

TPCE636

Reconfigurable FPGA with 16x 16bit Analog Input and 16x 16bit Analog Output

Version 1.0

User Manual

Issue 1.0.7

May 2025

TPCE636-11R

16x Analog In, 16x Analog Out and 64 direct
FPGA Back I/O Lines,

XC7K325T-2 FBG676 Kintex™ 7 FPGA, 1GB
DDR3

This document contains information, which is
proprietary to TEWS TECHNOLOGIES GmbH. Any
reproduction without written permission is forbidden.

TEWS TECHNOLOGIES GmbH has made any effort
to ensure that this manual is accurate and complete.
However TEWS TECHNOLOGIES GmbH reserves
the right to change the product described in this
document at any time without notice.

TEWS TECHNOLOGIES GmbH is not liable for any
damage arising out of the application or use of the
device described herein.

Style Conventions

Hexadecimal characters are specified with prefix 0x,
i.e. 0x029E (that means hexadecimal value 029E).

For signals on hardware products, an ‚Active Low‘ is
represented by the signal name with # following, i.e.
IP_RESET#.

Access terms are described as:

W	Write Only
R	Read Only
R/W	Read/Write
R/C	Read/Clear
R/S	Read/Set

©2025 by TEWS TECHNOLOGIES GmbH

All trademarks mentioned are property of their respective owners.

Issue	Description	Date
1.0.0	First Issue	July 2018
1.0.1	Chapter On-Board Indicators added to the manual. Correction and extension of the JTAG chain description.	August 2018
1.0.2	Correction of the Electrical Interface specification in the Technical Specification table.	January 2019
1.0.3	Additional information about the ADC and DAC voltage ranges provided in the calibration memory.	February 2020
1.0.4	Presentation of the Connector overview renewed.	May 2021
1.0.5	Correction table Board-Status and User LEDs	October 2021
1.0.6	Feature List of TPCE636-12R corrected	September 2022
1.0.7	General Update	May 2025

Table of Contents

1	PRODUCT DESCRIPTION	9
2	TECHNICAL SPECIFICATION	10
3	HANDLING AND OPERATION INSTRUCTION	12
3.1	ESD Protection	12
3.2	Thermal Considerations	12
4	PCI DEVICE TOPOLOGY	13
4.1	User FPGA (Kintex™ 7)	14
4.2	BCC (Board Configuration Controller) FPGA	14
4.2.1	PCI Configuration Registers (PCR)	14
4.2.2	PCI BAR Overview	14
4.2.2.1	Local Configuration Register Space	15
4.2.2.2	In-System Programming Data Space	16
5	REGISTER DESCRIPTION	17
5.1	User FPGA (Kintex™ 7)	17
5.2	BCC (Board Configuration Controller) FPGA	18
5.2.1	DAC Control / Status Register – 0x00	18
5.2.2	Lspci DAC Output Voltage Range Register – 0x04	19
5.2.3	Reference DAC Voltage Control Register – 0x10 to 0x4C	20
5.2.1	User FPGA JTAG Control and Status Register – 0x80	22
5.2.2	User FPGA JTAG Signal Line Register – 0x84	23
5.2.3	User FPGA JTAG TMS Data Register - 0x88	24
5.2.4	User FPGA JTAG TDI Data Register - 0x8C	24
5.2.5	User FPGA JTAG TDO Data Register - 0x90	25
5.2.6	I2C Bridge Register - 0xA0	25
5.2.7	Interrupt Enable Register - 0xC0	25
5.2.8	Interrupt Status Register - 0xC4	26
5.2.9	User FPGA Configuration Control/Status Register - 0xD0	26
5.2.10	User FPGA Configuration Data Register - 0xD4	27
5.2.11	ISP Control Register - 0xE0	27
5.2.12	ISP Configuration Register - 0xE4	28
5.2.13	ISP Command Register - 0xE8	28
5.2.14	ISP Status Register - 0xEC	29
5.2.15	TPCE636 Temperature Sensor Register - 0xF4	30
5.2.16	TPCE636 Serial Number - 0xF8	30
5.2.17	BCC - FPGA Code Version - 0xFC	31
6	INTERRUPTS	32
6.1	Interrupt Sources	32
6.1.1	User FPGA (Kintex™ 7)	32
6.1.2	BCC (Board Configuration Controller) FPGA	32
6.2	Interrupt Handling	32
6.2.1	User FPGA (Kintex™ 7)	32
6.2.2	BCC (Board Configuration Controller) FPGA	32
7	FUNCTIONAL DESCRIPTION	33
7.1	User FPGA Block Diagram	33
7.2	User FPGA Highlights	34
7.3	User FPGA Gigabit Transceiver (MGT)	35
7.4	User FPGA Configuration	37

7.4.1	Master Serial SPI Flash Configuration.....	37
7.4.2	Manually User FPGA SPI Flash Reconfiguration	38
7.4.3	Slave Select Map Configuration	39
7.4.4	Configuration via JTAG.....	41
7.4.4.1	User JTAG Chain.....	41
7.4.4.2	TEWS Factory JTAG Chain.....	41
7.4.5	Programming User FPGA SPI Configuration Flash.....	42
7.4.6	Erasing User FPGA SPI Configuration Flash	43
7.4.7	Sector Erasing User FPGA SPI Configuration Flash.....	44
7.4.8	Reading User FPGA SPI Configuration Flash	45
7.5	BCC (Board Configuration Controller) FPGA	46
7.5.1	I2C Interface to BCC Register	46
7.6	Clocking	47
7.6.1	FPGA Clock Sources	47
7.6.2	Si514 Free Programming Clock source	49
7.7	Back I/O Interface.....	50
7.8	Memory	52
7.8.1	DDR3 SDRAM	52
7.8.2	SPI-Flash	55
7.8.3	I2C - EEPROM.....	55
7.8.3.1	I2C Calibration Data	56
7.8.3.2	ADC and DAC Calibration Data Values.....	56
7.8.3.3	DAC Calibration Data Values	57
7.8.3.4	ADC Data Correction Formula.....	58
7.8.3.5	DAC Data Correction Formula.....	58
7.8.3.6	ADC and DAC voltage ranges.....	59
7.8.3.7	Version of EEPROM data structure	60
7.9	Serial ADC Interface.....	61
7.9.1	Overview	61
7.9.2	ADC digital Output Coding.....	62
7.9.3	User FPGA Pinning.....	63
7.9.4	Programming Hints LTC2323-16	66
7.10	Parallel DAC Interface	67
7.10.1	Overview	67
7.10.2	User FPGA Pinning.....	68
7.10.3	Programming Hints for AD5547	70
7.10.4	Output Voltage Range	71
7.11	Digital Interface to FireFly Connector.....	72
7.12	JTAG Controller to K7 JTAG Interface	73
7.12.1	Bit-IO	73
7.12.2	Vector-IO.....	73
7.13	I2C Bridge	74
7.14	On-Board Indicators	75
7.14.1	User FPGA Pinning.....	76
7.15	User FPGA Reset Inputs	76
8	DESIGN HELP	77
8.1	Board Reference Design	77
9	I/O INTERFACES	78
9.1.1	Front I/O - ADC Analog Input Level	78
9.1.2	Front I/O – Analog Output Level	79
9.1.3	Back I/O Interface	79
10	I/O DESCRIPTION	80
10.1	Overview	80

10.2 Front I/O Connector (X1)	81
10.2.1 Connector Type	81
10.2.2 Pin Assignment	81
10.3 Digital Back I/O Connector (X7)	83
10.3.1 Connector Type	83
10.3.2 Pin Assignment	83
10.4 MGT Back I/O Connector (X8/X9)	85
10.4.1 Connector Type	85
10.4.2 Pin Assignment	85
10.4.3 Connector Type	86
10.4.4 Pin Assignment	86
10.5 FPGA JTAG Header (X5)	87
10.5.1 Connector Type	87
10.5.2 Pin Assignment	87
10.6 FPGA USB Connector (X2)	88
10.6.1 Connector Type	88
10.6.2 Pin Assignment	88
11 APPENDIX A	89

List of Figures

FIGURE 1-1 : BLOCK DIAGRAM	9
FIGURE 4-1 : PCIE/PCI DEVICE TOPOLOGY	13
FIGURE 5-1 : DAC AND REF. DAC SCHEMATA	20
FIGURE 5-2 : DAC OUTPUT CHANNEL	20
FIGURE 7-1 : FPGA BLOCK DIAGRAM	33
FIGURE 7-2 : MGT BLOCK DIAGRAM	35
FIGURE 7-3 : USER JTAG-CHAIN	41
FIGURE 7-4 : TEWS FACTORY JTAG-CHAIN	41
FIGURE 7-5 : FPGA CLOCK SOURCES	47
FIGURE 7-6 : ANALOG INPUT SECTION	61
FIGURE 7-7 : ANALOG INPUT BLOCK DIAGRAM	61
FIGURE 7-8 : DIGITAL ADC TO FPGA INTERFACE	66
FIGURE 7-9 : TIMING DIAGRAM LTC2323-16	66
FIGURE 7-10 : ANALOG OUTPUT SECTION	67
FIGURE 7-11 : ANALOG OUTPUT SECTION	67
FIGURE 7-12 : USER FPGA I2C TO BCC I2C BRIDGE	74
FIGURE 9-1 : DAC OUTPUT INTERFACE	79
FIGURE 10-1 : PIN ASSIGNMENT BACK I/O CONNECTOR TPCE636	84
FIGURE 10-2 : FIREFLY BACK I/O CONNECTOR TPCE636	85
FIGURE 10-3 : PIN ASSIGNMENT FIREFLY BACK I/O CONNECTOR TPCE636	85
FIGURE 10-4 : FIREFLY BACK I/O CONNECTOR TPCE636	86
FIGURE 10-5 : PIN ASSIGNMENT FIREFLY BACK I/O CONNECTOR TPCE636	86
FIGURE 10-6 JTAG HEADER TPCE636	87
FIGURE 10-7 FPGA USB CONNECTOR TPCE636	88

LIST OF TABLES

TABLE 2-1 : TECHNICAL SPECIFICATION	11
TABLE 4-1 : ON-BOARD PCIE / PCI DEVICES	14
TABLE 4-2 : PCI CONFIGURATION REGISTERS	14
TABLE 4-3 : PCI BAR OVERVIEW	14
TABLE 4-4 : LOCAL CONFIGURATION REGISTER SPACE	16
TABLE 4-5 : REGISTER BIT ACCESS TYPES	16
TABLE 5-1 : DAC CONTROL AND STATUS REGISTER	18
TABLE 5-2 : DAC OUTPUT VOLTAGE RANGE REGISTER	19
TABLE 5-3 : REFERENCE DAC VOLTAGE CONTROL REGISTER	20
TABLE 5-4 : VOLTAGE CODING FOR THE REFERENCE DAC	21
TABLE 5-5 : USER FPGA JTAG CONTROL AND STATUS REGISTER	22
TABLE 5-6 : USER FPGA JTAG SIGNAL LINE REGISTER	24
TABLE 5-7 : USER FPGA JTAG TMS DATA REGISTER	24
TABLE 5-8 : USER FPGA JTAG TDI DATA REGISTER	24
TABLE 5-9 : USER FPGA JTAG TDO DATA REGISTER	25
TABLE 5-10 : I2C BRIDGE REGISTER	25
TABLE 5-11 : INTERRUPT ENABLE REGISTER	25
TABLE 5-12 : INTERRUPT STATUS REGISTER	26
TABLE 5-13 : USER FPGA CONFIGURATION CONTROL/STATUS REGISTER	26
TABLE 5-14 : USER FPGA CONFIGURATION DATA REGISTER	27
TABLE 5-15 : ISP CONTROL REGISTER	27
TABLE 5-16 : ISP CONFIGURATION REGISTER	28
TABLE 5-17 : ISP COMMAND REGISTER	28
TABLE 5-18 : ISP STATUS REGISTER	29
TABLE 5-19 : TPCE636 TEMPERATURE SENSOR REGISTER	30
TABLE 5-20 : TPCE636 SERIAL NUMBER	30
TABLE 5-21: BCC - FPGA CODE VERSION	31
TABLE 7-1 : TPCE636 FPGA FEATURE OVERVIEW	34
TABLE 7-2 : FPGA BANK USAGE	34
TABLE 7-3 : MGT CONNECTIONS	35
TABLE 7-4 : MULTI GIGABIT TRANSCEIVER REFERENCE CLOCKS	36
TABLE 7-5: USER FPGA I2C INTERFACE TO BCC	46
TABLE 7-6 : AVAILABLE FPGA CLOCKS	48
TABLE 7-7 : FPGA I2C SI514 CONNECTIONS	49
TABLE 7-8 : DIGITAL BACK I/O INTERFACE	51
TABLE 7-9 : DDR3 SDRAM INTERFACE	53
TABLE 7-10 : FPGA SPI-FLASH CONNECTIONS	55
TABLE 7-11: FPGA I2C EEPROM CONNECTIONS	55
TABLE 7-12: ADC CALIBRATION DATA VALUES	56
TABLE 7-13: DAC CALIBRATION DATA VALUES	57
TABLE 7-13: ADC AND DAC VOLTAGE RANGES	59

TABLE 7-13: VERSION OF EEPORM DATA STRUCTURE	60
TABLE 7-14: ADC DATA CODING EXAMPLE	62
TABLE 7-15: ADC DATA CODING	62
TABLE 7-16: ADC INTERFACE CONNECTIONS	65
TABLE 7-17: TPCE636 PARALLEL DAC INTERFACE.....	69
TABLE 7-18 : FIREFLY BACK I/O INTERFACE	72
TABLE 7-19: BOARD-STATUS AND USER LEDS	75
TABLE 7-20: TPCE636 USER ON-BOARD INDICATORS	76
TABLE 7-21: USER FPGA RESET INPUTS.....	76
TABLE 9-1 : DIFFERENTIAL INPUT VOLTAGE RANGES	78
TABLE 9-2 : DAC ELECTRICAL INTERFACE.....	79
TABLE 10-1 : FRONT I/O CONNECTOR	81
TABLE 10-2: PIN ASSIGNMENT FRONT PANEL I/O CONNECTOR.....	82
TABLE 10-3 : BACK I/O CONNECTOR.....	83
TABLE 10-4: PIN ASSIGNMENT JTAG HEADER.....	87
TABLE 10-5: PIN ASSIGNMENT USB TYPE C CONNECTOR	88

1 Product Description

The TPCE636 is a half-length x4 PCIe compatible module providing a user configurable Kintex™ 7 FPGA with 16 ADC input channels and 16 DAC output channels.

The TPCE636 ADC input channels are based on the Linear Dual 16bit 5Msps Differential LTC2323-16 ADC. The TPCE636 provides 16 ADC channels. Each of the 16 channels has a resolution of 16bit and can work with up to 5Msps. The analog input circuit is designed to allow input voltages of up to $\pm 10V$ on each input-pin (results in $\pm 20V$ differential voltage range)

The TPCE636 DAC output channels are based on the Dual 16bit AD5547 DAC. Each DAC output is designed as a single-ended bipolar $\pm 10V$ analog output.

For customer specific I/O extension or inter-board communication, the TPCE636 provides 64 FPGA I/Os on a back I/O connector and 4 FPGA Multi-Gigabit-Transceivers on a Samtec Firefly connector. Digital back I/O lines can be configured as 64 single-ended LVCMOS25 lines or as 32 differential LVDS25 lines.

The User FPGA is connected to a 1GB, 32bit wide DDR3 SDRAM. The SDRAM-interface uses an internal Memory Controller of the Kintex™ 7.

The User FPGA is configured by a serial SPI flash. For full PCIe specification compliance, the XILINX Tandem Configuration Feature can be used for FPGA configuration. XILINX Tandem Methodologies “Tandem PROM” should be the preferred methodology. The SPI flash device is in-system programmable. An in-circuit debugging option is available via a JTAG header for read back and real-time debugging of the FPGA design (using Xilinx “ChipScope”).

User applications for the TPCE636 with Kintex™ 7 FPGA can be developed using the design software Vivado Design Suite. A license for the Vivado Design Suite design tool is required.

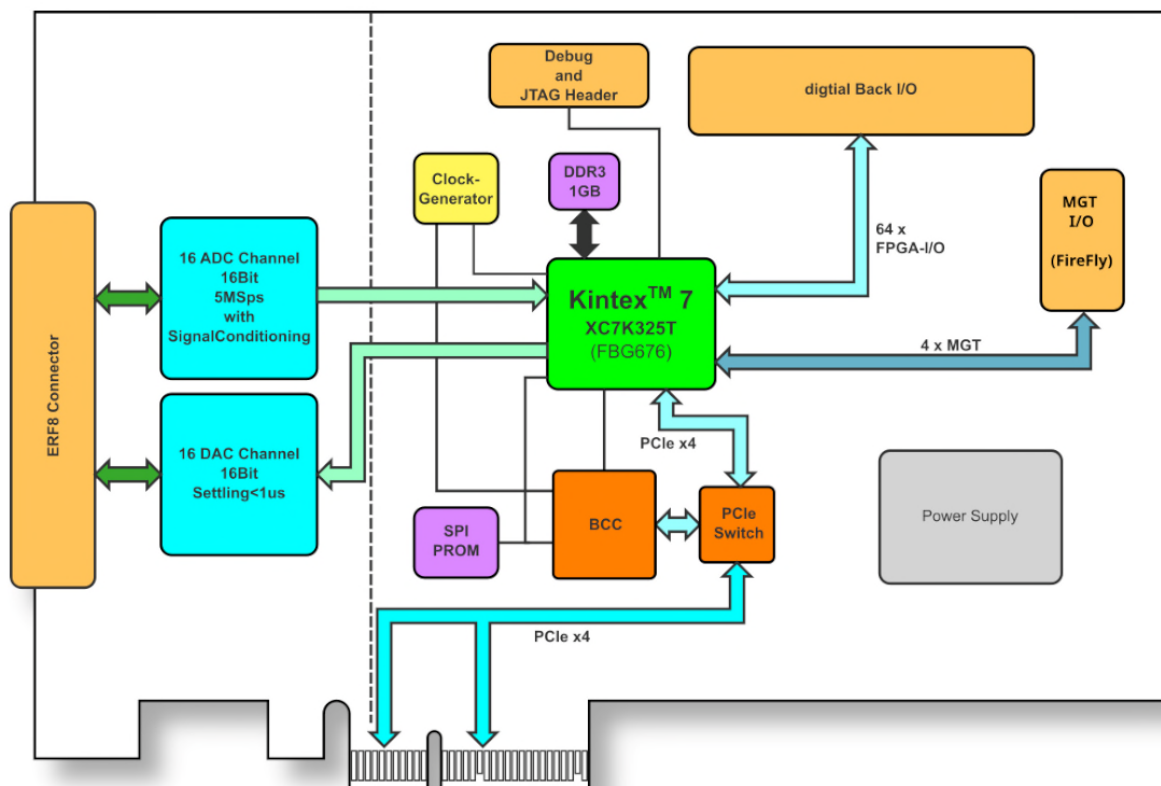


Figure 1-1 : Block Diagram

2 Technical Specification

PCIe Interface		
Mechanical Interface	Peripheral Component Interconnect Express (PCIe) PCI Express CEM R3.0 Standard Height, shortened Half Length PCI Express Add-in Card (98.4mm x 153.50mm)	
Electrical Interface	PCI Express x4 Link (Base Specification 2.1) compliant interface conforming to ANSI/VITA 42.3-2006 (PCI Express Protocol Layer Standard)	
On-Board Devices		
PCI Express Switch	PI7C9X2G312GP (Pericom)	
PCI Express to PCI Bridge	XIO2001 (Texas Instruments)	
User configurable FPGA	TPCE636-11R: XC7K325T-2FBG676I (Xilinx) Further Kintex™ 7 FPGA options on request	
SPI-Flash	MT25QL128 (Micron) 128Mbit (contains TPCE636 FPGA BRD) or compatible; +3.3V supply voltage	
DDR3 RAM	2 x MT41K256M16TW-107 (Micron) 256Meg x 32Bit	
Board Configuration Controller	LCMXO2-7000HC (Lattice)	
ADC	LTC2323IUFD-16 (Linear Technologies)	
DAC	AD5547BRUZ (Analog Devices)	
I/O Interface		
Number of analog Input	16 differential 16bit Inputs	
Analog Input Voltage	diff. VIN _{MAX} (allowed voltage between input pins): Common Mode Voltage Range: Input Voltage limit for each pin relative to common ground:	±20.0V 0.0V ±10V
Number of analog Outputs	16 single-ended 16bit Outputs	
Analog Output Voltage	Maximum single-ended Output Voltage – V _{out} Maximum Output Drive Current for each Output Maximum Capacitive Load for each Output Typical Settling Time for a 10mA / 1000pF	±10V 10mA 1000pF < 1µs
Number of digital Back I/O	64 direct FPGA I/O lines to P14 Back I/O connector - Can be used as single-ended or differential I/O - FPGA I/O Standard: LVCMOS25, LVTTTL25 and LVDS25	
I/O Connector	Front I/O Samtec – ERF8-049-01-L-D-RA-L Back I/O Connector male SMC-B 68 (ERNI 154766 64) High Speed Back I/O Connector (Samtec FireFly™)	
Physical Data		
Power Requirements	Depends on FPGA design With TPCE636 Board Reference Design / without external load	
		typical @ +12V PCIe
	TPCE636-xxR	1.3 A
		typical @ +3.3V PCIe
		not used by TPCE636

Temperature Range	Operating Storage	-40°C to +85°C -40°C to +85°C
MTBF	TPCE636-xxR: 240000 h MTBF values shown are based on calculation according to MIL-HDBK-217F and MIL-HDBK-217F Notice 2; Environment: G _B 20°C. The MTBF calculation is based on component FIT rates provided by the component suppliers. If FIT rates are not available, MIL-HDBK-217F and MIL-HDBK-217F Notice 2 formulas are used for FIT rate calculation.	
Humidity	5 – 95 % non-condensing	
Weight	TPCE636-xxR: 208g	

Table 2-1 : Technical Specification

3 Handling and Operation Instruction

3.1 ESD Protection



The TPCE636 is sensitive to static electricity. Packing, unpacking and all other handling of the TPCE636 has to be done in an ESD/EOS protected Area.

