_EN toggles high then low to latch the parallel data.
4. Two SCKs after LOAD_EN toggles, config_clk and config_valid will be driven. This allows enough settling time of the latched data.
5. Steps 1 through 4 are repeated for each 16-bit word in the following order: Upper Address, Lower Address, Upper Data, Lower Data.

Figure 8.2: Complete Upper and Lower Address Serial Bus Transfer

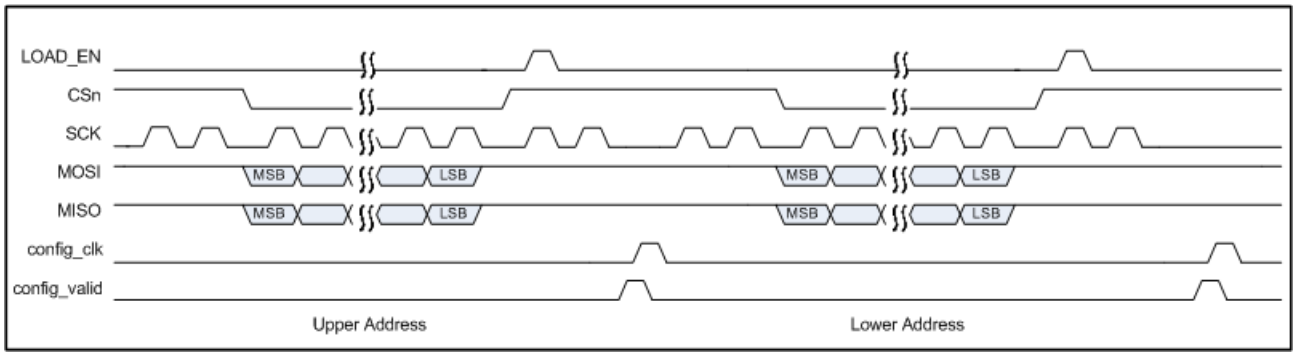
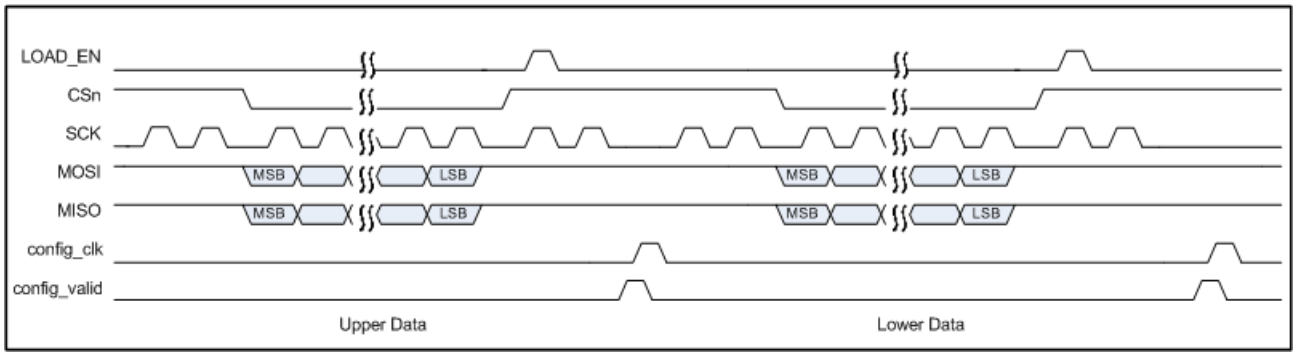


Figure 8.3: Complete Upper and Lower Data Serial Bus Transfer



8.5 Bus Signal Topology

The AsAP Serial Bus consists of three edge sensitive signals and two level sensitive signals in a multi-drop topology. To support a maximum data rate of 16Mb/s, yet allow filtering when necessary for isolation between assemblies connected to the bus, it is essential that proper sourcing, transmission, receiving, and filtering requirements be adhered to. Figure 2.5 1 shows the overall system layout while Figure 2.5 2 shows the lower level details. Refer to Table 2.5 1 and Table 2.5 2 for parameter values.

Figure 8.4: General Bus Signal Topology

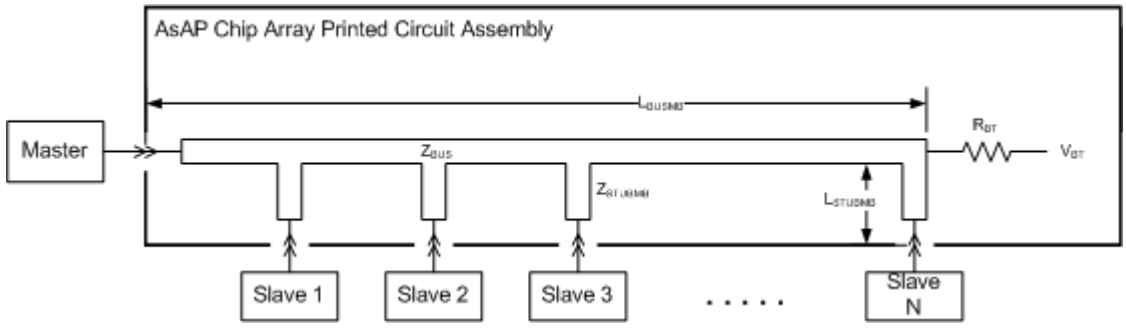
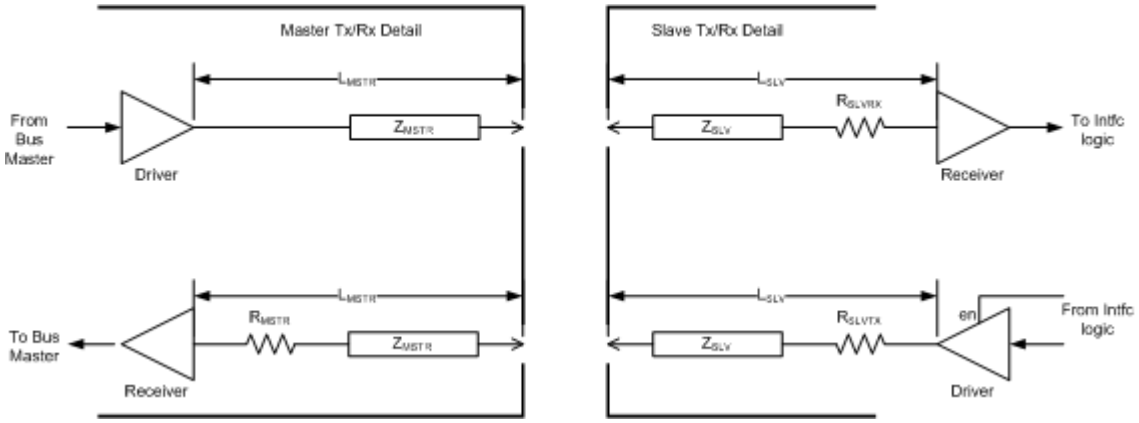