3.2 Thermal Considerations



Forced air cooling is recommended during operation. Without forced air cooling, damage to the device can occur.

Please also note chapter “Thermal Management”.

4 PCI Device Topology

The TPCE636 consists of two FPGAs. Both FPGAs are designed as PCIe / PCI endpoint devices. One FPGA is the User FPGA (Kintex™ 7) which can be programmed with user defined FPGA code. The second FPGA takes control of on-board hardware functions of TPCE636 and also the configuration control of the User FPGA. This second FPGA is the BCC (Board Configuration Controller).

The BCC PCI endpoint is connected via a PCI-to-PCIe Bridge to the second x1 Downstream Port of the PCIe Switch (Pericom PI7C9X2G312GP). The User FPGA (Kintex™ 7 PCIe endpoint) is directly connected to the first x4 Downstream Port.

The x4 Upstream Port of the PCIe Switch is connected to the XMC P15 Connector, communicating with the host system.

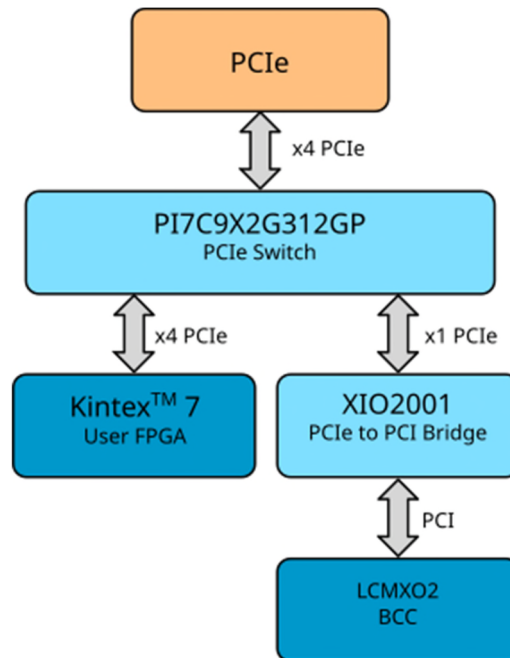


Figure 4-1 : PCIe/PCI Device Topology

Device	Vendor ID	Device ID	Class Code	Description (as shown by lspci)
PI7C9X2G312GP	0x12D8 (Pericom)	0x2312	0x060400	PCI bridge: 0x04h to indicate device as PCI-to-PCI Bridge 0x06h to indicate device as Bridge device
XIO2001	0x104C (Texas Instruments)	0x8240	0x060400	PCI bridge: Texas Instruments 0x04h to indicate device as PCI-to-PCI Bridge 0x06h to indicate device as Bridge device
XC7KxxxT-2	user defined			Device identification for the User programmable FPGA is defined by user. The data will be created with the Xilinx Vivado "7 Series Integrated Block for PCI Express" IP generator.
BCC	0x1498	0x727c	0x068000	Bridge Device: TEWS Technologies GmbH Device 927E

LCMX02	(TEWS)			(TPCE636).
--------	--------	--	--	------------

Table 4-1 : On-Board PCIe / PCI Devices

4.1 User FPGA (Kintex™ 7)

The User FPGA address map depends on the user application and is not part of this specification.

4.2 BCC (Board Configuration Controller) FPGA

4.2.1 PCI Configuration Registers (PCR)

PCI CFG Register Address	Write '0' to all unused (Reserved) bits							PCI writeable	Initial Values (Hex Values)	
	31	24	23	16	15	8	7			0
0x00	Device ID				Vendor ID				N	727C 1498
0x04	Status				Command				Y	0480 000B
0x08	Class Code					Revision ID			N	068000 01
0x0C	BIST		Header Type		PCI Latency Timer		Cache Line Size		Y[7:0]	00 00 00 08
0x10	PCI Base Address 0 for Local Address Space 0							Y	FFFFFF00	
0x14	PCI Base Address 1 for Local Address Space 1							Y	FFFFFF00	
0x18	PCI Base Address 2 for Local Address Space 2							N	00000000	
0x1C	PCI Base Address 3 for Local Address Space 3							N	00000000	
0x20	PCI Base Address 4 for Local Address Space 4							N	00000000	
0x24	PCI Base Address 5 for Local Address Space 5							N	00000000	
0x28	PCI CardBus Information Structure Pointer							N	00000000	
0x2C	Subsystem ID				Subsystem Vendor ID				N	727C 1498
0x30	PCI Base Address for Local Expansion ROM							Y	00000000	
0x34	Reserved					New Cap. Ptr.			N	000000 40
0x38	Reserved							N	00000000	
0x3C	Max_Lat		Min_Gnt		Interrupt Pin		Interrupt Line		Y[7:0]	00 00 01 00

Table 4-2 : PCI Configuration Registers

4.2.2 PCI BAR Overview

BAR	Size (Byte)	Space	Prefetch	Port Width (Bit)	Endian Mode	Description
0	256	MEM	No	32	Little	Local Configuration Register Space
1	256	MEM	No	32	Little	In-System Programming Data Space

Table 4-3 : PCI BAR Overview

4.2.2.1 Local Configuration Register Space

Offset to PCI Base Address	Register Name	Size (Bit)
0x00	DAC Control and Status Register	32
0x04	DAC Output Voltage Range Register	32
0x08 – 0x0C	Reserved	-
0x10	Register for VOFF and VREF of DAC Channel 1	32
0x14	Register for VOFF and VREF of DAC Channel 2	32
0x18	Register for VOFF and VREF of DAC Channel 3	32
0x1C	Register for VOFF and VREF of DAC Channel 4	32
0x20	Register for VOFF and VREF of DAC Channel 5	32
0x24	Register for VOFF and VREF of DAC Channel 6	32
0x28	Register for VOFF and VREF of DAC Channel 7	32
0x2C	Register for VOFF and VREF of DAC Channel 8	32
0x30	Register for VOFF and VREF of DAC Channel 9	32
0x34	Register for VOFF and VREF of DAC Channel 10	32
0x38	Register for VOFF and VREF of DAC Channel 11	32
0x3C	Register for VOFF and VREF of DAC Channel 12	32
0x40	Register for VOFF and VREF of DAC Channel 13	32
0x44	Register for VOFF and VREF of DAC Channel 14	32
0x48	Register for VOFF and VREF of DAC Channel 15	32
0x4C	Register for VOFF and VREF of DAC Channel 16	32
0x50 – 0x7F	Reserved	-
0x80	Kintex™ 7 JTAG Control Register	32
0x84	Kintex™ 7 JTAG Interface Register	32
0x88	Kintex™ 7 JTAG Signal TMS Data Register	32
0x8C	Kintex™ 7 JTAG Signal TDI Data Register	32
0x90	Kintex™ 7 JTAG Signal TDO Data Register	32
0x94 – 0x9F	Reserved	-
0xA0	I2C Bridge Register	32
0xA4 – 0xBF	Reserved	-
0xC0	Interrupt Enable Register	32
0xC4	Interrupt Status Register	32
0xC8	Reserved	-
0xCC	Reserved	-
0xD0	User FPGA Configuration Control/Status Register	32
0xD4	User FPGA Configuration Data Register (Slave SelectMAP)	32
0xD8	Reserved	-
0xDC	Reserved	-
0xE0	ISP Control Register (SPI)	32
0xE4	ISP Configuration Register (SPI)	32

0xE8	ISP Command Register (SPI)	32
0xEC	ISP Status Register (SPI)	32
0xF0	Reserved	-
0xF4	TPCE636 Board Temperature	32
0xF8	TPCE636 Serial Number	32
0xFC	BCC Code Version	32

Table 4-4 : Local Configuration Register Space

Register Bit Access Type		Description
R	Read	The bit is readable by software.
R/W	Read/Write	The bit is readable and writeable by software.
R/C	Read/Clear	The bit is readable by software. The bit is set by firmware. Software may clear the bit by writing a '1'.
R/S	Read/Set	The bit is readable by software. Software may set this bit to '1'. The bit is cleared by firmware.

Table 4-5 : Register Bit Access Types

When reading reserved register bits, the value is undefined.

Reserved register bits shall be written as '0'.

4.2.2.2 In-System Programming Data Space

The In-System Programming (ISP) Data Space is used for passing user FPGA configuration data for in-system programming of the User FPGA SPI Flash.

For ISP write/program instructions, the data must be written (zero-based) to the ISP Data Space before the instruction is started. The data must cover a complete SPI Flash memory page.

For ISP read instructions, the data can be read (zero-based) from the ISP Data Space after the instruction is done. The data is passed for a complete SPI Flash memory page.

The ISP Data Space size is 256byte, covering an SPI Flash Memory Page. All supported SPI Flash read and write instructions are page-based.

Control and status register for ISP are located in the Local Configuration Register Space. The data register for direct FPGA ISP is also located in the Local Configuration Register Space.

5 Register Description

5.1 User FPGA (Kintex™ 7)

The FPGA register description depends on the user application and is not part of this specification.

5.2 BCC (Board Configuration Controller) FPGA

5.2.1 DAC Control / Status Register – 0x00

The output voltage ranges of the TPCE636 DAC outputs are set via *DAC Control / Status Register* and *DAC Output Voltage Range Register*.

For the three predefined ranges ($\pm 10V$, $\pm 5V$ and $\pm 2.5V$), the *DAC Output Voltage Range Register* is set first and then the values are transferred via the *DAC Control / Status Register*.

For the individual range selection the *Reference DAC Voltage Control Register* must also be set before updating the data via *DAC Control / Status Register*.

Bit	Symbol	Description	Access	Reset Value
31:16	-	Reserved	R	0
15:14	-	Reserved	R	0
13	DAC2_OUTP_UPD	Bits correspond to DAC #1 Refer description for DAC #1		
12	DAC2_CLR			
11	DAC2_OVRTMP			
10	-			
9	DAC2_OUTPSTTLE			
8	DAC2_IOBSY			
7:6	-	Reserved	R	0
5	DAC1_OUTP_UPD	DAC #1 Output Update Initiate DAC Device Output Update with currently adjusted output values Self-Clearing	R/S	0
4	DAC1_CLR	DAC #1 Clear Causes DAC to be cleared. Includes setup with current adjusted values Self-Clearing	R/S	0
3	DAC1_OVRTMP	DAC #1 Over Temperature Status is provided by DAC device	R	0
2	-	Reserved		
1	DAC1_OUTPSTTLE	DAC #1 Output Settle Internal generated settling pulse (set to 20.5us)	R	0
0	DAC1_IOBSY	DAC #1 I/O Busy Indicates that <ul style="list-style-type: none"> Controller is not in idle Span-Configuration update is pending Output Value update is pending Internal synchronization is pending 	R	0

Table 5-1 : DAC Control and Status Register

5.2.2 Lspci DAC Output Voltage Range Register – 0x04

Bit	Symbol	Description	Access	Reset Value
31:30	REF_DAC16	DAC Output Voltage range selection for DAC channel 16 00 : $\pm 10V$ range (default value) 01 : $\pm 5V$ range 10 : $\pm 2.5V$ range 11 : individual range selection	R/W	0b00
...	-
1:0	REF_DAC1	DAC Output Voltage range selection for DAC channel 1 00 : $\pm 10V$ range (default value) 01 : $\pm 5V$ range 10 : $\pm 2.5V$ range 11 : individual range selection	R/W	0b00

Table 5-2 : DAC Output Voltage Range Register

For individual voltage range selection also see the chapter 5.2.3 *Reference DAC Voltage Control Register*.

5.2.3 .Reference DAC Voltage Control Register – 0x10 to 0x4C

The 16 TPCE636 DAC outputs consist of eight dual DAC devices (AD5547). Each of these DACs could be used independently.

For the generation of the reference voltage of these 16 TPCE636 DACs, two additional serial DAC devices are placed on the TPCE636. Each of this Reference DAC has 16 analog outputs. Thus 32 reference voltages are available. Two reference voltages for each TPCE636 DAC output.

Via these reference voltages (VREF and VOFF), an individual voltage range can be set for each TPCE636 DAC channel. Both reference voltages are each set as 16 bit value via a 32bit register of the BCC. This results in 16 32bit registers via which the reference voltage and thus the output voltage range of the TPCE636 can be set.

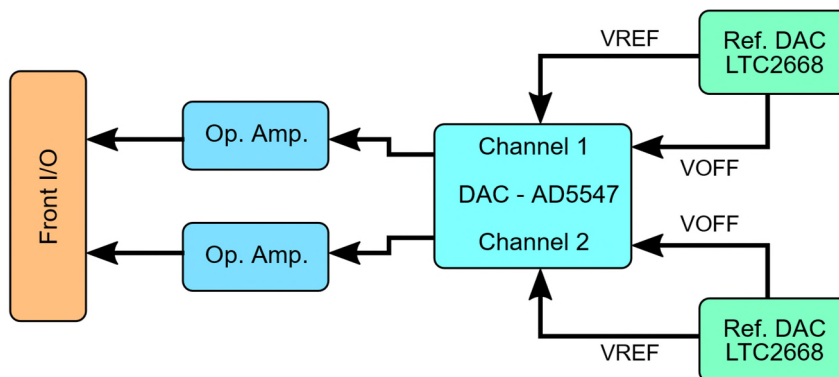


Figure 5-1 : DAC and Ref. DAC Schemata

One 32bit Register for each of the 16 TPCE636 DAC outputs.

Bit	Symbol	Description	Access	Reset Value
31:16	VREF	16-bit Data Value for VREF which specifies the half voltage swing of the reference voltage for the AD5547 DAC channel.	R/W	0x8000
15:0	VOFF	16-bit Data Value for VOFF which represents the negative reference voltage of the AD5547 DAC channel.	R/W	0x8000

Table 5-3 : Reference DAC Voltage Control Register

Each TPCE636 analog output consists of a half AD5547 output DAC, an operational amplifier driving the output load, and two references DAC outputs each for VREF and VOFF.

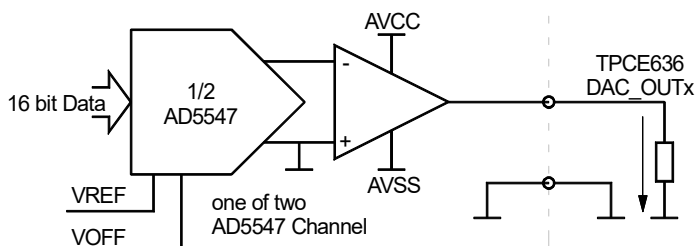


Figure 5-2 : DAC Output Channel

The following table shows the data coding for VREF and VOFF output voltage range.

Description	Digital Code
+9.9997V	0xFFFF
+5.0V	0xC000
+2.5V	0xA000
0V	0x8000
-2.5V	0x6000
-5.0V	0x4000
-10.0V	0x0000

Table 5-4 : Voltage coding for the Reference DAC

The output voltage range is then calculated with these two reference values (VREF and VOFF) as follows:

$$V_{range_high} = -1 * V_{OFF}$$

$$V_{range_low} = -1 * V_{OFF} - 2 * V_{REF}$$

See the following Examples for:

VOFF	VREF	Vrange_low	Vrange_high	Voltage Range	Note
+10V (0xFFFF)	-10V (0x0000)	-10V	+10V	-10V ... +10V	typical range
+5V (0xC000)	-5V (0x4000)	-5V	+5V	-5V ... +5V	typical range
+1.25V (0x9000)	-5V (0x4000)	-1.25V	+8.75V	-1.25V ... +8.75V	asymmetric range
-5V (0x4000)	+5V (0xC000)	+5V	-5V	+5V ... -5V	changes sign

Thus, any desired output voltages can be generated. But in any case it is important to ensure that the voltages Vout1 and Vout2 are never set above +10V or below -10V. Voltages above these limits cannot be generated and lead to incorrect outputs.

5.2.1 User FPGA JTAG Control and Status Register – 0x80

Bit	Symbol	Description	Access	Reset Value	
31:10	-	Reserved	-	00	
9:8	JTAG_VIO_CTRL_STAT	JTAG Vector-I/O Controller Status Signals the status of the Vector I/O controller. 0b00 = Controller disabled 0b01 = Controller idle (read-for-request) 0b10 = Ongoing Vector I/O controller operation 0b11 = illegal <i>Initiate JTAG operations only if the Vector I/O controller is in idle state.</i>	r	01	
7:5	-	Reserved	-	0	
4	JTAG_EXT#_INT	JTAG External/Internal Indicates whether a JTAG device is detected/present at the Debug Connector or not. 0b0 = Device detected/present 0b1 = No device detected <i>Detection is based on the Debug Connector Present signal</i>	r	-	
3	-	Reserved	-	0	
2:0	JTAG_MUX	BCC controlled JTAG Multiplexer to USER FPGA		r/w	0b000
		0b000	Fixed priority. Which JTAG connection is active depends on the inserted cable. <i>The priority is as follows:</i> 1) Debug Connector 2) USB JTAG 3) BCC JTAG (V-I/O or B-I/O)		
		0b001	Hardwired JTAG interface from Debug Connector to Kintex™ 7 is active.		
		0b010	USB to JTAG Interface via FTDI2232 is connected to Kintex™ 7.		
		0b100	BCC internal JTAG Controller is connected to Kintex™ 7.		

Table 5-5 : User FPGA JTAG Control and Status Register

5.2.2 User FPGA JTAG Signal Line Register – 0x84

Refer to the chapter JTAG Controller to K7 JTAG Interface for a functional description.

Bit	Symbol	Description	Access	Reset Value
31:24	JTAG_VIO_TCK_CLK_DIV	<p>JTAG Vector-I/O TCK Clock Divider</p> <p>Divider applied onto the internal processing clock to obtain/generate the JTAG TCK I/O clock</p> <p>$JTAG_TCK = 53.2MHz / (divider + 1)$</p> <p><i>Default value results in an 1.6625 MHz TCK frequency</i></p>	r/w	0x1F
23:16	JTAG_VIO_XFER_LEN	<p>JTAG Vector-I/O Transfer Length</p> <p>Sets the number of transfer cycles (TCK clocks) performed during a Vector-I/O operation</p> <p><i>Default value causes 256 full clock cycles (rising and falling edges) to be emitted on TCK</i></p>	r/w	0xFF
15:14	-	Reserved	-	0
13	JTAG_VIO_SHIFT_REQ	<p>JTAG Vector-I/O Shift Request</p> <p>Initiates a TMS/TDI shift-out operation (incl. TDO read-in) operation</p> <p><i>TMS/TDI are coupled. Both vectors are shifted out at the same time.</i></p> <p><i>The TMS/TDI data interface registers limit the transfer length</i></p>	r/s	0
12	JTAG_VIO_GET_REQ	<p>JTAG Vector-I/O Get Request</p> <p>Initiates a TDO read-in only operation</p> <p><i>The TDO data interface register limits the transfer length if all read-in data is required</i></p>	r/s	0
11:9	-	Reserved	-	0
8	JTAG_VIO_EN-	<p>JTAG Vector-I/O Enable</p> <p>Controls the state of the BCC JTAG Vector-I/O Interface.</p> <p>0b0 = Vector-I/O disabled</p> <p>0b1 = Vector-I/O enabled</p> <p><i>Functionality is held in reset until enabled is set</i></p>	r/w	0

7	JTAG_BIO_TDO	JTAG Bit-I/O - TDO TDO line output state of the User FPGA	r	-
6	JTAG_BIO_TDI	JTAG Bit-I/O - TDI Output state for the User FPGA TDI input line	r/w	1
5	JTAG_BIO_TMS	JTAG Bit-I/O - TMS Output state for the User FPGA TMS input line	r/w	1
4	JTAG_BIO_TCK	JTAG Bit-I/O - TCK Output state for the User FPGA TCK input line	r/w	0
3:1	-	Reserved	-	0
0	JTAG_BIO_EN	JTAG Bit-I/O Enable Controls the state of the BCC JTAG Bit-I/O Interface. 0b0 = Bit-I/O disabled 0b1 = Bit-I/O enabled	r/w	0

Table 5-6 : User FPGA JTAG Signal Line Register

5.2.3 User FPGA JTAG TMS Data Register - 0x88

Bit	Symbol	Description	Access	Reset Value
31:0	JTAG_VIO_TMS_DATA	JTAG Vector-I/O TMS Data Sets the JTAG TMS bit data that is shifted-out during TMS/TDI shift operations. <i>Note: Bit 0 is shifted-out first (right-alignment)</i>	r/w	0

Table 5-7 : User FPGA JTAG TMS Data Register

5.2.4 User FPGA JTAG TDI Data Register - 0x8C

Bit	Symbol	Description	Access	Reset Value
31:0	JTAG_VIO_TDI_DATA	JTAG Vector-I/O TDI Data Sets the JTAG TDI bit data that is shifted-out during TMS/TDI shift operations. <i>Note: Bit 0 is shifted-out first (right-alignment)</i>	r/w	0

Table 5-8 : User FPGA JTAG TDI Data Register

5.2.5 User FPGA JTAG TDO Data Register - 0x90

Bit	Symbol	Description	Access	Reset Value
31:0	JTAG_VIO_TDO_DATA	JTAG Vector-I/O TDO Data Accumulates TDO bit data read-in during TMS/TDI shift and TDO get request operations <i>Note: Bit 0 is shifted-in last (right-alignment)</i>	r/w	0

Table 5-9 : User FPGA JTAG TDO Data Register

5.2.6 I2C Bridge Register - 0xA0

Refer to the chapter I2C Bridge for a functional description.