Figure 8.5: Master/Slave Detail



8.5.1 Bus Parameters

Table 8.3: General Bus Parameters

Symbol	Parameter	Conditions	Min	Typ	Max	Units
N	Number of Slaves			10	15	
L_{BUSTOT}	Total Bus Length	$L_{BUSMB} + L_{MSTR}$		35	50	cm
$L_{STUBTOT}$	Total Stub Length	$L_{STUBMB} + L_{SLV}$			4	cm
Z_{BUS}	Bus Impedance		50	50	75	Ω
Z_{STUB}	Stub Impedance		50		75	Ω

Table 8.4: Signal Specific Parameters

Signal	Idle State	R_{BT}	V_{BT}	R_{SLVRX}	R_{SLVTX}	R_{MSTR}
SCK	0V	Z_{BUS} [1]	1.65V [2]	220 Ω		
ADDRn	3.3V	Z_{BUS} [1]	1.65V [2]	220 Ω		
DATAn	3.3V	Z_{BUS} [1]	1.65V [2]	220 Ω		
MOSI		NONE	NONE	220 Ω		
MISO	3.3V	3kΩ	3.3V		Z_{BUS}-R_O	220 Ω

1. Since Z_{BUS} is determined by motherboard trace routing, the final routing will dictate the bus terminating impedance.
2. V_{BT} can be supplied as a Thevinin equivalent voltage off the +3.3V supply with the Thevinin equivalent resistance equal to R_{BT}. The rationale for using a midpoint termination voltage is to optimize the signal swing on the bus given a non-zero impedance of the driver and the desire to provide the proper termination impedance.

8.5.2 Bus Signal Filtering

The impedance presented to the bus by slaves must not exceed that of a single gate per slave. Therefore bus cannot be filtered directly. If filtering is needed on a slave to prevent noise from entering or leaving on the bus, the filter must be isolated from the bus with a buffer. The buffer could be discrete or part of an existing CPLD/FPGA. However, the discrete buffer/filter would probably offer better isolation from noise generated with the CPLD/FPGA. The transmit signal, MISO, uses a tri-state driver on the slaves. The enable line as well as the driver input should be filtered.

Filtering on slaves as described above shall have a minimum bandwidth necessary to support the enumeration data rate (3x data rate suggested).

8.6 Electrical Specifications

8.6.1 DC Electrical Specifications

ABSOLUTE MAXIMUM RATINGS
Input Voltage (V_i - 0.5V) to (+3.3V + 0.5V)

Table 8.5: DC Characteristics for PC Boards

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{IH}	Input High Voltage		2.0		3.3 + 0.5	V
V_{IL}	Input Low Voltage		-0.5		0.8	V
C_{IN}	Input Pin Capacitance	$T_A = 25^oC$, f = 1MHz			10	pF
C_{CLK}	Clock Input Capacitance	$T_A = 25^oC$, f = 1MHz			10	pf
C_O	Output Pin Capacitance	$T_A = 25^oC$, f = 1MHz			10	pf

8.6.2 AC Electrical Specifications

Table 8.6: Data Rates

Symbol	Parameter	Conditions	Min	Typ	Max	Units
	Data Rate				20	MHz

8.6.3 Timing Parameters

Table 8.7: Timing Parameters

Symbol	Parameter	Min	Max	Units	Notes
T_{SSU}	Chip Select Setup	20		ns	
T_{SH}	Chip Select Hold	20		ns	
T_{SS}	Chip Select to Chip Select	TBD		ns	
T_{DOSU}	Data Output Setup	20		ns	
T_{DOH}	Data Output Hold	5		ns	
T_{DISU}	Data Input Setup	20		ns	
T_{DIH}	Data Input Hold	5		ns	
T_{LDSU}	Chip Select to Load Enable	100		ns	
NOTE: Setup and Hold times are based on a 74HC595 as being the slowest device to ever be put on the bus.					

In figure 8.6, STROBE refers to ADDRn, or DATAn.

Figure 8.6: Chip Select, Load Enable, and Data Timing

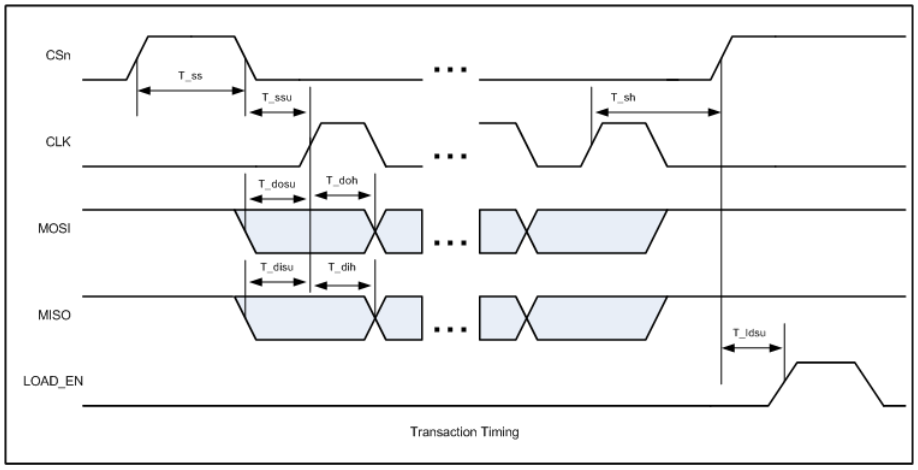
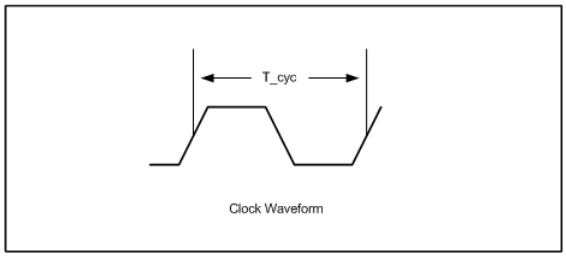


Table 8.8: Clock Period

Symbol	Parameter	Min	Max	Units	NOTES
T_{CYC}	Clock Cycle Time	50	-	ns	

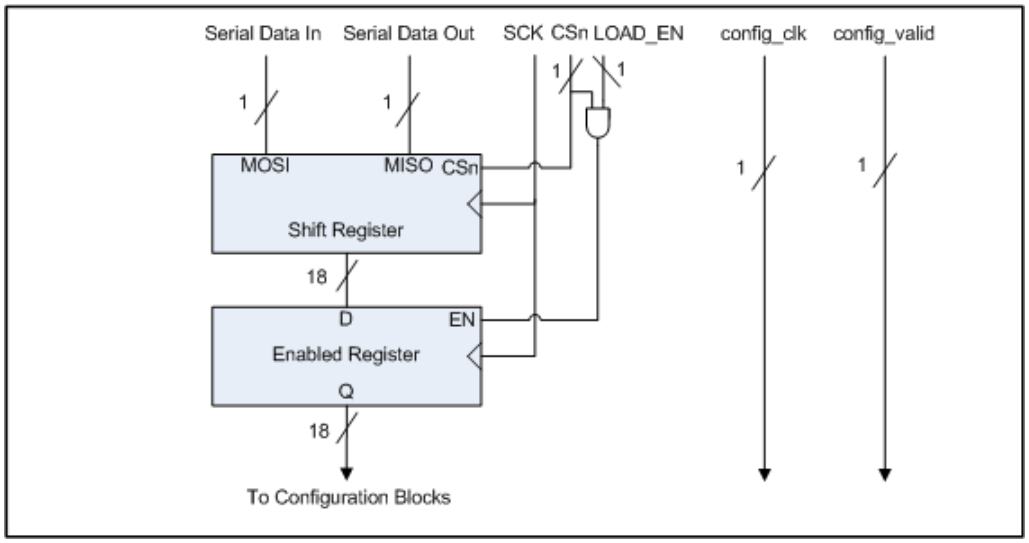
Figure 8.7: Clock Waveform



9 AsAP Serial Bus User Guide

9.1 AsAP Serial Bus Slave Design

Figure 9.1: AsAP Serial Bus Slave Block Diagram



10 Control FPGA Pin Out

The Measurement board contains an Xilinx Spartan 3A XC3S1400A FPGA for controlling the measurement control circuitry. The Xilinx part number is: XC3S1400A-4FG484.

Table 10.1: Xilinx Spartan 3A XC3S1400A FPGA Pin Out

Xilinx Spartan 3A XC3S1400A FPGA Pin Out			
Signal Name	**Bank #**	**Pin #**	**Description**
Dedicated Configuration Signals			
S3A_PROG_B	NA	C4	Spartan-3A Program Signal (Active Low).
S3A_CONFIG_DONE	NA	Y19	Spartan-3A Configuration Done (Active High).
TCK	NA	A21	JTAG Configuration TCK Signal.
TDI_TO_S3A	NA	F5	JTAG Configuration TDI Signal.
TDO_TO_V5	NA	E19	JTAG Configuration TDO Signal.
TMS	NA	D4	JTAG Configuration TMS Signal.
Bank 0			
No Connect	0	D19	Unused Pin.
No Connect	0	B19	Unused Pin.
No Connect	0	A20	Unused Pin.
No Connect	0	C19	Unused Pin.
No Connect	0	B20	Unused Pin.
No Connect	0	E17	Unused Pin.
No Connect	0	A18	Unused Pin.
No Connect	0	C18	Unused Pin.
No Connect	0	D18	Unused Pin.
No Connect	0	A19	Unused Pin.
No Connect	0	D16	Unused Pin.
No Connect	0	B17	Unused Pin.
No Connect	0	C17	Unused Pin.
FPGA_TMPSENS_2C_MOSI	0	D17	Temperature Sensor 2C Master-Out-Slave-In (MOSI).
FPGA_TMPSENS_2C_CSN	0	A17	Temperature Sensor 2C Chip Select (Active Low).
FPGA_CLK10MHZ_REF_CTRL[0]	0	B2	10MHz Reference Control Bit 0.
FPGA_CLK10MHZ_REF_CTRL[2]	0	A3	10MHz Reference Control Bit 2.
FPGA_CLK10MHZ_REF_CTRL[1]	0	B3	10MHz Reference Control Bit 1.
FPGA_EXT_CLK10MHZ_LOS	0	A2	10MHz Reference Loss of Signal (Active High).
FPGA_UI_MOSI	0	B4	User Interface Board SPI Master-Out-Slave-In (MOSI).
FPGA_UI_RDY	0	A5	User Interface Board Ready (Active High).
FPGA_UI_RSTN	0	C5	User Interface Board Reset (Active Low).
FPGA_CLK10MHZ_REF_CTRL[3]	0	A4	10MHz Reference Control Bit 3.
FPGA_TMPSENS_1B_SCK	0	F7	Temperature Sensor 1B Serial Clock.
FPGA_TMPSENS_1B_MISO	0	E6	Temperature Sensor 1B Master-In-Slave-Out (MISO).
FPGA_TMPSENS_1B_CSN	0	E7	Temperature Sensor 1B Chip Select (Active Low).
FPGA_UI_SCK	0	D5	User Interface Board SPI Serial Clock.
FPGA_UI_INTN	0	B6	User Interface Board Interrupt (Active Low).
FPGA_UI_CSN	0	C6	User Interface Board SPI Chip Select (Active Low).
FPGA_LOCAL_RSTN	0	D6	Measurement Board Local Reset (Active Low).
FPGA_UI_MISO	0	A6	User Interface Board SPI Master-In-Slave-Out (MISO).
FPGA_LEDS[1]	0	C7	LED 1 (Active Low).
			Continued on Next Page...