Bit	Symbol	Description	Access	Reset Value
31:1	-	-	-	0
0	I2C_BRDG_MODE_EN	I2C Bridge Mode Enable Controls the USER I2C Bus BCC reach-through onto the management I2C Bus '0' = disabled '1' = enabled	r/w	0

Table 5-10 : I2C Bridge Register

5.2.7 Interrupt Enable Register - 0xC0

Bit	Symbol	Description	Access	Reset Value
31:2		Reserved		0
1	ISP_INS_IE	ISP SPI Instruction Done Event Interrupt Enable 0: Interrupt Disabled 1: Interrupt Enabled While disabled, the corresponding bit in the Interrupt Status Register is '0'. Disabling interrupts does not affect the interrupt source.	R/W	0
0	ISP_DAT_IE	ISP SPI Page Data Request Event Interrupt Enable 0: Interrupt Disabled 1: Interrupt Enabled While disabled, the corresponding bit in the Interrupt Status Register is '0'. Disabling interrupts does not affect the interrupt source.	R/W	0

Table 5-11 : Interrupt Enable Register

5.2.8 Interrupt Status Register - 0xC4

Bit	Symbol	Description	Access	Reset Value
31:2		Reserved		0
1	ISP_INS_IS	ISP SPI Instruction Done Event Interrupt Status When set, the PCI INTA# interrupt is asserted. The Interrupt is cleared by writing a '1'. 0: Interrupt not active or disabled 1: Interrupt active and enabled	R/C	0
0	ISP_DAT_IS	ISP SPI Page Data Done Event Interrupt Status When set, the PCI INTA# interrupt is asserted. The Interrupt is cleared by writing a '1'. 0: Interrupt not active or disabled 1: Interrupt active and enabled	R/C	0

Table 5-12 : Interrupt Status Register

5.2.9 User FPGA Configuration Control/Status Register - 0xD0

Bit	Symbol	Description	Access	Reset Value
31:5		Reserved		0
4	K7_LINK_ENA	1: Kintex™ 7 to PCIe-Switch LINK is enabled 0: Kintex™ 7 to PCIe-Switch LINK is disabled	R/W	1
3	FP_INIT_STAT	User FPGA INIT_B Pin Status 0: FPGA INIT_B Pin Level is Low (active) 1: FPGA INIT_B Pin Level is High (not active)	R	x
2	FP_DONE_STAT	User FPGA DONE Pin Status The FPGA Done pin is high in case of successful FPGA configuration. 0: FPGA DONE Pin Level is Low (not active) 1: FPGA DONE Pin Level is High (active)	R	x
1	FP_RE_CFG	After power-up the FPGA automatically configures from the on-board SPI Flash in 'Master Serial / SPI' mode. User FPGA Re-Configuration 1: Set all FPGA I/O pins to High-Z and prepare a User FPGA Re-Configuration 1 → 0: Start User FPGA Re-Configuration	R/W	0
0	FP_CFG_MD	Set User FPGA Configuration Mode 0: Master Serial / SPI 1: Slave SelectMap (Parallel) After power-up the User FPGA automatically configures from the on-board SPI Flash in 'Master Serial / SPI' mode.	R/W	0

Table 5-13 : User FPGA Configuration Control/Status Register

5.2.10 User FPGA Configuration Data Register - 0xD4

Bit	Symbol	Description	Access	Reset Value
31:0	ISP_FP_DAT	ISP Select Map Write Data Write Data Register for direct Slave Select Map FPGA programming mode Must be written with 32-bit FPGA programming data until the FPGA Done pin goes high (after the actual programming data, writing some dummy data may be required).	W	-

Table 5-14 : User FPGA Configuration Data Register

The User FPGA Configuration Data Register is used to write data within the User FPGA Slave Select Map Configuration directly to the User FPGA.

5.2.11 ISP Control Register - 0xE0

Bit	Symbol	Description	Access	Reset Value
31:1		Reserved		0
0	ISP_EN	ISP Mode Enable 0: Disable ISP Mode 1: Enable ISP Mode This bit controls the BCC interface between BCC, SPI-Flash and the User FPGA (Kintex™ 7). When set, the BCC is both SPI Flash Master and FPGA Configuration Interface Master. Must be set to 1 for direct Slave Select Map mode or SPI Flash programming. Must be set to 0 when the User FPGA should configure from the SPI Flash (e.g. after SPI Flash programming) in 'Master Serial / SPI' mode. Note, that for ISP Direct FPGA Programming, the FPGA must first be set to Slave Select Map configuration mode.	R/W	0

Table 5-15 : ISP Control Register

5.2.12 ISP Configuration Register - 0xE4

Bit	Symbol	Description	Access	Reset Value
31:24	ISP_SPI_ADD	SPI Flash Address A7-A0	W	0x00
23:16		SPI Flash Address A15-A8	W	0x00
15:8		SPI Flash Address A23-A16	W	0x00
7:0	ISP_SPI_INS	SPI Flash Instruction Code Supported Instructions: 0x02 – Page Program 0x20 – Sector Erase 0xC7 – Chip Erase 0x03 – Read Data	W	0x00

Table 5-16 : ISP Configuration Register

5.2.13 ISP Command Register - 0xE8

Bit	Symbol	Description	Access	Reset Value
31:2		Reserved	-	0
1	ISP_SPI_RST_CMD	ISP SPI Reset Command Bit Writing a '1' sets the Instruction Busy Bit in the ISP Status Register (if not already set). Breaks any ISP SPI instruction in progress and resets the ISP SPI logic. Check the Instruction Busy Bit in the ISP Status Register for reset done status. Always read as '0'.	R/W	0
0	ISP_SPI_INS_CMD	ISP SPI Start Instruction Command Bit Writing a '1' sets the SPI Instruction Busy Bit in the ISP Status Register and starts the configured SPI instruction. Ignored (lost) while the Instruction Busy Bit is set in the ISP Status Register. Always read as '0'.	R/W	0

Table 5-17 : ISP Command Register

5.2.14 ISP Status Register - 0xEC

Bit	Symbol	Description	Access	Reset Value
31:2		Reserved	-	0x00_0000
1	ISP_SPI_INS_BSY	ISP SPI Instruction Busy Status Set & Cleared automatically by HW. Includes SPI Flash internal program/erase times. When clear again after being set, a new ISP SPI instruction may be started. Capable of generating an event based interrupt. 0: No ISP SPI Instruction in Progress 1: ISP SPI Instruction in Progress	R	0
0	ISP_SPI_DAT_BSY	ISP SPI Data Transfer Busy Status Set & Cleared automatically by HW. Does not include SPI Flash internal program/erase times. When clear again after being set, new SPI Flash page data may be written to the ISP Data Space (in program mode) or SPI Flash page data is available in the ISP data space (in read mode). Capable of generating an event based interrupt. 0: No ISP SPI Data Transfer in Progress 1: ISP SPI Data Transfer in Progress	R	0

Table 5-18 : ISP Status Register

5.2.15 TPCE636 Temperature Sensor Register - 0xF4

Bit	Symbol	Description	Access	Reset Value
31:21	-	Reserved	r	-
20	TMP441_AUTO_TRD_EN	TMP441 Automatic Temperature Read Enable Controls the periodic board temperature read feature. Refresh time = 1s '0' = disabled '1' = enabled <i>Automatic mode must be disabled before enabling the I2C bridge mode</i>	r/w	1
19:16	-	Reserved		0
15:8	TMP441_TEMP	TMP441 Temperature Data Measured data of the on-board temperature sensor The read value of the temperature sensor is stored sign-extended as a 8bit two's complement. To actually calculate the temperature from the two's complement data value, use the following formula: Temperature (°C) = TEMP	r	-
7:0	-	Reserved	-	-

Table 5-19 : TPCE636 Temperature Sensor Register

5.2.16 TPCE636 Serial Number - 0xF8

Bit	Symbol	Description	Access	Reset Value
31:0	S_NUMBER	The value is the unique serial number of each TPCE636 module	R	-

Table 5-20 : TPCE636 Serial Number

Example: 0x0091_DB0E => SNo.: 9558798

The serial number can also be read via an I2C interface from User FPGA (Kintex™ 7).

5.2.17 BCC - FPGA Code Version - 0xFC

Bit	Symbol	Description	Access	Reset Value
31:0	CODE_VER	The value shows the BCC Firmware code version of the TPCE636 module.	R	-

Table 5-21: BCC - FPGA Code Version

Example:

0x0100_0A00 => bit 32 downto 24 : Major FPGA Code Version

0x0100_0A00 => bit 23 downto 16 : Minor FPGA Code Version

0x0100_0A00 => bit 15 downto 08 : FPGA Code Revision

0x0100_0A00 => bit 07 downto 00 : FPGA Code Build Number

6 Interrupts

6.1 Interrupt Sources

6.1.1 User FPGA (Kintex™ 7)

The FPGA interrupt sources depend on the user application and are not part of this specification.

6.1.2 BCC (Board Configuration Controller) FPGA

The BCC - FPGA provides two interrupt sources. Both interrupts are only available during SPI programming instructions. The Slave Select Map Mode does not provide interrupt support.

- **ISP SPI Instruction Done Event Interrupt**

Event-based interrupt that becomes active, when the ISP SPI Instruction Busy status bit changes from busy to not-busy.

- **ISP SPI Page Data Done Event Interrupt**

Event-based interrupt that becomes active, when the ISP SPI Data Busy status bit changes from busy to not-busy.

6.2 Interrupt Handling

6.2.1 User FPGA (Kintex™ 7)

The interrupt handling depends on the user application and is not part of this specification.

6.2.2 BCC (Board Configuration Controller) FPGA

Both Interrupts of the BCC FPGA must be cleared via writing access to the corresponding Interrupt Status Flag in the Interrupt Status Register.

7 Functional Description

7.1 User FPGA Block Diagram

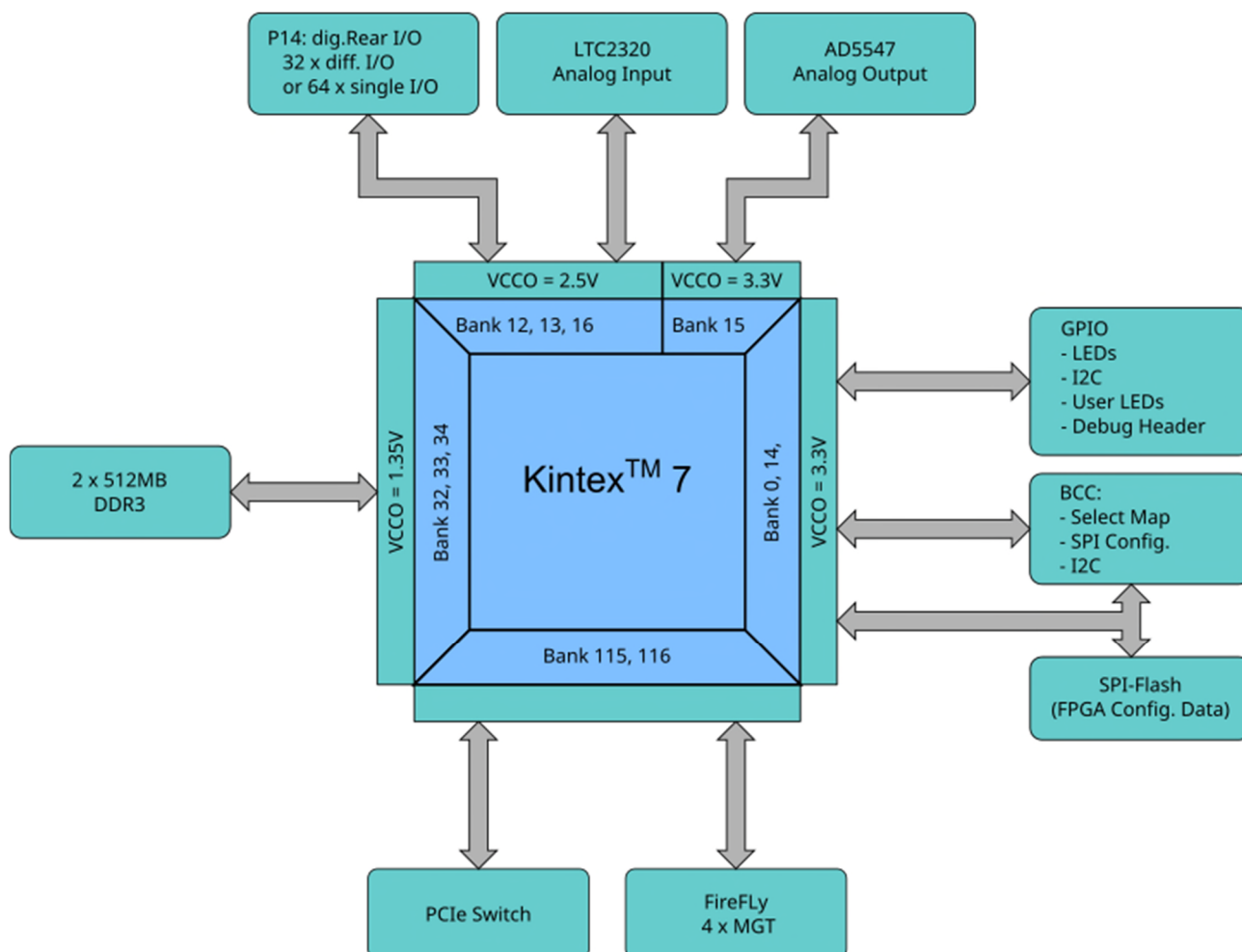


Figure 7-1 : FPGA Block Diagram

7.2 User FPGA Highlights

The FPGA is a Kintex™ XC7K325T FPGA. Each Kintex™ 7 FPGA in a FBG676 package provides eight MGT, four for high speed back I/O communication and four for the PCI Express interface (x4 Linkage).

Kintex™ 7	Logic Cells	Slices	DSP Slices	Block RAM (Kb)			CMTs	MGT	XADC Block
				18 Kb	36 Kb	max(Kb)			
XC7K160T	162,240	25.350	600	650	325	11,700	8	8	1
XC7K325T	326,080	50.950	840	890	445	16,020	10	8	1
XC7K410T	406,720	63.550	1540	1590	795	28,620	10	8	1

Table 7-1 : TPCE636 FPGA Feature Overview

PCI Express Highlights:

- Compliant to the PCI Express Base Specification 2.1 with Endpoint and Root Port capability.
- Supports Gen1 (2.5 Gb/s) and Gen2 (5 Gb/s)

XADC Highlights:

XADC (Analog-to-Digital Converter)

On-chip temperature ($\pm 4^{\circ}\text{C}$ max error) and power supply ($\pm 1\%$ max error) sensors

- Continuous JTAG access to ADC measurements
- Internal access to all internal sensors of the Kintex™ 7

The board supports JTAG, master serial mode configuration from SPI-Flash or Slave Select MAP configuration for the User FPGA (Kintex™ 7) via the Board Configuration Controller (BCC).

The User FPGA is equipped with 6 I/O banks and 8 MGT (Gigabit Transceiver).

Bank	VCCO	VREF	Signals	Note
Bank 0	3.3V	none	SPI Configuration with or without Tandem configuration Slave Select Map Configuration GPIOs parallel DAC Interface	
Bank 14	3.3V	none		
Bank 15	3.3V	none		
Bank 12	2.5V	none	ser. ADC Interface Back I/O	
Bank 13	2.5V	none		
Bank 16	2.5V	none		
Bank 32	1.35V	0.625V	DDR3 Memory Interface 1GB diff. digital Front I/O (Ch. 2)	
Bank 33	1.35V	0.625V		
Bank 34	1.35V	0.625V		
Bank 115	Connected to FireFly Back I/O link			
Bank 116	PCIe X4 Interface to PCIe Switch Device			

Table 7-2 : FPGA Bank Usage

7.3 User FPGA Gigabit Transceiver (MGT)

The TPCE636 provides four MGT as Kintex™ 7 PCI Express Endpoint Block and four MGT for high speed FireFly Back I/O interface.

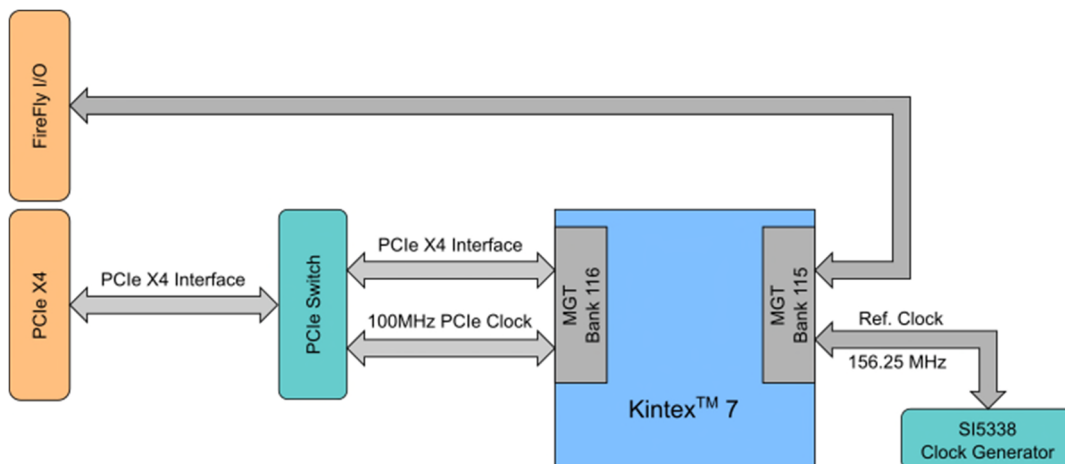


Figure 7-2 : MGT Block Diagram

MGT	TPCE636 Signal	FPGA Pins	Connected to
MGTXXTP0_115	MGTTX0	P2 / P1	connected to Back I/O FireFly Connector
	MGTRX0	R4 / R3	
MGTXXTP1_115	MGTTX1	M2 / M1	
	MGTRX1	N4 / N3	
MGTXXTP2_115	MGTTX2	K2 / K1	
	MGTRX2	L4 / L3	
MGTXXTP3_115	MGTTX3	H2 / H1	used for PCI Express Endpoint Block and connected to PCIe X4 Connector
	MGTRX3	J4 / J3	
MGTXXTP0_116	PET_03	G4 / G3	
	PER_03	F2 / F1	
MGTXXTP1_116	PET_02	E4 / E3	
	PER_02	D2 / D1	
MGTXXTP2_116	PET_01	C4 / C3	
	PER_01	B2 / B1	
MGTXXTP3_116	PET_00	B6 / B5	
	PER_00	A4 / A3	

Table 7-3 : MGT Connections

The MGT clock MGTREFCLK0_116 (PCI Express Endpoint Block clock reference) of 100 MHz is generated by the PI7C9X2G312GP PCIe Switch. The MGTREFCLK0_115 is connected to a 156.25 MHz clock output of the Si5338 low jitter clock generator. MGTREFCLK1_115 and MGTREFCLK1_116 are not used on the TPCE636.

MGT	TPCE636 Signal	FPGA Pins	Connected to
MGTREFCLK0_115	CLK_MGT	H6 / H5	156.25 MHz Si5338 Clock Generator
MGTREFCLK1_115	not used	K6 / K5	not connected
MGTREFCLK0_116	REFCLK_O2	D6 / D5	100 MHz PI7C9X2G312GP PCIe Switch
MGTREFCLK1_116	not used	F6 / F5	not connected

Table 7-4 : Multi Gigabit Transceiver Reference Clocks

7.4 User FPGA Configuration

The Kintex™ 7 can be configured by the following interfaces:

- Master Serial SPI Flash Configuration Interface
- JTAG Interface via FPGA JTAG Connector
- PCIe Interface via BCC FPGA Slave Select Map Interface Configuration

The change of the configuration mode is done with a configuration register of the BCC FPGA.

At Power-up, the TPCE636 User FPGA (Kintex™ 7) always configures via x4 SPI Interface by “Master Serial / SPI” mode.

At factory default the SPI Flash contains the TEWS example application for the TPCE636 User FPGA device.

7.4.1 Master Serial SPI Flash Configuration

It is important for User FPGA Configuration via SPI Master Mode that the ISP Mode Enable (ISP_EN) bit is clear to disable the ISP Mode. This is also the default value after Power Up.

See also Register Description of TPCE636 Configuration Device.

To comply with the PCI-Express specification it is necessary to perform the configuration as quickly as possible. The PCIe specification demands that a PCI device must be accessible after 100ms (120ms). To speed up the SPI Configuration the following points must be taken into account for SPI Bitstream generation.

- External Clock Master (53.2MHz) should be used.
- If external Clock Master is used, also the SPI Falling Edge Option must be used.
- SPI Configuration Bus Width should be set to X4.
- Xilinx Tandem Configuration Feature could be used for full PCI-Express specification compliance. Already during PCI-Express IP Core generation this configuration feature must be included. (For more information see: Xilinx XAPP1179).
- If the Tandem Configuration feature is used, the Persist Option is mandatory.
- For smaller FPGA content, it is sometimes also possible to comply with the PCI-Express specification, when only Bitstream Compression is used.

To avoid damage on the BCC or User FPGA (Kintex™ 7) if Tandem configuration or the Persist Option is used, the User FPGA must be set into reconfigure Mode by using the “FP_RE_CFG” Bit of the User FPGA Configuration Control/Status Register before Programming or Clearing the SPI Flash.