Xilinx Spartan 3A XC3S1400A FPGA Pin Out			
Signal Name	**Bank #**	**Pin #**	**Description**
No Connect	0	A7	Unused Pin.
No Connect	0	A8	Unused Pin.
FPGA_S3A_BOARD_RSTN	0	D7	Measurement Board Reset (Active Low).
FPGA_PUSHBUTTON[0]	0	H9	Push-Button 0 (Active Low).
FPGA_LEDS[3]	0	B8	LED 3 (Active Low).
No Connect	0	A9	Unused Pin.
No Connect	0	H10	Unused Pin.
No Connect	0	G10	Unused Pin.
No Connect	0	F10	Unused Pin.
No Connect	0	E10	Unused Pin.
No Connect	0	D10	Unused Pin.
No Connect	0	C10	Unused Pin.
No Connect	0	A10	Unused Pin.
FPGA_INT_CLK10MHZ_REF_N	0	A11	Internal 10MHz Reference Clock (Negative Differential).
FPGA_EXT_CLK10MHZ_REF_P	0	B11	External 10MHz Reference Clock (Positive Differential).
FPGA_EXT_CLK10MHZ_REF_N	0	C11	External 10MHz Reference Clock (Negative Differential).
No Connect	0	D11	Unused Pin.
No Connect	0	E11	Unused Pin.
No Connect	0	G11	Unused Pin.
No Connect	0	G12	Unused Pin.
No Connect	0	H12	Unused Pin.
No Connect	0	H13	Unused Pin.
FPGA_LEDS[2]	0	C8	LED 2 (Active Low).
No Connect	0	B9	Unused Pin.
No Connect	0	C9	Unused Pin.
FPGA_LEDS[0]	0	D8	LED 0 (Active Low).
No Connect	0	C12	Unused Pin.
No Connect	0	E12	Unused Pin.
FPGA_INT_CLK10MHZ_REF_P	0	A12	Internal 10MHz Reference Clock (Positive Differential).
No Connect	0	F16	Unused Pin.
No Connect	0	E16	Unused Pin.
No Connect	0	G15	Unused Pin.
No Connect	0	E15	Unused Pin.
No Connect	0	F15	Unused Pin.
No Connect	0	G16	Unused Pin.
No Connect	0	G14	Unused Pin.
No Connect	0	C16	Unused Pin.
No Connect	0	C14	Unused Pin.
FPGA_LA_DATA[1]	0	E14	Logic Analyzer Data 1.
No Connect	0	H14	Unused Pin.
FPGA_LA_DATA[4]	0	D15	Logic Analyzer Data 4.
No Connect	0	C15	Unused Pin.
No Connect	0	A16	Unused Pin.
FPGA_DDR2_SDRAM_SCL	0	A15	DDR2 SDRAM SODIMM I^2C Serial Clock.
FPGA_DDR2_SDRAM_SDA	0	B15	DDR2 SDRAM SODIMM I^2C Serial Data.
FPGA_LA_CLK	0	A14	Logic Analyzer Clock.
No Connect	0	G13	Unused Pin.
FPGA_LA_DATA[0]	0	F13	Logic Analyzer Data 0.
FPGA_LA_DATA[2]	0	E13	Logic Analyzer Data 2.
FPGA_LA_DATA[3]	0	D13	Logic Analyzer Data 3.

Continued on Next Page...

Xilinx Spartan 3A XC3S1400A FPGA Pin Out			
Signal Name	**Bank #**	**Pin #**	**Description**
FPGA_LA_DATA[5]	0	C13	Logic Analyzer Data 5.
No Connect	0	F12	Unused Pin.
FPGA_LA_DATA[6]	0	B13	Logic Analyzer Data 6.
FPGA_LA_DATA[7]	0	A13	Logic Analyzer Data 7.
No Connect	0	E9	Unused Pin.
No Connect	0	E8	Unused Pin.
No Connect	0	G9	Unused Pin.
FPGA_TMPSENS_1B_MOSI	0	F8	Temperature Sensor 1B Master-Out-Slave-In (MOSI).
FPGA_PUSHBUTTON[1]	0	G8	Push-Button 1 (Active Low).
FPGA_PUSHBUTTON[2]	0	G7	Push-Button 2 (Active Low).
Bank 1			
FPGA_ASAP1_SPI_LOAD	1	W21	AsAP #1 Serial Load Enable (Active High).
FPGA_ASAP1_SPI_CLK	1	Y21	AsAP #1 Serial Clock.
FPGA_ASAP1_CFG_CLK	1	W20	AsAP #1 Configuration Clock.
FPGA_ASAP1_SPI_MOSI	1	Y22	AsAP #1 Serial Master-Out-Slave-In (MOSI).
FPGA_ASAP1_SPI_CSN	1	W22	AsAP #1 Serial Chip Select (Active Low).
No Connect	1	T17	Unused Pin.
No Connect	1	T18	Unused Pin.
FPGA_DAC5682_SDO	1	U19	DAC5682Z Serial Data Output.
FPGA_ASAP1_RESET_COLD	1	V19	AsAP #1 Reset Cold (Active High).
FPGA_ASAP1_CFG_VALID	1	W19	AsAP #1 Configuration Valid.
FPGA_ASAP1_RST_CNTCLK	1	V20	AsAP #1 Reset Counter Clock.
FPGA_DAC5682_RSTB	1	U20	DAC5682Z Reset (Active Low).
FPGA_DAC5682_SDENB	1	U21	DAC5682Z Serial Data Enable (Active Low).
FPGA_DAC5682_SCLK	1	V22	DAC5682Z Serial Clock.
No Connect	1	T19	Unused Pin.
No Connect	1	T20	Unused Pin.
FPGA_DAC5682_SDIO	1	U22	DAC5682Z Serial Data Input.
FPGA_TMPSENS_2A_SCK	1	R19	Temperature Sensor 2A Serial Clock.
FPGA_TMPSENS_2A_MOSI	1	R20	Temperature Sensor 2A Master-Out-Slave-In (MOSI).
No Connect	1	T22	Unused Pin.
FPGA_TMPSENS_2A_CSN	1	R21	Temperature Sensor 2A Chip Select (Active Low).
No Connect	1	R22	Unused Pin.
FPGA_AD9516_PD	1	P20	AD9516 Power Down (Active Low).
CUST_INIT_B	1	B21	Virtex-5 INIT_B (Active Low).
FPGA_DP_CTRL_GPIO[4]	1	E22	Data Path Control GPIO Bit 4.
CUST_CCLK	1	C21	Virtex-5 Configuration Clock.
CUST_PROG_B	1	C22	Virtex-5 PROG_B (Active Low).
No Connect	1	E20	Unused Pin.
FPGA_DP_CTRL_GPIO[3]	1	D22	Data Path Control GPIO Bit 3.
FPGA_DP_CTRL_GPIO[2]	1	D21	Data Path Control GPIO Bit 2.
FPGA_DP_CTRL_MISO	1	F18	Data Path Control Serial Master-In-Slave-Out (MISO).
FPGA_DP_CTRL_RSTN	1	F19	Data Path Control Reset (Active Low).
FPGA_DP_CTRL_SCK	1	F20	Data Path Control Serial Clock.
FPGA_DP_CTRL_CSN	1	F21	Data Path Control Serial Chip Select (Active Low).
FPGA_DP_CTRL_MOSI	1	F22	Data Path Control Serial Master-Out-Slave-In (MOSI).
No Connect	1	G22	Unused Pin.
FPGA_DP_CTRL_GPIO[1]	1	D20	Data Path Control GPIO Bit 1.
FPGA_ASAP1_SPI_MISO	1	AA22	AsAP #1 Serial Master-In-Slave-Out (MISO).
CUST_CFG_DONE	1	B22	Virtex-5 Configuration Done (Active High).
			Continued on Next Page...

Xilinx Spartan 3A XC3S1400A FPGA Pin Out			
Signal Name	**Bank #**	**Pin #**	**Description**
FPGA_AD9516_SYNC	1	M22	AD9516 Sync.
No Connect	1	K17	Unused Pin.
No Connect	1	K18	Unused Pin.
No Connect	1	K19	Unused Pin.
No Connect	1	L19	Unused Pin.
FPGA_AD9516_CSB	1	N19	AD9516 Chip Select (Active Low).
FPGA_AD9516_SDIO	1	N20	AD9516 Serial Data Input.
FPGA_AD9516_RESET	1	N21	AD9516 Reset (Active High).
FPGA_AD9516_REF_SEL	1	N22	AD9516 Reference Select.
No Connect	1	L22	Unused Pin.
No Connect	1	L21	Unused Pin.
No Connect	1	L20	Unused Pin.
No Connect	1	M18	Unused Pin.
No Connect	1	M20	Unused Pin.
No Connect	1	K20	Unused Pin.
No Connect	1	U18	Unused Pin.
No Connect	1	R15	Unused Pin.
No Connect	1	R16	Unused Pin.
No Connect	1	P15	Unused Pin.
No Connect	1	P16	Unused Pin.
No Connect	1	R17	Unused Pin.
FPGA_TMPSENS_2A_MISO	1	R18	Temperature Sensor 2A Master-In-Slave-Out (MISO).
FPGA_AD9516_SDO	1	P22	AD9516 Serial Data Output.
No Connect	1	N15	Unused Pin.
No Connect	1	N16	Unused Pin.
FPGA_AD9516_LD	1	P18	AD9516 Lock Detect.
FPGA_AD9516_SCLK	1	N18	AD9516 Serial Clock.
FPGA_AD9516_STATUS	1	N17	AD9516 Status.
FPGA_AD9516_REFMON	1	M17	AD9516 Reference Monitor.
No Connect	1	M16	Unused Pin.
FPGA_TMPSENS_2B_MOSI	1	J22	Temperature Sensor 2B Master-Out-Slave-In (MOSI).
FPGA_TMPSENS_2B_CSN	1	K22	Temperature Sensor 2B Chip Select (Active Low).
No Connect	1	M15	Unused Pin.
No Connect	1	L16	Unused Pin.
No Connect	1	L18	Unused Pin.
FPGA_TMPSENS_2B_SCK	1	J21	Temperature Sensor 2B Serial Clock.
FPGA_TMPSENS_2B_MISO	1	J20	Temperature Sensor 2B Master-In-Slave-Out (MISO).
No Connect	1	H22	Unused Pin.
No Connect	1	L15	Unused Pin.
No Connect	1	K16	Unused Pin.
No Connect	1	H21	Unused Pin.
No Connect	1	H20	Unused Pin.
No Connect	1	K14	Unused Pin.
No Connect	1	K15	Unused Pin.
No Connect	1	G19	Unused Pin.
No Connect	1	G20	Unused Pin.
No Connect	1	J18	Unused Pin.
No Connect	1	H19	Unused Pin.
No Connect	1	H17	Unused Pin.
No Connect	1	H18	Unused Pin.

Continued on Next Page...