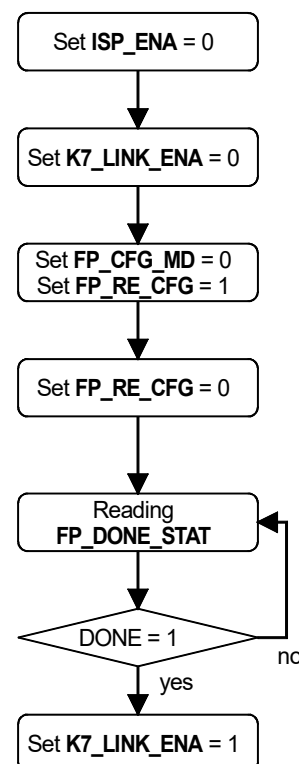
7.4.2 Manually User FPGA SPI Flash Reconfiguration

A manually User FPGA Reconfiguration can be performed with the User FPGA Reconfigure Command in the Global Configuration Register.

Set the User FPGA Reconfigure Command to set the User FPGA to configuration state with all FPGA I/O pins are High-Z.

Use the following procedure to perform a User FPGA SPI Reconfiguration

- Assure that ISP Mode Enable is disabled.
- By Reconfiguring the Kintex™ 7 the XILINX PCIe endpoint is reloaded and is temporarily not available on the PCI bus. To avoid error messages of the PCIe switch the link between the PCIe Switch and the Kintex™ 7 is disabled.
- Set the User FPGA Configuration Mode (FP_CFG_MD) to Master Serial / SPI and prepare the FPGA Reconfiguration.
- Start the FPGA Reconfiguration by setting the FP_RE_CFG bit of the User FPGA Configuration Control/Status Register to 0.
- Assure that the FPGA DONE Pin status shows a successful FPGA Configuration.
0: FPGA DONE Pin Level is Low (FPGA is not configured)
1: FPGA DONE Pin Level is High (FPGA is configured)
- The link between the PCIe Switch and the Kintex™ 7 must be enabled.



A successful User FPGA configuration is indicated with FPGA_DONE status in the Global Status Register and the on-board User FPGA Done LED.

It must be considered in any case, that the Reconfiguration of the User FPGA also reconfigures the PCIe Endpoint of the User FPGA. This leads to the consequence that the PCI Header of the User FPGA PCIe Endpoint no longer exists. For this purpose it is necessary to disable the link between the PCIe switch and the User FPGA PCIe Endpoint before preparing the FPGA Reconfiguration and to enable the link again after reconfiguration.

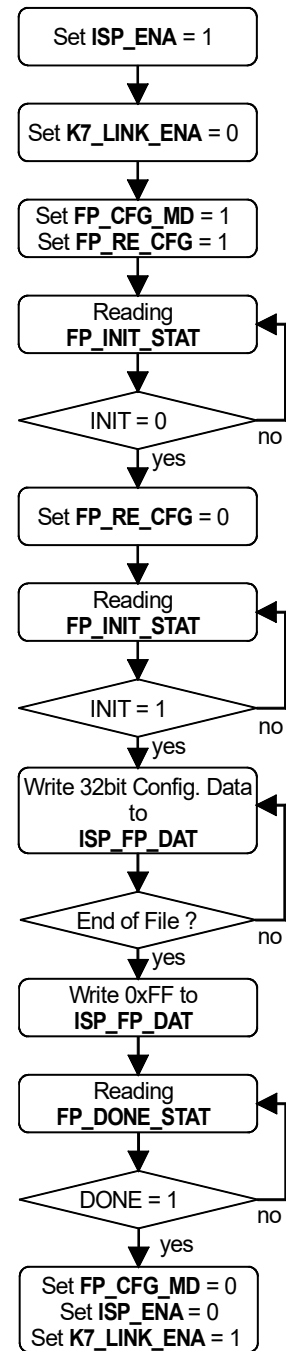
Additionally, after FPGA Reconfiguration the User FPGA PCIe Endpoint PCI Header must be configured again. If the PCIe interface of the User FPGA PCIe Endpoint does not change. Device ID, Vendor ID, Class Code and PCI Bars do not change, the PCI header could be saved before the FPGA Reconfiguration and written back to configuration space after the Reconfiguration.

7.4.3 Slave Select Map Configuration

For direct User FPGA configuration via PCIe Interface the **User FPGA Configuration Mode** must be set to **Slave SelectMap** Mode. The on-board logic sets the User FPGA in configuration state with all FPGA I/O pins switches to High-Z. User FPGA is now ready for new configuration data.

The following procedure is required for Select Map Mode User FPGA configuration / reconfiguration.

- First the In System Program (ISP) Mode must be enabled.
 - By reconfiguring the Kintex™ 7 the XILINX PCIe endpoint is reloaded and is temporarily not available on the PCI bus. To avoid error messages of the PCIe switch the link between the PCIe Switch and the Kintex™ 7 is disabled.
 - Check response of the Kintex™ 7 by reading the FPGA INIT_B pin value. If the Level is low the Kintex™ 7 FPGA is in Reset Mode, and then configuration process could be continued.
 - Start the FPGA Reconfiguration by setting the FP_RE_CFG bit of the User FPGA Configuration Control/Status Register to 0.
 - Check response of the Kintex™ 7 by reading the FPGA INIT_B pin value. While the FPGA INIT_B pin Level is low the Kintex™ 7 isn't ready for configuration.
 - If FPGA INIT_B pin high then the configuration data must be continually written to the ISP SelectMap Data Register. Typically 2860903 PCI write accesses are required to configure a Kintex™ 7 325T.
 - Dummy Write accesses to create configuration clock cycles while FP_DONE_STAT is low.
 - A successful configuration of the User FPGA is indicated with FP_DONE_STAT in the User FPGA Configuration Control/Status Register and the on-board User FPGA Done LED.
- 0: FPGA DONE Pin Level is Low (FPGA is not configured)
1: FPGA DONE Pin Level is High (FPGA is configured)
- After reconfiguration was successful the User FPGA Configuration Mode and the ISP Mode could be disabled. Also the link between the PCIe Switch and the Kintex™ 7 must be enabled.



If not all configuration data bytes are written the User FPGA is not configured correctly.

The number of bytes that must be written corresponds to the size of the XILINX configuration files. Typically the .bin or the .bit file could be used as data source.

The .bit file is the standard generated programming file. This is a binary configuration data file which contains header information that does not need to be downloaded to the FPGA. For generating the .bin file the BitGen option must be used. This is also a binary configuration data file but without header information. For configuration of the Kintex™ 7 FPGA on the TPCE636 both files could be used. Both binary configuration data files have additional data to the actual configuration data.

See also the XILINX User Guide (ug470) “7 Series FPGAs Configuration” for more information about Configuration Details and Configuration Data File Formats.

The following BitGen options are mandatory for the Slave Select Map Configuration via BBC.

- External Clock Master (53.2MHz) must be used.
- In contrast to SPI Configuration Mode, the Falling Edge Option must be switched off.

Additional important BitGen Options:

- For a faster configuration the Bitstream Compression could be used.
- The Persist Option is not needed. But if this option is used, the User FPGA must be set into reconfigure Mode by using the “FP_RE_CFG” Bit of the User FPGA Configuration Control/Status Register before Programming or Clearing the SPI Flash.

Xilinx Tandem Configuration Feature cannot be used for Slave Select Map Configuration. It is therefore necessary to remove the Tandem Configuration Feature from the PCIe IP Core.

A design that is intended for the SPI configuration cannot be used by Slave Select Map configuration and vice versa.

7.4.4 Configuration via JTAG

The TPCE636 provides two JTAG chains which are accessible by the following connector options:

7.4.4.1 User JTAG Chain

For direct FPGA configuration, FPGA read back or in-system diagnostics with Vivado Logic Analyzer, the Molex Debug Connector can be used to access the JTAG-chain. Also an indirect SPI – PROM programming is possible via the User JTAG Chain. This Connector provides a direct connection to a Xilinx USB Programmer II compatible 2 mm shrouded header.

The second possible source is the BCC internal JTAG TAB controller. See chapter 7.12 JTAG Controller to K7 JTAG Interface.

The third option is the USB to JTAG capable FTDI controller, which can be accessed via a USB C interface (X2).

The Molex JTAG connector X5 has default priority. If a cable is connected here, the BCC switches this interface active. However, each of the three sources can be selected via the User FPGA JTAG Control and Status Register - 0x80 in BCC.

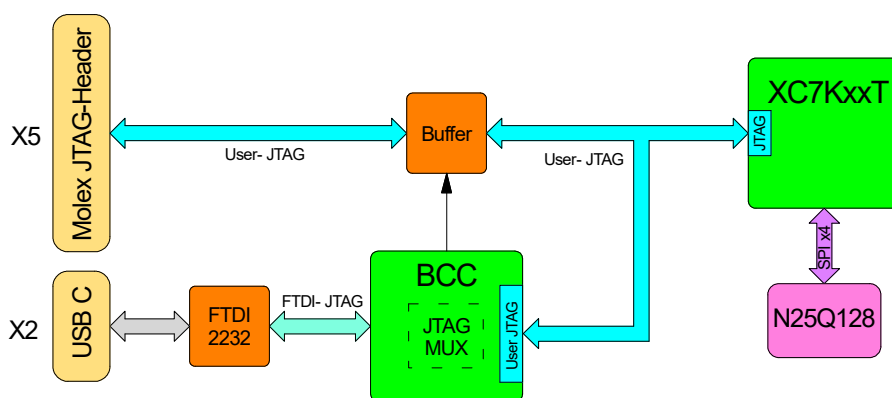


Figure 7-3 : User JTAG-Chain

7.4.4.2 TEWS Factory JTAG Chain

The TEWS Factory JTAG Chain is accessible from the PCIe Edge Connector.

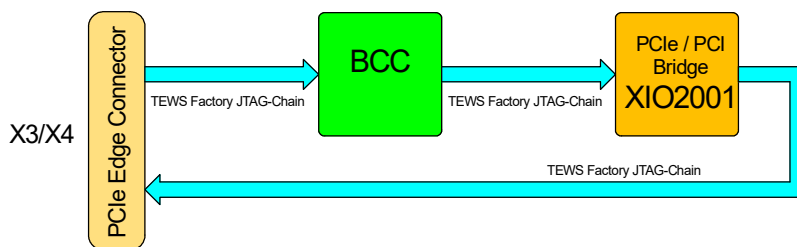


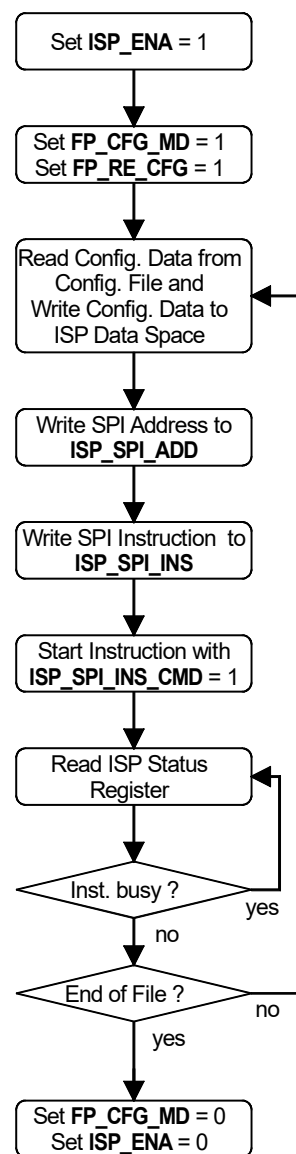
Figure 7-4 : TEWS Factory JTAG-Chain

7.4.5 Programming User FPGA SPI Configuration Flash

To program the User FPGA SPI Configuration Flash the **User FPGA Configuration Mode** must be set to **Master Serial / SPI** and the ISP Mode must be enabled.

The following procedure is required for User FPGA SPI Configuration Flash programming and subsequent reconfiguration of the User FPGA.

- Enable the ISP Mode in the ISP Mode Enable Register.
- Assure that User FPGA Configuration Mode is set to SPI Flash. If the FPGA is not configured or if it is possible that the FPGA accesses the SPI flash during BCC access set FP_RE_CFG = 0b1. Link must be set to disable previously!
- Read Configuration data from Configuration File and write Data to the In Circuit Programming Data Space. 256Byte (1 SPI Flash page) each time can be programmed maximally.
- Set the programming start address and write instruction in the ISP Configuration Register.
- Start the Instruction with ISP Command Register
- Wait on ISP SPI Instruction Done or ISP SPI Page Data Done for next write instruction.
- Process should be repeated until all configuration data is written to the SPI Flash
- After completion of the data programming, the ISP Mode bit must be cleared to set configuration path to User FPGA and a Reconfiguration can be performed.



A successful configuration of the User FPGA is indicated with FP_DONE_STAT in the User FPGA Configuration Control/Status Register and the on-board User FPGA Done LED.

The Programming Instruction always starts at address 0x00 to write data from the ISP Programming Data Space to the SPI flash.

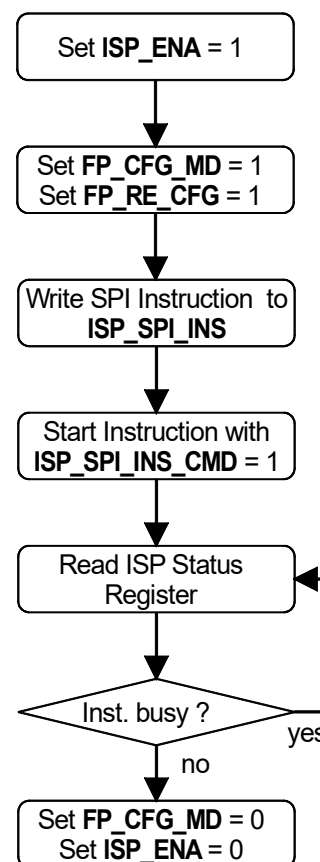
If not all configuration data bytes are written, the User FPGA is not configured correctly.

The source for the User FPGA SPI Configuration Flash data could be the .bin file. This file format can be created from the .bit file by using the XILINX Vivado software.

7.4.6 Erasing User FPGA SPI Configuration Flash

For Chip Erase of the User FPGA SPI Configuration Flash the **User FPGA Configuration Mode** must be set to **Master Serial / SPI** and the ISP Mode must be enabled.

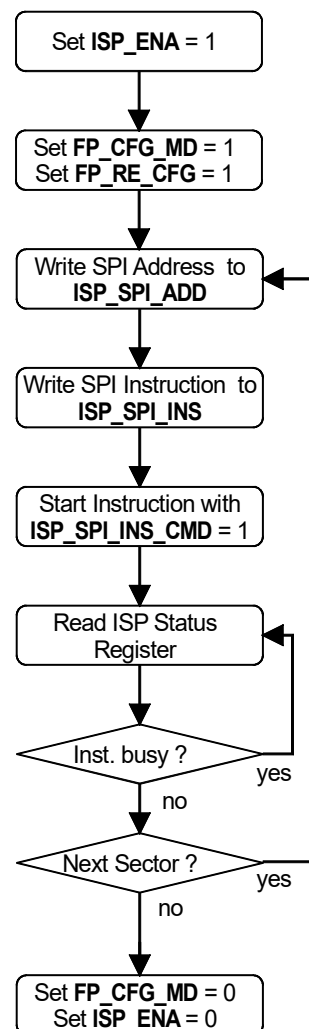
- Enable the ISP Mode in the ISP Mode Enable Register.
- Assure that User FPGA Configuration Mode is set to SPI Flash. If the FPGA is not configured or if it is possible that the FPGA accesses the SPI flash during BCC access set `FP_RE_CFG = 0b1`. Link must be set to disable previously!
- Set the Chip Erase instruction in the ISP Configuration Register.
- Start the Instruction with ISP Command Register
- Wait on ISP SPI Instruction Done or ISP SPI Page Data Done for erasing process end.
- After completion of the erasing process, the ISP Mode bit should be cleared to set configuration path to User FPGA or a User FPGA SPI Configuration Flash programming process could be done.



7.4.7 Sector Erasing User FPGA SPI Configuration Flash

For Sector Erase of the User FPGA SPI Configuration Flash the **User FPGA Configuration Mode** must be set to **Master Serial / SPI** and the ISP Mode must be enabled.

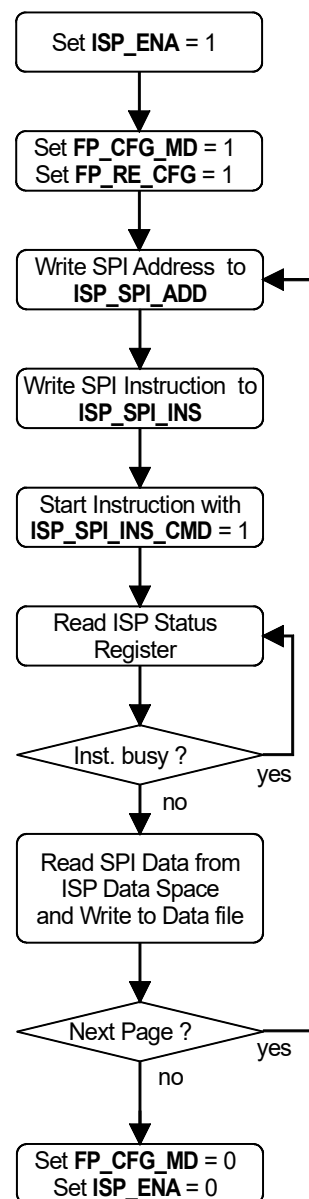
- Enable the ISP Mode in the ISP Mode Enable Register.
- Assure that User FPGA Configuration Mode is set to SPI Flash. If the FPGA is not configured or if it is possible that the FPGA accesses the SPI flash during BCC access set FP_RE_CFG = 0b1. Link must be set to disable previously!
- Write the Sector Address to the ISP Configuration Register
- Set the Chip Erase instruction in the ISP Configuration Register.
- Start the Instruction with ISP Command Register
- Wait on ISP SPI Instruction Done or ISP SPI Page Data Done for erasing process end.
- Process could be repeated for other sectors.
- After completion of the erasing process, the ISP Mode bit should be cleared to set configuration path to User FPGA or a User FPGA SPI Configuration Flash programming process could be done.



7.4.8 Reading User FPGA SPI Configuration Flash

To read the User FPGA SPI Configuration Flash the **User FPGA Configuration Mode** must be set to **Master Serial / SPI** and the ISP Mode must be enabled.

- Enable the ISP Mode in the ISP Mode Enable Register.
- Assure that User FPGA Configuration Mode is set to SPI Flash. If the FPGA is not configured or if it is possible that the FPGA accesses the SPI flash during BCC access set FP_RE_CFG = 0b1. Link must be set to disable previously!
- Set the reading start address and write instruction in the ISP Configuration Register.
- Start the Instruction with ISP Command Register
- Wait on ISP SPI Instruction Done or ISP SPI Page Data Done for next write instruction.
- Read one page of SPI Data from In Circuit Programming Data Space and write to Data file
- Process could be repeated until all needed data is written to the Data file.
- After completion of the reading process, the ISP Mode bit must be cleared to set configuration path back to User FPGA.



7.5 BCC (Board Configuration Controller) FPGA

The Board Configuration FPGA is factory configured, and handles the basic board setup and User FPGA (Kintex™ 7) Configuration.

Changing or erasing the BCF (Board Configuration Firmware) content leads to an inoperable TPCE636 FPGA configuration.

7.5.1 I2C Interface to BCC Register

The TPCE636 BCC provides an I2C Interface to the User FPGA (Kintex™ 7). Via this I2C Interface the TPCE636 Serial Number Register from the BCC Local Configuration Register Space could be read.

For the User FPGA (Kintex™ 7) an I2C Master interface is required to use this interface. For this purpose the BCC provides an I2C slave interface.

Signal	Bank	V _{CCO}	Pin	Description
FPGA_SCL	15	3.3V	L18	Serial Clock Output A negative edge clock data out.
FPGA_SDA	15	3.3V	M17	Bidirectional Serial Data

Table 7-5: User FPGA I2C Interface to BCC

The following table lists the available clock sources on the TPCE636:

FPGA Clock Signal Name	FPGA Pin Number	Source	Description
CLK_MGT±	H6 / H5	SI5338 low-jitter clock generator	156.25 MHz differential MGT Reference clock
REFCLKO2±	D6 / D5	PCIe Switch PI7C9X2G312GP	100 MHz differential PCIe Reference clock input
MCB_CLK±	AB11 / AC11	SI5338 low-jitter clock generator	88.889 MHz differential MCB CLK
REF_CLK±	AA10 / AB10	SI5338 low-jitter clock generator	200 MHz differential Reference clock
USER_CLKA	F22	SI5338 low-jitter clock generator	105 MHz Clock Input This clock is designated for ADC interface clock source.
Si514_CLK±	G22 / F23	Si514 prog. Oscillator	Differential free I2C prog. XO 100kHz up to 250MHz Default = 156 MHz
SCKOUT_00±	Y22 / AA22	LTC2323	Diff. Clock ADC Ch. 0 and 1
SCKOUT_01±	AC23 / AC24	LTC2323	Diff. Clock ADC Ch. 2 and 3
SCKOUT_02±	Y23 / AA24	LTC2323	Diff. Clock ADC Ch. 4 and 5
SCKOUT_03±	AA23 / AB24	LTC2323	Diff. Clock ADC Ch. 6 and 7
SCKOUT_04±	R22 / R23	LTC2323	Diff. Clock ADC Ch. 8 and 9
SCKOUT_05±	R21 / P21	LTC2323	Diff. Clock ADC Ch. 10 and 11
SCKOUT_06±	N21 / N22	LTC2323	Diff. Clock ADC Ch. 12 and 13
SCKOUT_07±	P23 / N23	LTC2323	Diff. Clock ADC Ch. 14 and 15
BACK_IO31±	C12 / C11	Back I/O Connector	Diff. Back I/O Clock Input (IO31)
BACK_IO30±	E11 / D11	Back I/O Connector	Diff. Back I/O Clock Input (IO30)
BACK_IO28±	G11 / F10	Back I/O Connector	Diff. Back I/O Clock Input (IO28)
DIG_IO_01±	E10 / D10	FireFly I/O Connector	Diff. I/O Clock Input (DIG_IO_01)
K7_EMCCLK	B26	BCC	53.2 MHz used for external configuration clock (CCLK)

Table 7-6 : Available FPGA clocks

7.6.2 Si514 Free Programming Clock source

A second clock source is the free programmable Si514 XO from Silicon Laboratories.