Xilinx Spartan 3A XC3S1400A FPGA Pin Out			
Signal Name	**Bank #**	**Pin #**	**Description**
FPGA_DP_CTRL_GPIO[0]	1	J16	Data Path Control GPIO Bit 0.
No Connect	1	J15	Unused Pin.
FPGA_DP_CTRL_INTN	1	G18	Data Path Service Request (Active Low).
No Connect	1	G17	Unused Pin.
No Connect	1	H16	Unused Pin.
No Connect	1	H15	Unused Pin.
Bank 2			
FPGA_FTDI_RDN	2	U13	FTDI Read Enable (Active Low).
FPGA_TMPSENS_1C_CSN	2	Y17	Temperature Sensor 1C Chip Select (Active Low).
FPGA_ASAP2_RST_CNTCLK	2	AA17	AsAP #2 Reset Counter Clock.
FPGA_AMC6821_FAN2_SCK	2	AB9	AMC6821 Fan #2 I^2C Serial Clock.
FPGA_SLOTID[2]	2	T11	Slot ID Bit 2.
FPGA_REACH_TX	2	U12	Reach Technologies Display RS-232 Transmit Data.
FPGA_REACH_RX	2	V12	Reach Technologies Display RS-232 Receive Data.
FPGA_CLK100MHZ_N	2	AB12	100MHz Clock (Negative Differential).
FPGA_CLK100MHZ_P	2	AA12	100MHz Clock (Positive Differential).
No Connect	2	Y12	Unused Pin.
No Connect	2	W12	Unused Pin.
FPGA_RS232_RX	2	AB5	CP2102 RS-232 Receive Data.
FPGA_TMPSENS_1A_MISO	2	AB6	Temperature Sensor 1A Master-In-Slave-Out (MISO).
FPGA_RS232_CTS	2	Y6	CP2102 RS-232 Clear To Send Signal.
No Connect	2	W6	Unused Pin.
FPGA_RS232_RTS	2	AA6	CP2102 RS-232 Ready To Send Signal.
FPGA_SD_BUSY_LED	2	AB4	microSD Card Busy LED (Active Low).
FPGA_SD_RXD	2	AB3	microSD Card Receive Data.
FPGA_SD_CLK	2	AA3	microSD Card Serial Clock.
FPGA_SLOTID[1]	2	T10	Slot ID Bit 1.
FPGA_SD_TXD	2	AB2	microSD Card Transmit Data.
FPGA_AMC6821_FAN2_SMBALERTN	2	T9	AMC6821 Fan #2 SMB Alert (Active Low).
FPGA_SD_CARD_DETECT	2	AA4	microSD Card Chip Select (Active Low).
FPGA_AMC6821_FAN1_FAULTN	2	T7	AMC6821 Fan #1 Fault Flag (Active Low.)
FPGA_AMC6821_FAN1_THERMN	2	U7	AMC6821 Fan #1 Thermal Flag (Active Low.)
FPGA_SLOTID[0]	2	R11	Slot ID Bit 0.
FPGA_FTDI_TXEN	2	P12	FTDI Transmit Enable (Active Low).
FPGA_FTDI_RXFN	2	R12	FTDI Receive Data Enable (Active Low).
FPGA_FTDI_PWRENN	2	R13	FTDI Power Enable (Active Low).
No Connect	2	U11	Unused Pin.
FPGA_RS232_TX	2	Y5	CP2102 RS-232 Transmit Data.
No Connect	2	T16	Unused Pin.
FPGA_TMPSENS_1C_MISO	2	U16	Temperature Sensor 1C Master-In-Slave-Out (MISO).
FPGA_TMPSENS_1C_SCK	2	V17	Temperature Sensor 1C Serial Clock.
FPGA_AMC6821_FAN1_OVRN	2	T8	AMC6821 Fan #1 Over Temp Flag (Active Low.)
FPGA_AMC6821_FAN1_SMBALERTN	2	V7	AMC6821 Fan #1 SMB Alert (Active Low).
FPGA_AMC6821_FAN2_FAULTN	2	U8	AMC6821 Fan #2 Fault Flag (Active Low.)
FPGA_AMC6821_FAN1_SCK	2	W7	AMC6821 Fan #1 I^2C Serial Clock.
FPGA_AMC6821_FAN2_THERMN	2	V8	AMC6821 Fan #2 Thermal Flag (Active Low.)
FPGA_AMC6821_FAN1_SDA	2	Y7	AMC6821 Fan #1 I^2C Serial Data.
FPGA_TMPSENS_1A_SCK	2	AB7	Temperature Sensor 1A Serial Clock.
FPGA_AMC6821_FAN2_SDA	2	Y9	AMC6821 Fan #2 I^2C Serial Data.
CUST_TDO	2	U10	Custom JTAG TDO Signal for Configuration via MicroBlaze.
			Continued on Next Page...

Xilinx Spartan 3A XC3S1400A FPGA Pin Out			
Signal Name	**Bank #**	**Pin #**	**Description**
FPGA_TMPSENS_1A_CSN	2	AA8	Temperature Sensor 1A Chip Select (Active Low).
FPGA_TMPSENS_1A_MOSI	2	AB8	Temperature Sensor 1A Master-Out-Slave-In (MOSI).
CUST_TCK	2	Y10	Custom JTAG TCK Signal for Configuration via MicroBlaze.
JTAG_CCN	2	V10	Xilinx JTAG Cable Connected Signal (Active Low).
CUST_TDI	2	AA10	Custom JTAG TDI Signal for Configuration via MicroBlaze.
CUST_TMS	2	AB10	Custom JTAG TMS Signal for Configuration via MicroBlaze.
FPGA_FTDI_WRN	2	V14	FTDI Write Enable (Active Low).
FPGA_FTDI_DATA[3]	2	W16	FTDI Data Bit 3.
FPGA_FTDI_DATA[0]	2	V15	FTDI Data Bit 0.
FPGA_FTDI_DATA[1]	2	V16	FTDI Data Bit 1.
No Connect	2	Y13	Unused Pin.
No Connect	2	Y14	Unused Pin.
FPGA_FTDI_SI_WU	2	W13	FTDI Send Immediate / Wake Up Enable.
FPGA_FTDI_DATA[2]	2	W15	FTDI Data Bit 2.
S3A_SPI_WPN	2	AA14	Spartan-3A SPI Write Protect Signal (Active Low).
S3A_SPI_HOLDN	2	AB13	Spartan-3A SPI Hold Signal (Active Low).
FPGA_TMPSENS_1C_MOSI	2	W17	Temperature Sensor 1C Master-Out-Slave-In (MOSI).
FPGA_FTDI_DATA[4]	2	Y15	FTDI Data Bit 4.
FPGA_FTDI_DATA[6]	2	AB15	FTDI Data Bit 6.
FPGA_FTDI_DATA[7]	2	AB16	FTDI Data Bit 7.
S3A_SPI_DATA_TO_V5	2	AA15	Spartan-3A Serial Data Output to Virtex-5.
FPGA_FTDI_DATA[6]	2	Y16	FTDI Data Bit 6.
FPGA_ASAP2_RESET_COLD	2	AB17	AsAP #2 Reset Cold (Active High).
FPGA_ASAP2_SPI_CLK	2	AB18	AsAP #2 Serial Clock.
FPGA_ASAP2_MOSI	2	Y18	AsAP #2 Master-Out-Slave-In (MOSI).
FPGA_ASAP2_MISO	2	W18	AsAP #2 Master-In-Slave-Out (MISO).
FPGA_ASAP2_CFG_CLK	2	AA21	AsAP #2 Configuration Clock.
FPGA_ASAP2_CFG_VALID	2	AB21	AsAP #2 Configuration Valid.
FPGA_ASAP2_SPI_CSN	2	AA19	AsAP #2 Serial Chip Select (Active Low).
FPGA_ASAP2_SPI_LOAD	2	AB19	AsAP #2 Serial Load Enable (Active High).
FPGA_AMC6821_FAN2_OVRN	2	V9	AMC6821 Fan #1 Over Temp Flag (Active Low.)
S3A_CCLK	2	AA20	Spartan-3A Configuration Clock.
S3A_INIT_B	2	V13	Spartan-3A Configuration Initialization (Active Low).
S3A_M0	2	W5	Spartan-3A Configuration Mode Pin 0.
S3A_M1	2	V6	Spartan-3A Configuration Mode Pin 1.
S3A_M2	2	W4	Spartan-3A Configuration Mode Pin 2.
S3A_SPI_CSO	2	Y4	Spartan-3A SPI Chip Select (Active Low).
No Connect	2	AB11	Unused Pin.
S3A_SPI_MISO	2	AB20	Spartan-3A SPI Master-In-Slave-Out (MISO).
S3A_SPI_MOSI	2	AB14	Spartan-3A SPI Master-Out-Slave-In (MOSI).
No Connect	2	Y11	Unused Pin.
S3A_V0	2	Y8	Spartan-3A SPI Mode Pin 0.
S3A_V1	2	W8	Spartan-3A SPI Mode Pin 1.
S3A_V2	2	W9	Spartan-3A SPI Mode Pin 2.
No Connect	2	V11	Unused Pin.
No Connect	2	T13	Unused Pin.
No Connect	2	T15	Unused Pin.
No Connect	2	U15	Unused Pin.
No Connect	2	T14	Unused Pin.
FPGA_HWID[0]	2	R9	Hardware ID Bit 0.

Continued on Next Page...

Xilinx Spartan 3A XC3S1400A FPGA Pin Out			
Signal Name	**Bank #**	**Pin #**	**Description**
FPGA_FTDI_RSTOUTN	2	R14	FTDI Reset Output (Active Low).
FPGA_HWID[1]	2	R10	Hardware ID Bit 1.
Bank 3			
FPGA_DDR_SDRAM_ADDR[0]	3	K5	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[1]	3	K2	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[2]	3	K3	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[3]	3	L3	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[4]	3	L5	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[5]	3	L1	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[6]	3	K1	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[7]	3	M2	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[8]	3	M1	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[9]	3	M4	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[10]	3	M3	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[11]	3	Y2	DDR SDRAM Address.
FPGA_DDR_SDRAM_ADDR[12]	3	Y1	DDR SDRAM Address.
FPGA_DDR_SDRAM_BA[0]	3	H2	DDR SDRAM Bank Address.
FPGA_DDR_SDRAM_BA[1]	3	K4	DDR SDRAM Bank Address.
FPGA_DDR_SDRAM_CASN	3	G3	DDR SDRAM Column Address Strobe (Active Low).
FPGA_DDR_SDRAM_CKE	3	H1	DDR SDRAM Clock Enable (Active Low).
+1.25V DDR VREF	3	R6	DDR SDRAM Reference Voltage.
FPGA_DDR_SDRAM_CLKFB	3	W3	DDR SDRAM Clock Feedback Input.
FPGA_DDR_SDRAM_CLK_N	3	AA2	DDR SDRAM Clock (Negative Differential).
FPGA_DDR_SDRAM_CLK_P	3	AA1	DDR SDRAM Clock (Positive Differential).
FPGA_DDR_SDRAM_CSN	3	H4	DDR SDRAM Chip Select (Active Low).
FPGA_DDR_SDRAM_DATA[0]	3	W1	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[1]	3	W2	DDR SDRAM Data.
FPGA_DDR_SDRAM_CLKFB	3	V4	DDR SDRAM Clock Feedback Output.
FPGA_DDR_SDRAM_DATA[2]	3	U3	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[3]	3	U4	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[4]	3	V1	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[5]	3	V3	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[6]	3	U1	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[7]	3	U2	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[8]	3	R5	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[9]	3	R4	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[10]	3	R3	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[11]	3	R2	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[12]	3	P3	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[13]	3	P5	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[14]	3	P1	DDR SDRAM Data.
FPGA_DDR_SDRAM_DATA[15]	3	P2	DDR SDRAM Data.
FPGA_DDR_SDRAM_LDM	3	T4	DDR SDRAM Lower Data Mask.
FPGA_DDR_SDRAM_LDQS	3	U5	DDR SDRAM Lower Data Strobe.
FPGA_DDR_SDRAM_RASN	3	H3	DDR SDRAM Row Address Strobe (Active Low).
FPGA_DDR_SDRAM_UDM	3	R1	DDR SDRAM Upper Data Mask.
FPGA_DDR_SDRAM_UDQS	3	N4	DDR SDRAM Upper Data Strobe.
FPGA_DDR_SDRAM_WEN	3	G1	DDR SDRAM Write Enable (Active Low).
No Connect	3	R7	Unused Pin.
+1.25V DDR VREF	3	T6	DDR SDRAM Reference Voltage.

Continued on Next Page...

Xilinx Spartan 3A XC3S1400A FPGA Pin Out			
Signal Name	**Bank #**	**Pin #**	**Description**
No Connect	3	T5	Unused Pin.
No Connect	3	R8	Unused Pin.
No Connect	3	P7	Unused Pin.
No Connect	3	T3	Unused Pin.
+1.25V DDR VREF	3	T1	DDR SDRAM Reference Voltage.
No Connect	3	P8	Unused Pin.
No Connect	3	N7	Unused Pin.
No Connect	3	N5	Unused Pin.
No Connect	3	N6	Unused Pin.
No Connect	3	M5	Unused Pin.
No Connect	3	N9	Unused Pin.
No Connect	3	N8	Unused Pin.
+1.25V DDR VREF	3	N1	DDR SDRAM Reference Voltage.
No Connect	3	N3	Unused Pin.
No Connect	3	M6	Unused Pin.
No Connect	3	M7	Unused Pin.
No Connect	3	M8	Unused Pin.
No Connect	3	L7	Unused Pin.
+1.25V DDR VREF	3	J1	DDR SDRAM Reference Voltage.
No Connect	3	J3	Unused Pin.
+1.25V DDR VREF	3	L8	DDR SDRAM Reference Voltage.
No Connect	3	K7	Unused Pin.
No Connect	3	K8	Unused Pin.
No Connect	3	F2	Unused Pin.
No Connect	3	J7	Unused Pin.
No Connect	3	J5	Unused Pin.
No Connect	3	H6	Unused Pin.
No Connect	3	K6	Unused Pin.
No Connect	3	G4	Unused Pin.
No Connect	3	H5	Unused Pin.
No Connect	3	F3	Unused Pin.
+1.25V DDR VREF	3	J8	DDR SDRAM Reference Voltage.
No Connect	3	G5	Unused Pin.
No Connect	3	G6	Unused Pin.
+1.25V DDR VREF	3	H7	DDR SDRAM Reference Voltage.
No Connect	3	H8	Unused Pin.
No Connect	3	D3	Unused Pin.
No Connect	3	C2	Unused Pin.
No Connect	3	B1	Unused Pin.
No Connect	3	D2	Unused Pin.
No Connect	3	C1	Unused Pin.
No Connect	3	E4	Unused Pin.
No Connect	3	E1	Unused Pin.
No Connect	3	D1	Unused Pin.
No Connect	3	F4	Unused Pin.
No Connect	3	E3	Unused Pin.
No Connect	3	F1	Unused Pin.