The Si514 is programmable via I2C interface. An internal PLL allows output frequency of 100 kHz up to 250 MHz with programming resolution of 0.026 parts per billion.

The Si514 on TPCE636 is factory configured to 156 MHz default frequency.

The Si514 is connected via I2C interface to User FPGA (Kintex™ 7). As usual for the I2C interface, the two pins must be realized as open drain buffer. The same I2C interface is used for the calibration data prom.

Si514 I2C Bus Signal	Bank	V _{CCO}	Pin	Description / Kintex™ 7
USER_SCL	14	3.3V	F25	Serial clock
USER_SDA	14	3.3V	G26	Serial data

Table 7-7 : FPGA I2C SI514 Connections

Configuration and operation of the Si514 is controlled by reading and writing to the Si514 internal RAM space using the I2C interface. The device operates in slave mode with 7-bit addressing and can operate in Standard-Mode (100kbps) or Fast-Mode (400kbps).

The I2C 7-bit slave address of the Si514 is 0x55h.

To use the serial I2C interface between the USER FPGA (Kintex™ 7) and the Si514 please also see the Silicon Labs Si514 data sheet which describes the register map and serial communication process.

7.7 Back I/O Interface

The Back I/O Pins of the TPCE636 are directly routed to the User FPGA (Kintex™ 7). The I/O functions of these FPGA pins are directly dependent on the configuration of the FPGA.

The Kintex™ 7 VCCO voltage is set to 2.5V, so only the 2.5V I/O standards LVCMOS25, LVTTTL25 and LVDS_25 are possible when using the TPCE636 back I/O interface.

Signal Name	Pin Number	Direction	IO Standard for example
BACK_IO0+	AE23	IN/OUT	LVDS_25
BACK_IO0-	AF23	IN/OUT	LVDS_25
BACK_IO1+	AE22	IN/OUT	LVDS_25
BACK_IO1-	AF22	IN/OUT	LVDS_25
BACK_IO2+	AD21	IN/OUT	LVDS_25
BACK_IO2-	AE21	IN/OUT	LVDS_25
BACK_IO3+	W20	IN/OUT	LVDS_25
BACK_IO3-	Y21	IN/OUT	LVDS_25
BACK_IO4+	T18	IN/OUT	LVDS_25
BACK_IO4-	T19	IN/OUT	LVDS_25
BACK_IO5+	R16	IN/OUT	LVDS_25
BACK_IO5-	R17	IN/OUT	LVDS_25
BACK_IO6+	N18	IN/OUT	LVDS_25
BACK_IO6-	M19	IN/OUT	LVDS_25
BACK_IO7+	P16	IN/OUT	LVDS_25
BACK_IO7-	N17	IN/OUT	LVDS_25
BACK_IO8+	J13	IN/OUT	LVDS_25
BACK_IO8-	H13	IN/OUT	LVDS_25
BACK_IO9+	H14	IN/OUT	LVDS_25
BACK_IO9-	G14	IN/OUT	LVDS_25
BACK_IO10+	J11	IN/OUT	LVDS_25
BACK_IO10-	J10	IN/OUT	LVDS_25
BACK_IO11+	H12	IN/OUT	LVDS_25
BACK_IO11-	H11	IN/OUT	LVDS_25
BACK_IO12+	G12	IN/OUT	LVDS_25
BACK_IO12-	F12	IN/OUT	LVDS_25
BACK_IO13+	H9	IN/OUT	LVDS_25
BACK_IO13-	H8	IN/OUT	LVDS_25
BACK_IO14+	F14	IN/OUT	LVDS_25
BACK_IO14-	F13	IN/OUT	LVDS_25
BACK_IO15+	G10	IN/OUT	LVDS_25
BACK_IO15-	G9	IN/OUT	LVDS_25

BACK_IO16+	E13	IN/OUT	LVDS_25
BACK_IO16-	E12	IN/OUT	LVDS_25
BACK_IO17+	F9	IN/OUT	LVDS_25
BACK_IO17-	F8	IN/OUT	LVDS_25
BACK_IO18+	P19	IN/OUT	LVDS_25
BACK_IO18-	P20	IN/OUT	LVDS_25
BACK_IO19+	C14	IN/OUT	LVDS_25
BACK_IO19-	C13	IN/OUT	LVDS_25
BACK_IO20+	D9	IN/OUT	LVDS_25
BACK_IO20-	D8	IN/OUT	LVDS_25
BACK_IO21+	B15	IN/OUT	LVDS_25
BACK_IO21-	A15	IN/OUT	LVDS_25
BACK_IO22+	B12	IN/OUT	LVDS_25
BACK_IO22-	B11	IN/OUT	LVDS_25
BACK_IO23+	B14	IN/OUT	LVDS_25
BACK_IO23-	A14	IN/OUT	LVDS_25
BACK_IO24+	C9	IN/OUT	LVDS_25
BACK_IO24-	B9	IN/OUT	LVDS_25
BACK_IO25+	A13	IN/OUT	LVDS_25
BACK_IO25-	A12	IN/OUT	LVDS_25
BACK_IO26+	B10	IN/OUT	LVDS_25
BACK_IO26-	A10	IN/OUT	LVDS_25
BACK_IO27+	A9	IN/OUT	LVDS_25
BACK_IO27-	A8	IN/OUT	LVDS_25
BACK_IO28+	G11	IN/OUT	LVDS_25
BACK_IO28-	F10	IN/OUT	LVDS_25
BACK_IO29+	D14	IN/OUT	LVDS_25
BACK_IO29-	D13	IN/OUT	LVDS_25
BACK_IO30+	E11	IN/OUT	LVDS_25
BACK_IO30-	D11	IN/OUT	LVDS_25
BACK_IO31+	C12	IN/OUT	LVDS_25
BACK_IO31-	C11	IN/OUT	LVDS_25

Table 7-8 : Digital Back I/O Interface

7.8 Memory

The TPCE636 is equipped with a 1GB, 32bit wide DDR3 SDRAM and a 128Mbit non-volatile SPI-Flash. The SPI-Flash can also be used as the User FPGA configuration memory.

7.8.1 DDR3 SDRAM

The TPCE636 provides two MT41... (96-ball) DDR3 memory devices. The memory is accessible through a Memory Interface Controller Block IP in bank 32, 33 and 34 of the User FPGA.

Signal	FPGA Pin Number	Termination	Memory Devices	
			Pin	Name
A0	AC8	49.9Ω VTT	N3	A0
A1	AA7	49.9Ω VTT	P7	A1
A2	AA8	49.9Ω VTT	P3	A2
A3	AF7	49.9Ω VTT	N2	A3
A4	AE7	49.9Ω VTT	P8	A4
A5	W8	49.9Ω VTT	P2	A5
A6	V9	49.9Ω VTT	R8	A6
A7	Y10	49.9Ω VTT	R2	A7
A8	Y11	49.9Ω VTT	T8	A8
A9	Y7	49.9Ω VTT	R3	A9
A10	Y8	49.9Ω VTT	L7	A10
A11	V7	49.9Ω VTT	R7	A11
A12	V8	49.9Ω VTT	N7	A12
A13	W11	49.9Ω VTT	T3	NC/A13
A14	V11	49.9Ω VTT	T7	NC/A14
BA0	AC7	49.9Ω VTT	M2	BA0
BA1	AB7	49.9Ω VTT	N8	BA1
BA2	AD8	49.9Ω VTT	M3	BA2
RAS#	AA9	49.9Ω VTT	J3	RAS#
CAS#	AB9	49.9Ω VTT	K3	CAS#
WE#	AC9	49.9Ω VTT	L3	WE#
RESET#	W10	4.7kΩ GND	T2	RESET#
CKE[0]	AB12	4.7kΩ GND	K9	CKE
ODT[0]	AC12	4.7kΩ GND	K1	ODT
DM_0	AE17	ODT	E7	LDM
DM_1	AA14	ODT	D3	UDM
DM_2	U6	ODT	E7	LDM
DM_3	Y3	ODT	D3	UDM
DQ0	AF17	ODT	E3	DQ0
DQ1	AF14	ODT	F7	DQ1
DQ2	AF15	ODT	F2	DQ2
DQ3	AD15	ODT	F8	DQ3

Signal	FPGA Pin Number	Termination	Memory Devices	
			Pin	Name
DQ4	AE15	ODT	H3	DQ4
DQ5	AF19	ODT	H8	DQ5
DQ6	AF20	ODT	G2	DQ6
DQ7	AD16	ODT	H7	DQ7
DQ8	AA15	ODT	D7	DQ8
DQ9	AC14	ODT	C3	DQ9
DQ10	AD14	ODT	C8	DQ10
DQ11	AB14	ODT	C2	DQ11
DQ12	AB15	ODT	A7	DQ12
DQ13	AA17	ODT	A2	DQ13
DQ14	AA18	ODT	B8	DQ14
DQ15	AB16	ODT	A3	DQ15
DQ16	U5	ODT	E3	DQ0
DQ17	U2	ODT	F7	DQ1
DQ18	U1	ODT	F2	DQ2
DQ19	V3	ODT	F8	DQ3
DQ20	W3	ODT	H3	DQ4
DQ21	U7	ODT	H8	DQ5
DQ22	V6	ODT	G2	DQ6
DQ23	V4	ODT	H7	DQ7
DQ24	Y2	ODT	D7	DQ8
DQ25	V2	ODT	C3	DQ9
DQ26	V1	ODT	C8	DQ10
DQ27	W1	ODT	C2	DQ11
DQ28	Y1	ODT	A7	DQ12
DQ29	AB2	ODT	A2	DQ13
DQ30	AC2	ODT	B8	DQ14
DQ31	AA3	ODT	A3	DQ15
CK_p	AC13	100Ω	J7	CK
CK_n	AD13		K7	CK#
DQS_0_p	AE18	ODT	F3	LDQS
DQS_0_n	AF18	ODT	G3	LDQS#
DQS_1_p	Y15	ODT	C7	UDQS
DQS_1_n	Y16	ODT	B7	UDQS#
DQS_2_p	W6	ODT	F3	LDQS
DQS_2_n	W5	ODT	G3	LDQS#
DQS_3_p	AB1	ODT	C7	UDQS
DQS_3_n0	AC1	ODT	B7	UDQS#

Table 7-9 : DDR3 SDRAM Interface

DDR3 Memory Device 01
DDR3 Memory Device 02
Both DDR3 Memory Devices 01 & 02

For details regarding the DDR3 SDRAM interface, please refer to XILINX Memory Interface Generator Documentation. Xilinx UG586: *Zynq-7000 AP SoC and 7 Series Devices Memory Interface Solutions v4.0*.

7.8.2 SPI-Flash

The TPCE636 provides a Cypress S25FL127S 128-Mbit serial Flash memory. This Flash is used as FPGA configuration source (default configuration source).

After configuration, it is always accessible from the FPGA, so it also can be used for code or user data storage.

The SPI-EEPROM is connected via Quad (x4) SPI interface to the User FPGA (Kintex™ 7) configuration interface.

SPI-PROM Signal	Bank	V _{CCO}	Pin	Description / Kintex™ 7
CLK	0	3.3V	C8	Serial Clock (CCLK_B)
CS#	14	3.3V	C23	Chip Select (FCS_B)
DI (bit0)	14	3.3V	B24	Serial Data input (MOSI) / MISO[0]
DO (bit1)	14	3.3V	A25	Serial Data output (DIN) / MISO[1]
WP# (bit2)	14	3.3V	B22	MISO[2] – D02
HOLD# (bit3)	14	3.3V	A22	MISO[3] – D03

Table 7-10 : FPGA SPI-Flash Connections

7.8.3 I2C - EEPROM

The TPCE636 provides an Atmel AT24C04D (512x8) I2C-Compatible (2-wire) Serial EEPROM.

This EEPROM is used as ADC and DAC calibration data memory. In addition, for each correction data set ADC Input, DAC Output $\pm 10V$, DAC Output $\pm 5V$ and DAC Output $\pm 2.5V$ the corresponding voltage range of ADC inputs and DAC outputs are also stored.

In the first versions of TPCE636 only the calibration data were stored. A version register is used to identify whether the voltage ranges are also present. In version 0x00, only the calibration data is available. The version 0x01 additionally contains the voltage ranges.

During factory test the analog input and output channel gain error and offset error are determined. For each device, 16bit correction values are stored to the I2C EEPROM. These calibration data values have been determined with TEWS test environment. If system specific calibration data is needed, the calibration of the entire system can be done by the user and the I2C EEPROM could be used as a possible memory to store the custom calibration data.

The I2C EEPROM is connected via 2-wire interface to User FPGA (Kintex™ 7). As usual for the I2C interface the two pins must be realized as open drain buffers.

SPI-PROM Signal	Bank	V _{CCO}	Pin	Description / Kintex™ 7
USER_SCL	14	3.3V	F25	Serial clock
USER_SDA	14	3.3V	G26	Serial data

Table 7-11: FPGA I2C EEPROM Connections

For using the serial I2C interface between the USER FPGA (Kintex™ 7) and the I2C EEPROM please see the Atmel AT24C04D data sheet which describes the serial communication process.

7.8.3.1 I2C Calibration Data

There are two errors affecting the accuracy of the ADC and DAC that can be corrected using the factory calibrated calibration data. The correction values are obtained during factory calibration and are stored in an on-board I2C EEPROM as 2-complement 16bit values in the range from -32768 to +32767. To achieve a higher accuracy, they are scaled to $\frac{1}{4}$ LSB.

DAC Offset Error:

For the DAC, this is the data value that is required to produce a zero voltage output signal. This error is corrected by subtracting the offset from the DAC data value.

ADC Offset Error:

The offset error is the data value when converting with the input connected to its own ground in single-ended mode, or with shorted inputs in differential mode. This error is corrected by subtracting the known error from the reading.

DAC Gain Error:

The gain error is the difference between the ideal gain and the actual gain of the DAC. It is corrected by multiplying the DAC data value with a correction factor.

ADC Gain Error:

The gain error is the difference between the ideal gain and the actual gain of the programmable gain amplifier and the ADC. This error is corrected by multiplying the reading with a correction factor.

7.8.3.2 ADC and DAC Calibration Data Values

The 16 ADC channels are realized with eight dual LTC2323-16 ADC devices.

For each ADC channel a 16bit Offset correction and a 16bit gain correction value is stored.

I2C EEPROM Address	Description	Size (Bit)
0x000	ADC Channel 1 Offset _{corr} High Byte	8
0x001	ADC Channel 1 Offset _{corr} Low Byte	8
0x002	ADC Channel 1 Gain _{corr} High Byte	8
0x003	ADC Channel 1 Gain _{corr} Low Byte	8
...		
0x03C	ADC Channel 16 Offset _{corr} High Byte	8
0x03D	ADC Channel 16 Offset _{corr} Low Byte	8
0x03E	ADC Channel 16 Gain _{corr} High Byte	8
0x03F	ADC Channel 16 Gain _{corr} Low Byte	8

Table 7-12: ADC Calibration Data Values

7.8.3.3 DAC Calibration Data Values

The 16 DAC channels are realized with eight dual AD5547 DAC devices.

For each DAC channel three calibration sets are stored. One set for each DAC Output Voltage Range Selection ($\pm 10V$ range (default value), $\pm 5V$ range and $\pm 2.5V$ range).

Each calibration set consists of a 16bit value for Offset correction and Gain correction for each channel.

I2C EEPROM Address	Description	Size (Bit)
0x040	DAC Range ± 10 Channel 1 Offset _{corr} High Byte	8
0x041	DAC Range ± 10 Channel 1 Offset _{corr} Low Byte	8
0x042	DAC Range ± 10 Channel 1 Gain _{corr} High Byte	8
0x043	DAC Range ± 10 Channel 1 Gain _{corr} Low Byte	8
...		
0x07C	DAC Range ± 10 Channel 16 Offset _{corr} High Byte	8
0x07D	DAC Range ± 10 Channel 16 Offset _{corr} Low Byte	8
0x07E	DAC Range ± 10 Channel 16 Gain _{corr} High Byte	8
0x07F	DAC Range ± 10 Channel 16 Gain _{corr} Low Byte	8
0x080	DAC Range ± 5 Channel 1 Offset _{corr} High Byte	8
0x081	DAC Range ± 5 Channel 1 Offset _{corr} Low Byte	8
0x082	DAC Range ± 5 Channel 1 Gain _{corr} High Byte	8
0x083	DAC Range ± 5 Channel 1 Gain _{corr} Low Byte	8
...		
0x0BC	DAC Range ± 5 Channel 16 Offset _{corr} High Byte	8
0x0BD	DAC Range ± 5 Channel 16 Offset _{corr} Low Byte	8
0x0BE	DAC Range ± 5 Channel 16 Gain _{corr} High Byte	8
0x0BF	DAC Range ± 5 Channel 16 Gain _{corr} Low Byte	8
0x0C0	DAC Range ± 2.5 Channel 1 Offset _{corr} High Byte	8
0x0C1	DAC Range ± 2.5 Channel 1 Offset _{corr} Low Byte	8
0x0C2	DAC Range ± 2.5 Channel 1 Gain _{corr} High Byte	8
0x0C3	DAC Range ± 2.5 Channel 1 Gain _{corr} Low Byte	8
...		
0x0FC	DAC Range ± 2.5 Channel 16 Offset _{corr} High Byte	8
0x0FD	DAC Range ± 2.5 Channel 16 Offset _{corr} Low Byte	8
0x0FE	DAC Range ± 2.5 Channel 16 Gain _{corr} High Byte	8
0x0FF	DAC Range ± 2.5 Channel 16 Gain _{corr} Low Byte	8

Table 7-13: DAC Calibration Data Values

7.8.3.4 ADC Data Correction Formula

Please use the total 16 bit data register value for the ADC correction formula.

The basic formula for correcting any ADC reading for the TPCE636 (bipolar input voltage range) is:

$$Value = Reading \cdot \left(1 - \frac{Gain_{corr}}{131072} \right) - \frac{Offset_{corr}}{4}$$

Value is the corrected result.

Reading is the data read from the ADC Data Register.

Gain_{corr} and *Offset_{corr}* are the ADC correction factors from the Calibration Data ROM stored for each programmable gain factor.

The correction values are stored as two's complement 16 bit values in the range -32768 to 32767. For higher accuracy they are scaled to ¼ LSB.

Floating point arithmetic or scaled integer arithmetic is necessary to avoid rounding error while computing above formula.

7.8.3.5 DAC Data Correction Formula

The basic formula for correcting any DAC value is:

$$Data = Value \cdot \left(1 - \frac{Gain_{corr}}{131072} \right) - \frac{Offset_{corr}}{4}$$

Value is the desired DAC value.

Data is the corrected DAC value that must be sending to the DAC.

Gain_{corr} and *Offset_{corr}* are the DAC correction values from the Calibration Data ROM. They are stored separately for each of the 8 DAC channels.

The correction values are stored as two's complement byte wide values in the range from -32768 to +32767. For higher accuracy they are scaled to ¼ LSB.

Floating point arithmetic or scaled integer arithmetic must be used to avoid rounding errors in computing above formula.

7.8.3.6 ADC and DAC voltage ranges

The voltage range for each ADC input and each DAC output is composed of two bytes, one high byte and one low byte. Together they provide the 16bit voltage range in mV.

For Example:

value high byte = 0x27 and value low byte = 0x10 => 0x2710 => ± 10000 mV

value high byte = 0x28 and value low byte = 0x63 => 0x2863 => ± 10339 mV

value high byte = 0x09 and value low byte = 0xC4 => 0x09C4 => ± 2500 mV

I2C EEPROM Address	Description	Size (Bit)
0x100	ACD Input Voltage Range Channel 1 High Byte	8
0x101	ACD Input Voltage Range Channel 1 Low Byte	8
0x102	ACD Input Voltage Range Channel 2 High Byte	8
0x103	ACD Input Voltage Range Channel 2 Low Byte	8
...		
0x11E	ACD Input Voltage Range Channel 16 High Byte	8
0x11F	ACD Input Voltage Range Channel 16 Low Byte	8
0x120	DAC Output Range ± 10 Channel 1 High Byte	8
0x121	DAC Output Range ± 10 Channel 1 Low Byte	8
0x122	DAC Output Range ± 10 Channel 2 High Byte	8
0x123	DAC Output Range ± 10 Channel 2 Low Byte	8
...		
0x13E	DAC Output Range ± 10 Channel 16 High Byte	8
0x13F	DAC Output Range ± 10 Channel 16 Low Byte	8
0x140	DAC Output Range ± 5 Channel 1 High Byte	8
0x141	DAC Output Range ± 5 Channel 1 Low Byte	8
0x142	DAC Output Range ± 5 Channel 2 High Byte	8
0x143	DAC Output Range ± 5 Channel 2 Low Byte	8
...		
0x15E	DAC Output Range ± 5 Channel 16 High Byte	8
0x15F	DAC Output Range ± 5 Channel 16 Low Byte	8
0x160	DAC Output Range ± 2.5 Channel 1 High Byte	8
0x161	DAC Output Range ± 2.5 Channel 1 Low Byte	8
0x162	DAC Output Range ± 2.5 Channel 2 High Byte	8
0x163	DAC Output Range ± 2.5 Channel 2 Low Byte	8
...		
0x17E	DAC Output Range ± 2.5 Channel 16 High Byte	8
0x17F	DAC Output Range ± 2.5 Channel 16 Low Byte	8

Table 7-14: ADC and DAC voltage ranges

7.8.3.7 Version of EEPROM data structure

In the first versions of TPCE636 only the calibration data were stored. A version register is used to identify whether the voltage ranges are also present.