A Verilog HDL Implementations

A.1 AsAP Serial Bus Slave Design Verilog HDL Code

```
/*****************************************************************

spi_slave.v module

*****************************************************************

                            CONFIDENTIAL

 Copyright (C) 2003, 2004, Regents of the University of California,

 The information contained herein is the exclusive property
 of the VCL group and shall not be distributed, reproduced,
 or disclosed in whole or in part without prior written
 permission of VCL.

 This work has been developed by members of the VLSI Computation Lab
 (VCL) in the Department of Electrical and Computer Engineering at
 the University of California at Davis. Contact: bbaas@ece.ucdavis.edu

*****************************************************************

 created on:    03/26/05
 created by:    Jeremy W. Webb
 last edit on:  02/09/07
 last edit by:  Jeremy W. Webb
 rev:           007
 comments:      Created.

                Added description, and note of what to
                investigate.

                Simulated, and verified functionality against
                the ERS.

                Corrected LATCH_PAR_DATA sensitivity list.
                Removed redundandt assignment. Re-simulated
                and verified functionality.

                Verified functionality with cfg_unpack.v.

                Changed the MISO output connection. It used to
                tie directly to MOSI input, but now it ties into
                par_data[1] to check flip-flop outputs.

                Added support for 35-bits of config data.
                ser_par_data now 20-bits wide.

*****************************************************************
```

```
 Asynchronous Array of simple Processors (AsAP)

 This module implements the serial peripheral interface (SPI)
 for use in configuring the AsAP chip.

 The interface consists of 7 signals:

 MOSI: Master Out, Slave In data input. This is data from the
       SPI Master.

 MISO: Master In, Slave Out data output. This is data sent by
       the AsAP to the SPI Master.

 SCK: Serial Clock. Only active during serial transactions.

 CSn: Active Low Chip Select. This line idles in a high state.
      The Serial Address/Data is valid on MOSI when CSn is low.

 LOAD_EN: Active High Load Enable. This line idles in a
          low state. Data sent on MOSI is latched on a
          rising edge of LOAD_EN.

 The incoming serial data is sent at 18-bit packets/symbols,
 and the parallel data is formated as follows:

  par[17:0]     par[18]         par[19]   Description
 -----------------------------------------------------------
 Upper Addr     0b0             0b0        Upper Address Word
                                           config[55:53]
 Lower Addr     0b0             0b1        Lower Address Word
                                           config[52:35]
 Upper Data     0b1             0b0        Upper Data Word
                                           config[34:18]
 Lower Data     0b1             0b1        Lower Data Word
                                           config[17:0]

 par[18] is used for determining if address or data is being
 sent. If par[18] = 0, then address is being sent. If
 par[18] = 1, then data is being sent.

 par[19] is used for determining which part of the word is
 being sent: upper or lower. If par[19] = 0, then the upper
 part of the word is being sent. If par[19] = 1, then the
 lower part of the word is being sent.

 Note the address bus of the processor configuration blocks
 only require 21-bits, so the upper 11-bits of the 16-bit
 address word are set to all zeros. The configuration blocks
 know how to interpret the data based on the status of the
 2 MSB bits of each 18-bit word. Each of the 4 serial
 transactions outlined above are sent with a configuration
 clock and a configuration valid signal to clock the parallel
 data into the AsAP processor configuration blocks.

*****************************************************************/
```

```
‘timescale 10ps/1ps
‘celldefine

module  spi_slave  (// *** Inputs ***
                    input  wire         SCK,         // SPI Clock (max: 20MHz).
                    input  wire         CSn,         // Chip Select(Active Low).
                    input  wire         MOSI,        // Master Out Slave In.
                    input  wire         LOAD_EN,     // Load Enable(Active Hi).

                    // *** Outputs ***
                    output wire         MISO,        // Master In Slave Out.
                    output wire [19:0]  ser_par_data // Parallel Data/Address
                   );

// *** Local Variable Declarations ***
// Local Wire Declarations
wire            load_n_cs;      // Latch Enable. (Active High).

// Local Register Declarations
reg     [19:0]  par_data;       // Input shift register.
reg     [19:0]  par_data_l;     // Parallel Data Latch.

// Shift In the MOSI data MSB first.
always @ (posedge SCK) // or posedge CSn)
begin : SHIFT_IN
        if (CSn)
                par_data <= 20’h0_0000;
        else
                par_data <= {par_data[18:0], MOSI};
end

// Bitwise AND of CSn and LOAD_EN.
// Only latch data when CS and LOAD_EN are high.
assign load_n_cs = LOAD_EN & CSn;

// Latch parallel data on rising edge of LOAD_EN.
always @ (posedge SCK)
begin : LATCH_PAR_DATA
        if (load_n_cs)
                par_data_l <= par_data;
end

// Assign latched data to output.
assign ser_par_data = par_data_l;

// Loopback MOSI data through MISO.
assign MISO = par_data[1];

endmodule
‘endcelldefine
```

Bibliography

[1] High-speed 16-bit 1gs/s d/a converter. http://focus.ti.com/lit/ds/symlink/dac5682z.pdf.

[2] 14-output clock generator with integrated 2.0 ghz vco. http://www.analog.com/UploadedFiles/Data_Sheets/AD9516_3.pdf.

[3] Intelligent temperature monitor and pwm fan controller. http://focus.ti.com/lit/ds/symlink/amc6821.pdf.

[4] Digital temperature sensor. http://focus.ti.com/lit/ds/symlink/tmp125.pdf.

Index