I2C EEPROM Address	Description	Size (Bit)
0x180	not used	
...		
0x1FE		
0x1FF	EEPROM data structure Version 0x00: Only the ADC and DAC calibration data are available. 0x01: ADC and DAC calibration data and voltage range data are available.	8

Table 7-15: Version of EEPROM data structure

7.9 Serial ADC Interface

7.9.1 Overview

The 16 analog inputs of the TPCE636 are realized with 8 LTC2323-16 ADC devices. Each of these SAR-ADCs has two ADC channels. Thus, a total of 16 ADC channels are available on the TPCE636.

A coarse overview of the analog input section of the TPCE636 is shown by the following figure:

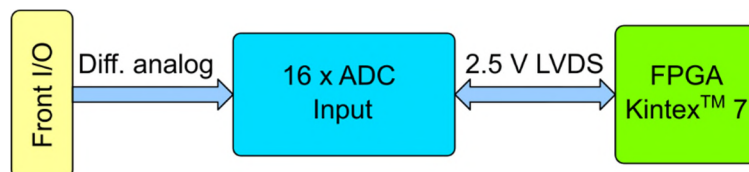


Figure 7-6 : Analog Input Section

The key-features of the LTC2323-16 are:

- Dual Channel - SAR-ADC
- 16bit resolution
- 5Msps Throughput Rate for each Channel
- LVDS SPI-Compatible Serial Interface to User FPGA (Kintex™ 7)

In order to adapt the LTC2323-16 to a $\pm 10V$ input voltage on each input-pin ($\pm 20V$ differential voltage range) two stage input operational amplifiers for input impedance conversion and gain adaption are used in addition to the ADC.

The following figure shows the structure and principle of two ADC inputs. Both are connected via impedance converter and level adjustment to the LTC2323-16 dual ADC.

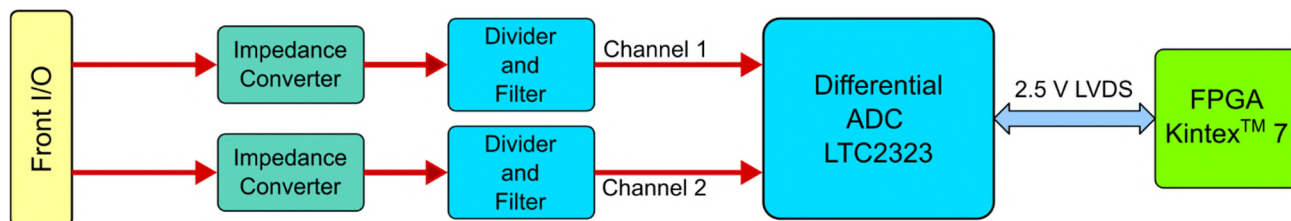


Figure 7-7 : Analog Input Block Diagram

To minimize power requirement, unnecessary temperature rise and signal interference, unused ADC inputs should be set to 0V differential or GND.

7.9.2 ADC digital Output Coding

Differential common mode voltage is compensated by the analog input stage. In addition, the differential input voltage is reduced by an analogue divider which is built with a differential operational amplifier. The feedback resistors of this operational amplifier determine the divider value. The two resistance values 4.7k Ω and 931 Ω build the divider of 5.0483. With the ADC LTC2323-16's maximum analogue differential input voltage range of ± 4.096 V a theoretical maximum of ± 20.677 V differential input voltage range is given for a TPCE636 analog input channel.

Due to the ADC's true differential inputs, the ADC output coding significantly differs compared to a single-ended input.

Analogue to a single-ended input, where the range setting directly describes the ground related input voltage range, the ADC range setting describes the range of ground related voltages that can be tied to the ADC differential inputs. This results in an extended input voltage range, since the ADC measures the voltage between the differential inputs VIN- and VIN+.

An Example: The TPCE636 voltage range is ± 10 V, so the allowed (single-ended, ground related) voltage on each ADC input pin is ± 10 V. When we examine the two largest differential voltages, we get following results:

VIN-	VIN+	ADC Input Value
-10V	+10V	+20V
+10V	-10V	-20V

Table 7-16: ADC Data Coding Example

The example shows that the range of differential ADC input values is -20V to +20V, which results to a full scale range of 40V for the ± 10 V ADC Input Range.

The TPCE636 data coding is two's complement.

Description	TPCE636	LTC2323-16	Digital Code
Full Scale Range	41.35470 V	8.192 V	-
Least Significant Bit	631 μ V	125 μ V	-
Full Scale (pos.)	20.67735 V	4.095875 V	0x7FFF
FSR - 1LSB	20.67672 V	4.095750 V	0x7FFE
Midscale + 1LSB	631 μ V	125 V	0x0001
Midscale	0.0 V	0.0 V	0x0000
Midscale - 1LSB	-631 μ V	-125 μ V	0xFFFF
-FSR + 1LSB	-20.67735 V	-4.095875 V	0x8001
Full Scale (neg.)	-20.67798 V	-4.096000 V	0x8000

Table 7-17: ADC Data Coding

7.9.3 User FPGA Pinning

Each ADC is connected to the User FPGA (Kintex™ 7) via a dedicated serial clocked Interface. Each ADC device has one input clock, one output clock and one conversion signal. For each ADC channel there is a respective data output line, so both ADC channel transfers data at the same time.

Signal	Bank	VCCO	Pin	Description
ADC_SCK_00+	12	2.5V	AB22	Differential Clock Output for ADC Channel 0 and 1
ADC_SCK_00-	12	2.5V	AC22	
ADC_SCKOUT_00+	12	2.5V	Y22	Differential Clock Input for ADC Channel 0 and 1
ADC_SCKOUT_00-	12	2.5V	AA22	
ADC_SDO1_00+	12	2.5V	AB21	Differential Data from ADC Channel 0
ADC_SDO1_00-	12	2.5V	AC21	
ADC_SDO2_00+	12	2.5V	AD23	Differential Data from ADC Channel 1
ADC_SDO2_00-	12	2.5V	AD24	
ADC_CNV_N_00	12	2.5V	AB26	Convert Signal for ADC Channel 0 and 1

Signal	Bank	VCCO	Pin	Description
ADC_SCK_01+	12	2.5V	AF24	Differential Clock Output for ADC Channel 2 and 3
ADC_SCK_01-	12	2.5V	AF25	
ADC_SCKOUT_01+	12	2.5V	AC23	Differential Clock Input for ADC Channel 2 and 3
ADC_SCKOUT_01-	12	2.5V	AC24	
ADC_SDO1_01+	12	2.5V	AD26	Differential Data from ADC Channel 2
ADC_SDO1_01-	12	2.5V	AE26	
ADC_SDO2_01+	12	2.5V	AD25	Differential Data from ADC Channel 3
ADC_SDO2_01-	12	2.5V	AE25	
ADC_CNV_N_01	12	2.5V	AC26	Convert Signal for ADC Channel 2 and 3

Signal	Bank	VCCO	Pin	Description
ADC_SCK_02+	12	2.5V	U26	Differential Clock Output for ADC Channel 4 and 5
ADC_SCK_02-	12	2.5V	V26	
ADC_SCKOUT_02+	12	2.5V	Y23	Differential Clock Input for ADC Channel 4 and 5
ADC_SCKOUT_02-	12	2.5V	AA24	
ADC_SDO1_02+	12	2.5V	U22	Differential Data from ADC Channel 4
ADC_SDO1_02-	12	2.5V	V22	
ADC_SDO2_02+	12	2.5V	U24	Differential Data from ADC Channel 5
ADC_SDO2_02-	12	2.5V	U25	
ADC_CNV_N_02	12	2.5V	W24	Convert Signal for ADC Channel 4 and 5

Signal	Bank	VCCO	Pin	Description
ADC_SCK_03+	12	2.5V	AA25	Differential Clock Output for ADC Channel 6 and 7
ADC_SCK_03-	12	2.5V	AB25	
ADC_SCKOUT_03+	12	2.5V	AA23	Differential Clock Input for ADC Channel 6 and 7
ADC_SCKOUT_03-	12	2.5V	AB24	
ADC_SDO1_03+	12	2.5V	V23	Differential Data from ADC Channel 6
ADC_SDO1_03-	12	2.5V	V24	
ADC_SDO2_03+	12	2.5V	V21	Differential Data from ADC Channel 7
ADC_SDO2_03-	12	2.5V	W21	
ADC_CNV_N_03	12	2.5V	W23	Convert Signal for ADC Channel 6 and 7

Signal	Bank	VCCO	Pin	Description
ADC_SCK_04+	13	2.5V	R18	Differential Clock Output for ADC Channel 8 and 9
ADC_SCK_04-	13	2.5V	P18	
ADC_SCKOUT_04+	13	2.5V	R22	Differential Clock Input for ADC Channel 8 and 9
ADC_SCKOUT_04-	13	2.5V	R23	
ADC_SDO1_04+	13	2.5V	T22	Differential Data from ADC Channel 8
ADC_SDO1_04-	13	2.5V	T23	
ADC_SDO2_04+	13	2.5V	R26	Differential Data from ADC Channel 9
ADC_SDO2_04-	13	2.5V	P26	
ADC_CNV_N_04	13	2.5V	N16	Convert Signal for ADC Channel 8 and 9

Signal	Bank	VCCO	Pin	Description
ADC_SCK_05+	13	2.5V	U17	Differential Clock Output for ADC Channel 10 and 11
ADC_SCK_05-	13	2.5V	T17	
ADC_SCKOUT_05+	13	2.5V	R21	Differential Clock Input for ADC Channel 10 and 11
ADC_SCKOUT_05-	13	2.5V	P21	
ADC_SDO1_05+	13	2.5V	T20	Differential Data from ADC Channel 10
ADC_SDO1_05-	13	2.5V	R20	
ADC_SDO2_05+	13	2.5V	U19	Differential Data from ADC Channel 11
ADC_SDO2_05-	13	2.5V	U20	
ADC_CNV_N_05	13	2.5V	U16	Convert Signal for ADC Channel 10 and 11

Signal	Bank	VCCO	Pin	Description
ADC_SCK_06+	13	2.5V	K25	Differential Clock Output for ADC Channel 12 and 13
ADC_SCK_06-	13	2.5V	K26	
ADC_SCKOUT_06+	13	2.5V	N21	Differential Clock Input for ADC Channel 12 and 13
ADC_SCKOUT_06-	13	2.5V	N22	
ADC_SDO1_06+	13	2.5V	P24	Differential Data from ADC Channel 12
ADC_SDO1_06-	13	2.5V	N24	
ADC_SDO2_06+	13	2.5V	M25	Differential Data from ADC Channel 13
ADC_SDO2_06-	13	2.5V	L25	
ADC_CNV_N_06	13	2.5V	M21	Convert Signal for ADC Channel 12 and 13

Signal	Bank	VCCO	Pin	Description
ADC_SCK_07+	13	2.5V	T24	Differential Clock Output for ADC Channel 14 and 15
ADC_SCK_07-	13	2.5V	T25	
ADC_SCKOUT_07+	13	2.5V	P23	Differential Clock Input for ADC Channel 14 and 15
ADC_SCKOUT_07-	13	2.5V	N23	
ADC_SDO1_07+	13	2.5V	N26	Differential Data from ADC Channel 14
ADC_SDO1_07-	13	2.5V	M26	
ADC_SDO2_07+	13	2.5V	R25	Differential Data from ADC Channel 15
ADC_SDO2_07-	13	2.5V	P25	
ADC_CNV_N_07	13	2.5V	M22	Convert Signal for ADC Channel 14 and 15

Table 7-18: ADC Interface Connections

To use the clocked serial interface between the User FPGA (Kintex™ 7) and one of the eight LTC2323-16 ADC devices please use the LTC2323-16 data sheet which describes the communication process.

7.9.4 Programming Hints LTC2323-16

The LTC2323-16 digital interface is a simple clocked SPI based interface.

This differential interface uses differential LVDS signals for serial data transfer. All LVDS signals need a termination on the receiver side of the connection. For the SCK± an external resistor is implemented on the TPMC636. The User FPGA inputs CLKOUT±, SDO1± and SDO2± of each ADC channel need an FPGA internal termination. The corresponding constraints for the pin assignment, the I/O standard, termination and slew rate is specified in Appendix A.

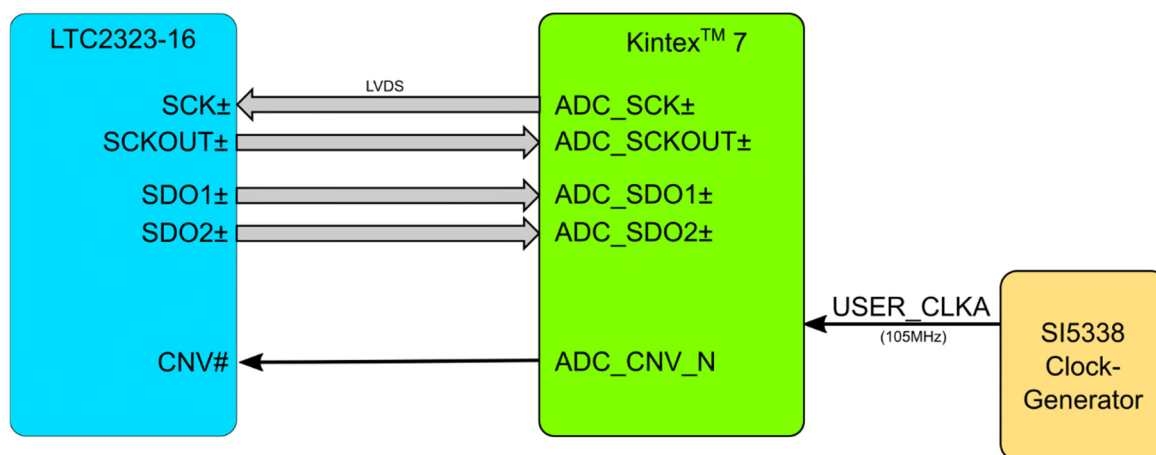


Figure 7-8 : Digital ADC to FPGA Interface

A conversion is triggered by a negative edge on the CNV# line. The acquisition is done during the positive phase of the CNV# signal. Following the FPGA drives the SCK clock, which then initiates the data transfer from the ADC to the FPGA. The ADC then transmits the serial data SDO1 / SDO2 synchronous to CLKOUT. The data sequence is MSB first and the LSB at least.

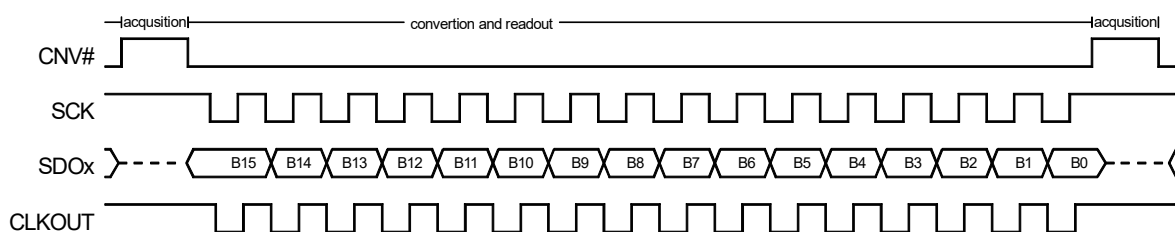


Figure 7-9 : Timing Diagram LTC2323-16

Note, that the one-cycle conversion latency has the result that the previous sample word is transmitted first. That means, at the beginning of a burst sampling period the first conversion result will be invalid.

For a detailed description of the LTC2323-16 interface and the LTC2323-16 function please use the data sheet which describes the whole communication process and all special characteristics of the ADC.

7.10 Parallel DAC Interface

7.10.1 Overview

The 16 analog DAC outputs of the TPCE636 are realized with eight AD5547 16bit Dual-Current DAC devices. Each of these DACs has two DAC channels. Thus, a total of 16 DAC channels are available on the TPCE636. Because of current output DACs it is necessary to use operational amplifier for each DAC output channel to generate an output voltage range up to $\pm 10V$.

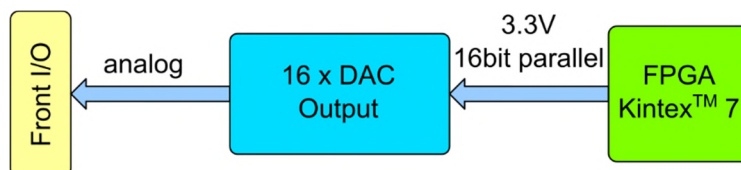


Figure 7-10 : Analog Output Section

As programming interface, a 16bit parallel Bus is implemented. Respectively four DAC devices (8 DAC channels) share this Kintex™ 7 16bit data bus. To set each output individually each DAC device has its control interface.

The following figure shows the structure and principle of two DAC outputs. Both are connected via independent operational amplifiers to the TPCE636 I/O Connector.

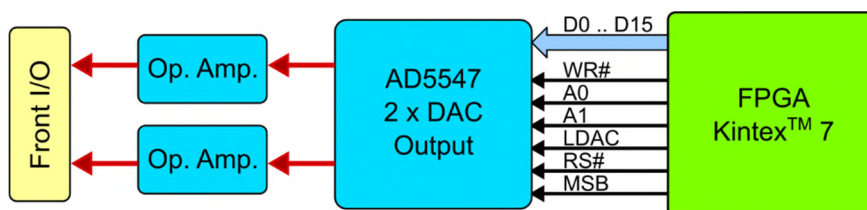


Figure 7-11 : Analog Output Section

The key-features of the TPCE636 DAC Interface are:

- 16bit Resolution
- Built-in 4-quadrant resistors in combination with an operational amplifier allow $\pm 10V$ outputs
- Outputs Drive $\pm 10mA$ per channel
- Capacitive Load Driving = 1000pF

7.10.2 User FPGA Pinning

First DAC configuration interface is for controlling the DAC channel 0 up to 7.

Signal	Bank	VCCO	Pin	Description
DAC_D0	15	3.3V	D20	First DAC Data Bus to DAC Device 0 .. 3 with DAC Channel 0 .. 7
DAC_D1	15	3.3V	E20	
DAC_D2	15	3.3V	H17	
DAC_D3	15	3.3V	D19	
DAC_D4	15	3.3V	G16	
DAC_D5	15	3.3V	F19	
DAC_D6	15	3.3V	F20	
DAC_D7	15	3.3V	E17	
DAC_D8	15	3.3V	E18	
DAC_D9	15	3.3V	D15	
DAC_D10	15	3.3V	D16	
DAC_D11	15	3.3V	C16	
DAC_D12	15	3.3V	C18	
DAC_D13	15	3.3V	B16	
DAC_D14	15	3.3V	B19	
DAC_D15	15	3.3V	A17	
DAC_ADR0	15	3.3V	E16	DAC Address Line to select the DAC channel A or B from one DAC Device.
DAC_'WR00_01'	15	3.3V	F15	A low active WR transfers data to DAC input register. One write signal for each DAC device respectively for two DAC channel.
DAC_'WR02_03'	15	3.3V	C19	
DAC_'WR04_05'	15	3.3V	J16	
DAC_'WR06_07'	15	3.3V	E15	
DAC_LDAC00_01	14	3.3V	H26	Load the DAC output register with contents of the input register. One write signal for each DAC device respectively for two DAC channel.
DAC_LDAC02_03	14	3.3V	H21	
DAC_LDAC04_05	14	3.3V	G21	
DAC_LDAC06_07	14	3.3V	H23	

Second DAC configuration interface is for controlling the DAC channel 8 up to 15.

Signal	Bank	VCCO	Pin	Description
DAC_D16	15	3.3V	G15	Second DAC Data Bus to DAC Device 4 .. 7 with DAC Channel 8 .. 15
DAC_D17	15	3.3V	H18	
DAC_D18	15	3.3V	K15	
DAC_D19	15	3.3V	K20	
DAC_D20	15	3.3V	J18	
DAC_D21	15	3.3V	J19	
DAC_D22	15	3.3V	J20	
DAC_D23	15	3.3V	A19	
DAC_D24	15	3.3V	B17	
DAC_D25	15	3.3V	C17	
DAC_D26	15	3.3V	L20	
DAC_D27	15	3.3V	H16	
DAC_D28	15	3.3V	G20	
DAC_D29	15	3.3V	A18	
DAC_D30	15	3.3V	G17	
DAC_D31	15	3.3V	D18	
DAC_ADR1	15	3.3V	G19	DAC Address Line to select the DAC channel A or B from one DAC Device.
DAC_'WR08_09'	15	3.3V	L19	A low active WR transfers data to DAC input register. One write signal for each DAC device respectively for two DAC channel.
DAC_'WR10_11'	15	3.3V	H19	
DAC_'WR12_13'	15	3.3V	K16	
DAC_'WR14_15'	15	3.3V	K17	
DAC_LDAC08_09	14	3.3V	H24	Load the DAC output register with contents of the input register. One write signal for each DAC device respectively for two DAC channel.
DAC_LDAC10_11	14	3.3V	J21	
DAC_LDAC12_13	14	3.3V	H22	
DAC_LDAC14_15	14	3.3V	J24	

The following signals are used for all 16 DAC channels.

Signal	Bank	VCCO	Pin	Description
DAC_RS#	12	2.5V	W25	Active low resets all 16 input and output DAC registers. Value depends on DAC_MSB line.
DAC_MSB	12	2.5V	W26	MSB Power-On Reset State. DAC_MSB = 0 corresponds to zero- scale reset; DAC_MSB = 1 corresponds to midscale reset

Table 7-19: TPCE636 parallel DAC Interface

7.10.3 Programming Hints for AD5547

TPCE636 DAC Channel write to DAC Input Register Decoding.

ADR0	'WR00_01'	'WR02_03'	'WR04_05'	'WR06_07'	DAC Channel
0	0	1	1	1	Channel 0
1	0	1	1	1	Channel 1
0	1	0	1	1	Channel 2
1	1	0	1	1	Channel 3
0	1	1	0	1	Channel 4
1	1	1	0	1	Channel 5
0	1	1	1	0	Channel 6
1	1	1	1	0	Channel 7

ADR1	'WR08_09'	'WR10_11'	'WR12_13'	'WR14_15'	DAC Channel
0	0	1	1	1	Channel 8
1	0	1	1	1	Channel 9
0	1	0	1	1	Channel 10
1	1	0	1	1	Channel 11
0	1	1	0	1	Channel 12
1	1	1	0	1	Channel 13
0	1	1	1	0	Channel 14
1	1	1	1	0	Channel 15

TPCE636 DAC Channel LOAD_DAC Register Decoding.

ADR0	'LDAC00_01'	'LDAC02_03'	'LDAC04_05'	'LDAC06_07'	DAC Channel
0	1	0	0	0	Channel 0
1	1	0	0	0	Channel 1
0	0	1	0	0	Channel 2
1	0	1	0	0	Channel 3
0	0	0	1	0	Channel 4
1	0	0	1	0	Channel 5
0	0	0	0	1	Channel 6
1	0	0	0	1	Channel 7

ADR1	'LDAC09_09'	'LDAC10_11'	'LDAC12_13'	'LDAC14_15'	DAC Channel
0	1	0	0	0	Channel 8
1	1	0	0	0	Channel 9
0	0	1	0	0	Channel 10
1	0	1	0	0	Channel 11
0	0	0	1	0	Channel 12
1	0	0	1	0	Channel 13
0	0	0	0	1	Channel 14
1	0	0	0	1	Channel 15

A detailed description of the AD5547 parallel interface and the AD5547 function please use the data sheet which describes the whole data transfer, data register and output process and all special characteristics of the DAC.

7.10.4 Output Voltage Range

The output voltage ranges of the TPCE636 DAC outputs are set via *DAC Control / Status Register* and *DAC Output Voltage Range Register*.

There are three predefined output voltage ranges $\pm 10V$, $\pm 5V$, $\pm 2,5V$ and a fourth mode in which the high and low voltage range can be set individually.

The first three predefined fixed voltage ranges were calibrated as part of the TEWS factory test. Determined correction values were stored in an I2C EEPROM. There are no correction values for the individually adjustable voltage range mode.

Due to tolerances of the reference voltage generation, basic tolerances of the DAC components and temperature dependence, it may happen that the limits for the output voltage cannot be reached.

7.11 Digital Interface to FireFly Connector

The digital I/O Pins of the TPCE636 Firefly Back I/O connector are directly routed to the User FPGA (Kintex™ 7). The I/O functions of these FPGA pins are directly dependent on the configuration of the FPGA.

The Kintex™ 7 VCCO voltage for these pins is set to 2.5V, so only the 2.5V I/O standards LVCMOS25, LVTTTL25 and LVDS25 are possible when using the TPCE636 FireFly Back I/O interface.

Signal Name	Bank	VCCO	Pin	IO Standard for example
DIG_IO_00+	13	2.5V	M24	LVDS25
DIG_IO_00-	13	2.5V	L24	LVDS25
DIG_IO_01+	16	2.5V	E10	LVDS25
DIG_IO_01-	16	2.5V	D10	LVDS25
DIG_IO_02+	13	2.5V	N19	LVDS25
DIG_IO_02-	13	2.5V	M20	LVDS25
DIG_IO_03+	12	2.5V	Y25	LVDS25
DIG_IO_04-	12	2.5V	Y26	LVDS25

Table 7-20 : FireFly Back I/O Interface

7.12 JTAG Controller to K7 JTAG Interface

The BCC provides two methods for accessing the USER FPGA via JTAG: Bit-I/O and Vector-I/O.

7.12.1 Bit-I/O

The Bit-I/O Interface is a bit-centric interface that allows stimulating the JTAG interface by setting and reading each line (TCK, TMS, TDI and TDO) of the JTAG interface independently.

All Bit-I/O register bits (JTAG_BIO_*) are directly linked/connected to the BCC JTAG Interface. If the Bit-I/O Interface is enabled (JTAG_BIO_EN = 1), these bits drive the corresponding JTAG lines.

Note that the flexibility comes along with a higher I/O effort since every signal constellation must be set and emitted on the JTAG interface via a TCK high and low sequence with separated register accesses.

7.12.2 Vector-I/O

The Vector-I/O Interface focuses on a high abstract level where operations are performed semi-automatically based on vectors and transfer lengths.

Via the register interface TMS (JTAG_VIO_TMS_DATA) and TDI data (JTAG_VIO_TDI_DATA) vectors are defined that shall be shifted-out onto the JTAG interface. These vectors are coupled in a way that the data (bit-information) is shifted-out at the same time.

Note that bit #0 is the first one that appears on the JTAG interface.

Shift operations require the transfer length (JTAG_VIO_XFER_LEN) information. The value has to be set in accordance to the size of the TMS/TDI data vectors or TDO read size and defines the number of transfer cycles or rather the number of TCK cycles (rising and falling edges with 50% duty factor) occurring on the JTAG interface.

The actual TCK I/O clock rate is adjustable (JTAG_VIO_TCK_CLK_DIV).

If the Vector-I/O Interface is enabled (JTAG_VIO_EN = 1) and while the Vector-I/O Controller is idle (JTAG_VIO_CTRL_STAT = 0b01), write (JTAG_VIO_SHIFT_REQ) or read (JTAG_VIO_GET_REQ) operations can be initiated. Requests initiated while the Vector-I/O Controller is not idle are lost.

Note that there is no check regarding the number of bits shifted-out or read-in. Hence illegal data can be shifted-out or read-in if the transfer length is not set appropriately.

Bits of the output data vectors are aligned to the beginning of a TCK cycle and are hence updated after a falling edge. If more data is shifted-out than defined, zero bits ('0') are transferred, which is useful in JTAG IR- or DR-Shift States.

Data is read-in with every rising edge during shift-out and get-requests. This also allows obtaining the bit-information provided in response of a TMS/TDI operation.

Read-in updates appear immediately on the JTAG Vector-I/O TDO data vector (JTAG_VIO_TDO_DATA).

Note that bit #0 is the last one that has been read-in from the JTAG interface.

7.13 I2C Bridge

The I2C bridge mode allows a unidirectional I2C bus BCC reach-through from the USER FPGA I2C Bus onto the BCCs isolated management BCC I2C bus. In consequence of that, the two-independent busses behave like one.

Unidirectional means in this context that only the SDA signal state is bidirectional (transferred to through the BCC) but not the SCL. SCL only traverses from USER FPGA to management BCC I2C bus.

Having this feature active, allows accessing the TMP441 temperature sensor and the SI5338 clock generator device. Both are accessible via their normal (original) I2C addresses.

Note that the Automatic Temperature Read Mode (TMP441_AUTO_TRD_EN) must be disabled before the bridge mode can be enabled.

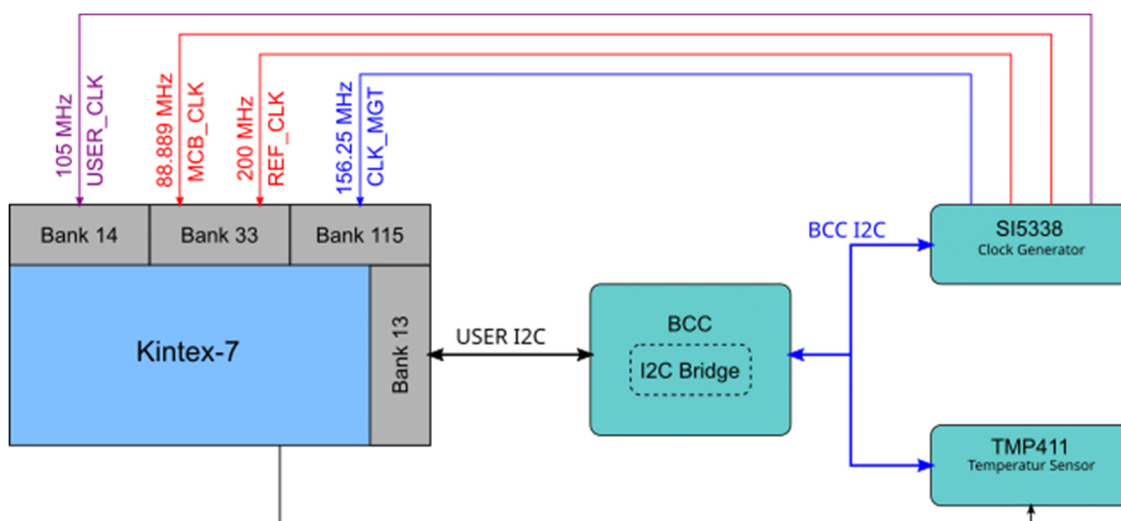


Figure 7-12 : User FPGA I2C to BCC I2C Bridge

For function description and register contents of the clock generator Si5338 and the temperature sensor TMP411 please use the data sheets of the respective device.



Do not change the SI5338 Output driver properties like "Format and Voltage"

Channel 0 must be 3.3V LVDS

Channel 1 must be 3.3V SSTL on A and B

Channel 2 must be 3.3V CMOS on A and B

Channel 4 must be 3.3V SSTL on A and B



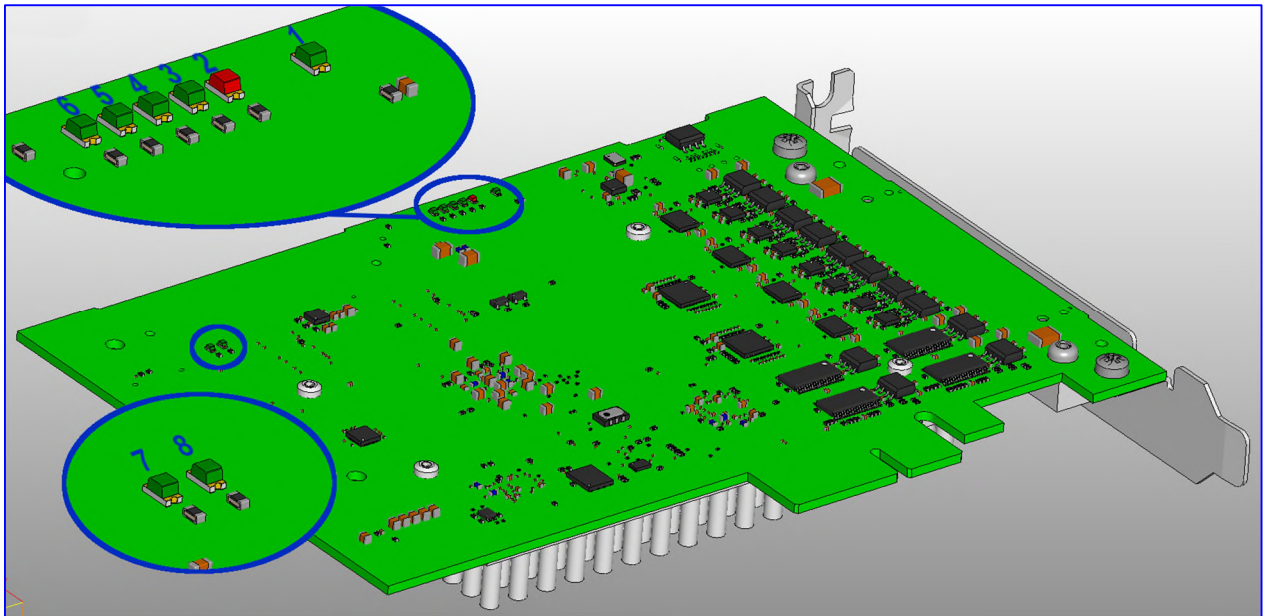
Do not change Si5338 Core VDD or I2C Bus Voltage

Core VDD = 3.3V

I2C Bus Voltage = 2.5V or 3.3V

7.14 On-Board Indicators

The TPCE636 provides a couple of board-status LEDs as shown below. These include Power-Good and FPGA configuration status indications as well as two general purpose LEDs.



No.	LED	Color	State		Description
1	Power Good	Green	off	On-Board Power Supplies are not ok	Power Good Signal for all on-board power supplies.
			on	On-Board Power Supplies are all ok	
2	User INIT	Red	off	INIT state is inactive	User FPGA (Kintex™ 7) INIT# - Pin LED. Indicates FPGA configuration
			on	INIT state is active	
3	BCC DONE	Green	off	Device is not configured	Board Configuration FPGA DONE-Pin LED Indicates successful FPGA configuration and initiation.
			flashing	Device is completely configured	
			on	Device is completely configured	
4	User DONE	Green	off	Device is not configured	User FPGA (Kintex™ 7) DONE-Pin LED. Indicates successful FPGA configuration
			on	Device is completely configured	
5	GPIO LED1	Green	off	General State is not IDLE	Configuration FPGA depends.
			on	General State is IDLE	
6	GPIO LED2	Green	off	PCI Reset is active	
			on	PCI Reset is inactive	
7	USER LED1	Green	-		Design dependent, can be controlled by the User FPGA.
8	USER LED2	Green	-		Refer to chapter "User-GPIO"

Table 7-21: Board-Status and User LEDs

7.14.1 User FPGA Pinning

General purpose I/O connected to the User FPGA Kintex™ 7.

Signal	Bank	VCCO	Pin	Description
USER_LED0	14	3.3V	E26	2x green on-board LEDs
USER_LED1			J26	

Table 7-22: TPCE636 User On-Board Indicators

7.15 User FPGA Reset Inputs

General purpose Reset input connected to the User FPGA Kintex™ 7.

Signal	Bank	VCCO	Pin	Description
DWNRST#	14	3.3V	K21	Reset from PCIe Switch Based directly on the downstream reset of the PCIe switch on the TPCE636. See the PCIe switch data sheet for more information.
FPGA_RST	14	3.3V	L23	Reset Input from BCC The reset signal is valid as long as the BCC is in its configuration phase. When all configuration processes on the TPCE636 are completed, this signal is deactivated.

Table 7-23: User FPGA Reset Inputs

8 Design Help

8.1 Board Reference Design

User applications for the TPCE636 may be developed by using the TPCE636 FPGA Board Reference Design.

TEWS offers this Board Reference Design as a well-documented basic example. It includes an .xdc constrain file with all necessary pin assignments and basic timing constraints. The example design covers the main functionalities of the TPCE636. It implements a PCIe endpoint with interrupt support, register mapping, DDR3 memory access and basic I/O functions. It comes as a Xilinx Vivado 2017.3.1 project with source code and as a ready-to-download bit stream. This example design can be used as a starting point for own projects.

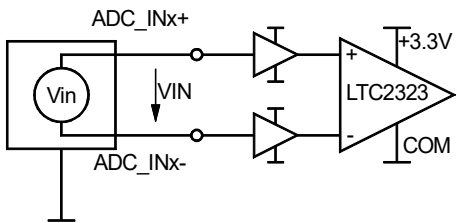
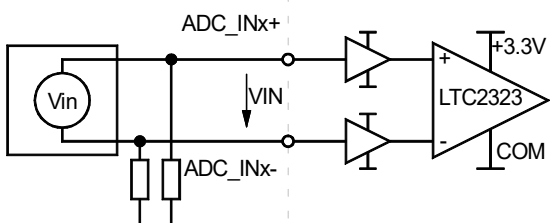
The TPCE636 FPGA Application design can be developed using the design software Vivado Design Suite. Licenses for design tools are required.

For TPCE636 FPGA Board Reference Design also see the included User Manual.

9 I/O Interfaces

9.1.1 Front I/O - ADC Analog Input Level

All analog inputs are connected via an impedance converter and a second operation amplifier for level adjustment and filtering to the differential ADC inputs. This also serves as a protection of the ADC from excessive analog input levels.

Diff. Source with Ground Reference	Diff. Source without Ground Reference
	
Maximum VIN = ±20.0V	Maximum VIN = ±20.0V

The TPCE636 has differential analog inputs. When talking about the input voltage range of a differential input, one has to differentiate between the differential input voltage between the two pins, and the input voltage relative to ground for each pin.

With an input voltage range of ±10V (ground related) for each pin of the differential input, we get the ±20V differential input voltages:

Voltage	Description	TPCE636
V_{diff_max}	Maximum differential input voltage (range)	±20.0V
V_{cm_max}	Maximum common mode input voltage (range)	±0V
V_{in_max}	Operating voltage range of the input pins	±10.0V
V_{in_abs}	Absolute maximum voltage range of the input	Power Off: ±5.0V Power On: ±18.5V

Table 9-1 : Differential Input Voltage Ranges

The range of this differential input is -20V to +20V, which results to a full scale range of 40V for the ±10V (per pin, relative to ground) Input voltage range.

9.1.2 Front I/O – Analog Output Level

All analog outputs of the AD5547 DAC are routed through operational amplifiers to front I/O connector.

Outputs Drive Current	±10mA
Capacitive Load	1000pF

Table 9-2 : DAC Electrical Interface

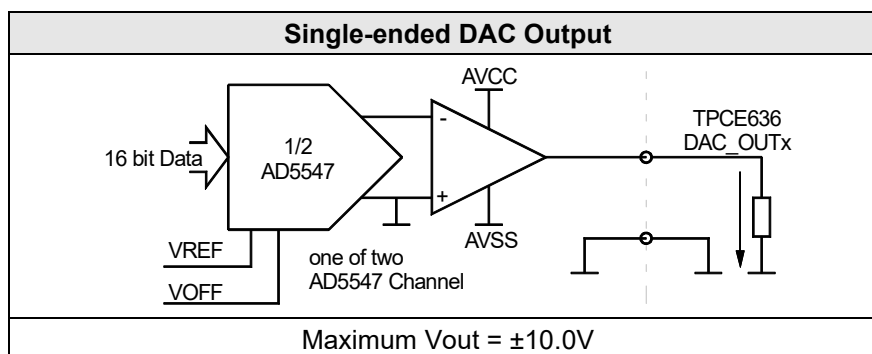


Figure 9-1 : DAC Output Interface

The output voltage range of each DAC channel can be set via VREF and VOFF.

See also the chapter about the configuration of the DAC output voltage in the BCC Register Description.

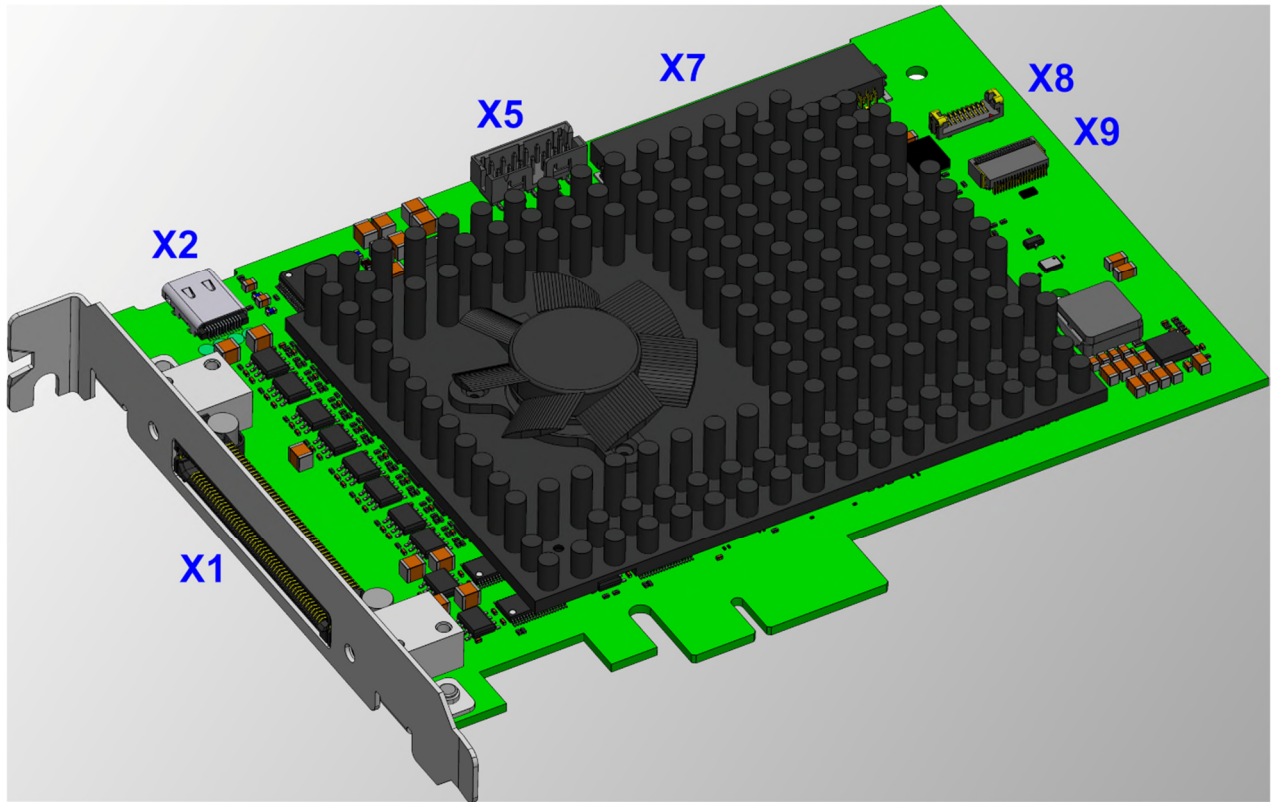
9.1.3 Back I/O Interface

All 64 single-ended / 32 differential digital back I/O Pins of the TPCE636 are directly routed from the User FPGA (Kintex™ 7) to the 68 pin ERNI flat cable connector. The I/O functions of these FPGA pins are directly dependent on the configuration of the FPGA.

The Kintex™ 7 VCCO voltage is set to 2.5V, so only the 2.5V I/O standards LVCMOS25 and LVDS25 are possible when using the TPCE636 back I/O interface.

10 I/O Description

10.1 Overview



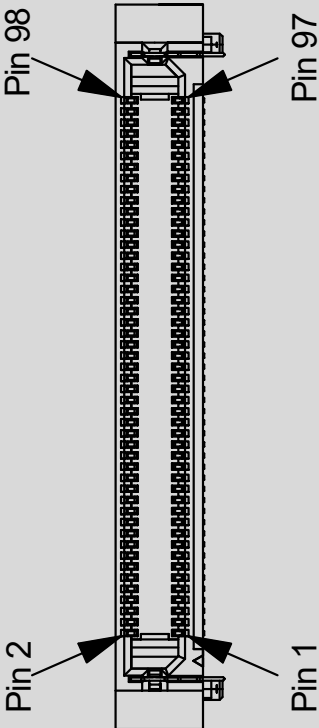
10.2 Front I/O Connector (X1)

10.2.1 Connector Type

Pin-Count	98
Connector Type	Rugged EdgeRate female connector
Source & Order Info	Samtec – ERF8-049-01-L-D-RA-L

Table 10-1 : Front I/O Connector

10.2.2 Pin Assignment

Pin	I/O	Connector View	Pin	I/O
1	GND		2	GND
3	ADC_IN1+		4	DAC_OUT1
5	ADC_IN1-		6	GND
7	GND		8	GND
9	ADC_IN2+		10	DAC_OUT2
11	ADC_IN2-		12	GND
13	GND		14	GND
15	ADC_IN3+		16	DAC_OUT3
17	ADC_IN3-		18	GND
19	GND		20	GND
21	ADC_IN4+		22	DAC_OUT4
23	ADC_IN4-		24	GND
25	GND		26	GND
27	ADC_IN5+		28	DAC_OUT5
29	ADC_IN5-		30	GND
31	GND		32	GND
33	ADC_IN6+		34	DAC_OUT6
35	ADC_IN6-		36	GND
37	GND		38	GND
39	ADC_IN7+		40	DAC_OUT7
41	ADC_IN7-		42	GND
43	GND		44	GND
45	ADC_IN8+		46	DAC_OUT8
47	ADC_IN8-		48	GND
49	GND		50	GND
51	ADC_IN9+		52	DAC_OUT9
53	ADC_IN9-		54	GND
55	GND		56	GND
57	ADC_IN10+		58	DAC_OUT10
59	ADC_IN10-		60	GND
61	GND		62	GND
63	ADC_IN11+		64	DAC_OUT11

Pin	I/O	Connector View	Pin	I/O
65	ADC_IN11-		66	GND
67	GND		68	GND
69	ADC_IN12+		70	DAC_OUT12
71	ADC_IN12-		72	GND
73	GND		74	GND
75	ADC_IN13+		76	DAC_OUT13
77	ADC_IN13-		78	GND
79	GND		80	GND
81	ADC_IN14+		82	DAC_OUT14
83	ADC_IN14-		84	GND
85	GND		86	GND
87	ADC_IN15+		88	DAC_OUT15
89	ADC_IN15-		90	GND
91	GND		92	GND
93	ADC_IN16+		94	DAC_OUT16
95	ADC_IN16-		96	GND
97	GND		98	GND

Table 10-2: Pin Assignment Front Panel I/O Connector

10.3 Digital Back I/O Connector (X7)

10.3.1 Connector Type

Pin-Count	68
Connector Type	ERNI SMC, right angle male, 1.27mm pitch
Source & Order Info	ERNI 154766

Table 10-3 : Back I/O Connector

10.3.2 Pin Assignment

Pin	differential I/O		Pin	differential I/O
a1	BACK_IO0+		b1	BACK_IO0-
a2	BACK_IO1+		b2	BACK_IO1-
a3	BACK_IO2+		b3	BACK_IO2-
a4	BACK_IO3+		b4	BACK_IO3-
a5	BACK_IO4+		b5	BACK_IO4-
a6	BACK_IO5+		b6	BACK_IO5-
a7	BACK_IO6+		b7	BACK_IO6-
a8	BACK_IO7+		b8	BACK_IO7-
a9	BACK_IO8+		b9	BACK_IO8-
a10	BACK_IO9+		b10	BACK_IO9-
a11	BACK_IO10+		b11	BACK_IO10-
a12	BACK_IO11+		b12	BACK_IO11-
a13	BACK_IO12+		b13	BACK_IO12-
a14	BACK_IO13+		b14	BACK_IO13-
a15	BACK_IO14+		b15	BACK_IO14-
a16	BACK_IO15+		b16	BACK_IO15-
a17	BACK_IO16+		b17	BACK_IO16-
a18	BACK_IO17+		b18	BACK_IO17-
a19	BACK_IO18+		b19	BACK_IO18-
a20	BACK_IO19+		b20	BACK_IO19-
a21	BACK_IO20+		b21	BACK_IO20-
a22	BACK_IO21+		b22	BACK_IO21-
a23	BACK_IO22+		b23	BACK_IO22-
a24	BACK_IO23+		b24	BACK_IO23-
a25	BACK_IO24+		b25	BACK_IO24-
a26	BACK_IO25+		b26	BACK_IO25-
a27	BACK_IO26+		b27	BACK_IO26-
a28	BACK_IO27+		b28	BACK_IO27-

Pin	differential I/O		Pin	differential I/O
a29	BACK_IO28+		b29	BACK_IO28-
a30	BACK_IO29+		b30	BACK_IO29-
a31	BACK_IO30+		b31	BACK_IO30-
a32	BACK_IO31+		b32	BACK_IO31-
a33	GND		b33	GND
a34	GND		b34	GND

Figure 10-1 : Pin Assignment Back I/O Connector TPCE636

10.4 MGT Back I/O Connector (X8/X9)

The MGT Back I/O Connector is a Samtec Firefly Micro Flyover Connector which consists of two parts. One part is the connector for the high speed signals (UEC5) and the second part (UCC8) is for power and control signals.

10.4.1 Connector Type

Pin-Count	38
Connector Type	Firefly Micro Flyover Connector
Source & Order Info	Samtec – UEC5-019-1-H-D-RA-2-A

Figure 10-2 : Firefly Back I/O Connector TPCE636

10.4.2 Pin Assignment

Pin	UEC5 - A	UEC5 - B
1	GND	GND
2	Tx0-	Tx1-
3	TX0+	Tx1+
4	GND	GND
5	Tx2-	Tx3-
6	Tx2+	Tx3+
7	GND	GND
8	DIG_IO_0-	DIG_IO_1-
9	DIG_IO_0+	DIG_IO_1+
10	GND	GND
11	DIG_IO_2+	DIG_IO_3+
12	DIG_IO_2-	DIG_IO_3-
13	GND	GND
14	Rx3+	Rx2+
15	Rx3-	Rx2-
16	GND	GND
17	Rx1+	Rx0+
18	Rx1-	Rx0-
19	GND	GND

Figure 10-3 : Pin Assignment Firefly Back I/O Connector TPCE636

10.4.3 Connector Type

Pin-Count	10
Connector Type	Firefly Micro Flyover Connector
Source & Order Info	Samtec – UCC8-010-1-H-S-2-A

Figure 10-4 : Firefly Back I/O Connector TPCE636

10.4.4 Pin Assignment

Pin	UCC8
1	+3.3V
2	GND
3	PRESENTL
4	SELECTL
5	INTL
6	RESETL
7	SDA
8	SCL
9	n.c.
10	+3.3V

Figure 10-5 : Pin Assignment Firefly Back I/O Connector TPCE636

10.5 FPGA JTAG Header (X5)

This header directly connects a JTAG interface cable to the JTAG pins to the on-board User FPGA JTAG chain. The pinout of this header is designed in conjunction with the Xilinx Platform Cable USB II. This allows the direct usage of Xilinx software-tools like Vivado Logic Analyzer or the Vivado Hardware Manager.

10.5.1 Connector Type

Pin-Count	14
Connector Type	2.00 mm Pitch Milli-Grid™ Header
Source & Order Info	Molex 87832-1420 or compatible

Figure 10-6 JTAG Header TPCE636

10.5.2 Pin Assignment

Pin	Signal	Description
1	NC	Not Connected
2	V _{REF}	JTAG Reference Voltage (3.3V)
3	GND	Ground
4	TMS	Test Mode Select Input
5	GND	Ground
6	TCK	Test Clock
7	GND	Ground
8	TDO	Test Data Output (TAP Controller: TDI)
9	GND	Ground
10	TDI	Test Data Input (TAP Controller: TDO)
11	GND	not connected on the TPCE636
12	TRST#	not connected on the TPCE636
13	PGND	Used on TXMC635 for XILINX Header present detection
14	NC	HALT_INIT_WP signal. Optional. Not connected on the TPCE636

Table 10-4: Pin Assignment JTAG Header

10.6 FPGA USB Connector (X2)

10.6.1 Connector Type

Pin-Count	24
Connector Type	Würth: USB Typ C, USB 3,1, 24pol. SMT/THT 90° or compatible
Source & Order Info	632 723 300 011

Figure 10-7 FPGA USB Connector TPCE636

10.6.2 Pin Assignment

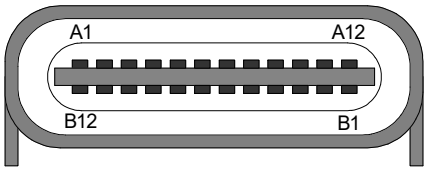
 <p>USB Typ C</p>	Pin	Description	Pin	Description
	A1	GND	B12	GND
	A2	n.c.	B11	n.c.
	A3	n.c.	B10	n.c.
	A4	VBUS	B9	VBUS
	A5	n.c.	B8	n.c.
	A6	D+	B7	D-
	A7	D-	B6	D+
	A8	n.c.	B5	n.c.
	A9	VBUS	B4	VBUS
	A10	n.c.	B3	n.c.
	A11	n.c.	B2	n.c.
	A12	GND	B1	GND

Table 10-5: Pin Assignment USB Type C Connector

11 Appendix A

This appendix contains the signal to pin assignments for the User FPGA Kintex™ 7.

```
## #####
## #####
##                                TEWS TECHNOLOGIES                                ##
## #####
##
## Project Name       : TPCE636
## File Name         : tpce636.xdc
## Target Device      : XC7KxxxT-FBG676-2
## Design Tool        : Xilinx Vivado Design Suite Design Edition 2017.3
## Simulation Tool    :
##
## Description        : Physical constraints for TPCE636_BRD FPGA/K7
##
## Owner              : TEWS TECHNOLOGIES GmbH
##                    : Am Bahnhof 7
##                    : D-25469 Halstenbek
##
##                    : Tel.: +49 / (0)4101 / 4058-0
##                    : Fax.: +49 / (0)4101 / 4058-19
##                    : e-mail: support@tews.com
##
##                    : Copyright (c) 2018
##                    : TEWS TECHNOLOGIES GmbH
##
## History            :
##   Version 1        : (AW, 2018-04-10)
##                    : Initial Version
##
## Comments           : none
##
## #####
## #####
## Section: MGT
## #####

# PCIe Lanes
set_property PACKAGE_PIN F1 [get_ports {PER_N[3]}]
set_property PACKAGE_PIN F2 [get_ports {PER_P[3]}]
set_property PACKAGE_PIN G3 [get_ports {PET_N[3]}]
set_property PACKAGE_PIN G4 [get_ports {PET_P[3]}]

set_property PACKAGE_PIN D1 [get_ports {PER_N[2]}]
set_property PACKAGE_PIN D2 [get_ports {PER_P[2]}]
set_property PACKAGE_PIN E3 [get_ports {PET_N[2]}]
set_property PACKAGE_PIN E4 [get_ports {PET_P[2]}]

set_property PACKAGE_PIN B1 [get_ports {PER_N[1]}]
set_property PACKAGE_PIN B2 [get_ports {PER_P[1]}]
set_property PACKAGE_PIN C3 [get_ports {PET_N[1]}]
set_property PACKAGE_PIN C4 [get_ports {PET_P[1]}]

set_property PACKAGE_PIN A3 [get_ports {PER_N[0]}]
set_property PACKAGE_PIN A4 [get_ports {PER_P[0]}]
set_property PACKAGE_PIN B5 [get_ports {PET_N[0]}]
set_property PACKAGE_PIN B6 [get_ports {PET_P[0]}]

# PCIe Reference Clock
set_property PACKAGE_PIN D5 [get_ports REFCLK_02_N]
set_property PACKAGE_PIN D6 [get_ports REFCLK_02_P]

# Firefly
set_property PACKAGE_PIN P2 [get_ports MGTTX0_P]
set_property PACKAGE_PIN P1 [get_ports MGTTX0_N]
set_property PACKAGE_PIN R4 [get_ports MGTRX0_P]
set_property PACKAGE_PIN R3 [get_ports MGTRX0_N]
```

```

set_property PACKAGE_PIN M2 [get_ports MGTTX1_P]
set_property PACKAGE_PIN M1 [get_ports MGTTX1_N]
set_property PACKAGE_PIN N4 [get_ports MGTRX1_P]
set_property PACKAGE_PIN N3 [get_ports MGTRX1_N]

set_property PACKAGE_PIN K2 [get_ports MGTTX2_P]
set_property PACKAGE_PIN K1 [get_ports MGTTX2_N]
set_property PACKAGE_PIN L4 [get_ports MGTRX2_P]
set_property PACKAGE_PIN L3 [get_ports MGTRX2_N]

set_property PACKAGE_PIN H2 [get_ports MGTTX3_P]
set_property PACKAGE_PIN H1 [get_ports MGTTX3_N]
set_property PACKAGE_PIN J4 [get_ports MGTRX3_P]
set_property PACKAGE_PIN J3 [get_ports MGTRX3_N]

# Reference Clock for Firefly MGTs
set_property PACKAGE_PIN H6 [get_ports CLK_MGT_P]
set_property PACKAGE_PIN H5 [get_ports CLK_MGT_N]

## #####
## Section: BCC
## #####

#set_property PACKAGE_PIN L23 [get_ports FPGA_RST_n]
#set_property IOSTANDARD LVCMOS33 [get_ports FPGA_RST_n]

## #####
## Section: PCIe Switch
## #####

set_property IOSTANDARD LVCMOS33 [get_ports DWN_RST_n]
set_property PACKAGE_PIN K21 [get_ports DWN_RST_n]

set_property PULLUP true [get_ports DWN_RST_n]

set_false_path -from [get_ports DWN_RST_n]

## #####
## Section: DDR3
## #####

# DDR3 Data (DQ)
set_property SLEW FAST [get_ports {DQ[0]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[0]}]
set_property PACKAGE_PIN AF17 [get_ports {DQ[0]}]

set_property SLEW FAST [get_ports {DQ[1]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[1]}]
set_property PACKAGE_PIN AF14 [get_ports {DQ[1]}]

set_property SLEW FAST [get_ports {DQ[2]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[2]}]
set_property PACKAGE_PIN AF15 [get_ports {DQ[2]}]

set_property SLEW FAST [get_ports {DQ[3]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[3]}]
set_property PACKAGE_PIN AD15 [get_ports {DQ[3]}]

set_property SLEW FAST [get_ports {DQ[4]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[4]}]
set_property PACKAGE_PIN AE15 [get_ports {DQ[4]}]

set_property SLEW FAST [get_ports {DQ[5]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[5]}]
set_property PACKAGE_PIN AF19 [get_ports {DQ[5]}]

set_property SLEW FAST [get_ports {DQ[6]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[6]}]
set_property PACKAGE_PIN AF20 [get_ports {DQ[6]}]

set_property SLEW FAST [get_ports {DQ[7]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[7]}]

```

```
set_property PACKAGE_PIN AD16 [get_ports {DQ[7]}]

set_property SLEW FAST [get_ports {DQ[8]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[8]}]
set_property PACKAGE_PIN AA15 [get_ports {DQ[8]}]

set_property SLEW FAST [get_ports {DQ[9]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[9]}]
set_property PACKAGE_PIN AC14 [get_ports {DQ[9]}]

set_property SLEW FAST [get_ports {DQ[10]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[10]}]
set_property PACKAGE_PIN AD14 [get_ports {DQ[10]}]

set_property SLEW FAST [get_ports {DQ[11]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[11]}]
set_property PACKAGE_PIN AB14 [get_ports {DQ[11]}]

set_property SLEW FAST [get_ports {DQ[12]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[12]}]
set_property PACKAGE_PIN AB15 [get_ports {DQ[12]}]

set_property SLEW FAST [get_ports {DQ[13]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[13]}]
set_property PACKAGE_PIN AA17 [get_ports {DQ[13]}]

set_property SLEW FAST [get_ports {DQ[14]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[14]}]
set_property PACKAGE_PIN AA18 [get_ports {DQ[14]}]

set_property SLEW FAST [get_ports {DQ[15]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[15]}]
set_property PACKAGE_PIN AB16 [get_ports {DQ[15]}]

set_property SLEW FAST [get_ports {DQ[16]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[16]}]
set_property PACKAGE_PIN U5 [get_ports {DQ[16]}]

set_property SLEW FAST [get_ports {DQ[17]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[17]}]
set_property PACKAGE_PIN U2 [get_ports {DQ[17]}]

set_property SLEW FAST [get_ports {DQ[18]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[18]}]
set_property PACKAGE_PIN U1 [get_ports {DQ[18]}]

set_property SLEW FAST [get_ports {DQ[19]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[19]}]
set_property PACKAGE_PIN V3 [get_ports {DQ[19]}]

set_property SLEW FAST [get_ports {DQ[20]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[20]}]
set_property PACKAGE_PIN W3 [get_ports {DQ[20]}]

set_property SLEW FAST [get_ports {DQ[21]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[21]}]
set_property PACKAGE_PIN U7 [get_ports {DQ[21]}]

set_property SLEW FAST [get_ports {DQ[22]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[22]}]
set_property PACKAGE_PIN V6 [get_ports {DQ[22]}]

set_property SLEW FAST [get_ports {DQ[23]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[23]}]
set_property PACKAGE_PIN V4 [get_ports {DQ[23]}]

set_property SLEW FAST [get_ports {DQ[24]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[24]}]
set_property PACKAGE_PIN Y2 [get_ports {DQ[24]}]

set_property SLEW FAST [get_ports {DQ[25]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[25]}]
set_property PACKAGE_PIN V2 [get_ports {DQ[25]}]
```

```
set_property SLEW FAST [get_ports {DQ[26]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[26]}]
set_property PACKAGE_PIN V1 [get_ports {DQ[26]}]

set_property SLEW FAST [get_ports {DQ[27]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[27]}]
set_property PACKAGE_PIN W1 [get_ports {DQ[27]}]

set_property SLEW FAST [get_ports {DQ[28]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[28]}]
set_property PACKAGE_PIN Y1 [get_ports {DQ[28]}]

set_property SLEW FAST [get_ports {DQ[29]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[29]}]
set_property PACKAGE_PIN AB2 [get_ports {DQ[29]}]

set_property SLEW FAST [get_ports {DQ[30]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[30]}]
set_property PACKAGE_PIN AC2 [get_ports {DQ[30]}]

set_property SLEW FAST [get_ports {DQ[31]}]
set_property IOSTANDARD SSTL135_T_DCI [get_ports {DQ[31]}]
set_property PACKAGE_PIN AA3 [get_ports {DQ[31]}]

# DDR3 Address
set_property SLEW FAST [get_ports {A[0]}]
set_property IOSTANDARD SSTL135 [get_ports {A[0]}]
set_property PACKAGE_PIN AC8 [get_ports {A[0]}]

set_property SLEW FAST [get_ports {A[1]}]
set_property IOSTANDARD SSTL135 [get_ports {A[1]}]
set_property PACKAGE_PIN AA7 [get_ports {A[1]}]

set_property SLEW FAST [get_ports {A[2]}]
set_property IOSTANDARD SSTL135 [get_ports {A[2]}]
set_property PACKAGE_PIN AA8 [get_ports {A[2]}]

set_property SLEW FAST [get_ports {A[3]}]
set_property IOSTANDARD SSTL135 [get_ports {A[3]}]
set_property PACKAGE_PIN AF7 [get_ports {A[3]}]

set_property SLEW FAST [get_ports {A[4]}]
set_property IOSTANDARD SSTL135 [get_ports {A[4]}]
set_property PACKAGE_PIN AE7 [get_ports {A[4]}]

set_property SLEW FAST [get_ports {A[5]}]
set_property IOSTANDARD SSTL135 [get_ports {A[5]}]
set_property PACKAGE_PIN W8 [get_ports {A[5]}]

set_property SLEW FAST [get_ports {A[6]}]
set_property IOSTANDARD SSTL135 [get_ports {A[6]}]
set_property PACKAGE_PIN V9 [get_ports {A[6]}]

set_property SLEW FAST [get_ports {A[7]}]
set_property IOSTANDARD SSTL135 [get_ports {A[7]}]
set_property PACKAGE_PIN Y10 [get_ports {A[7]}]

set_property SLEW FAST [get_ports {A[8]}]
set_property IOSTANDARD SSTL135 [get_ports {A[8]}]
set_property PACKAGE_PIN Y11 [get_ports {A[8]}]

set_property SLEW FAST [get_ports {A[9]}]
set_property IOSTANDARD SSTL135 [get_ports {A[9]}]
set_property PACKAGE_PIN Y7 [get_ports {A[9]}]

set_property SLEW FAST [get_ports {A[10]}]
set_property IOSTANDARD SSTL135 [get_ports {A[10]}]
set_property PACKAGE_PIN Y8 [get_ports {A[10]}]

set_property SLEW FAST [get_ports {A[11]}]
set_property IOSTANDARD SSTL135 [get_ports {A[11]}]
```

```
set_property PACKAGE_PIN V7 [get_ports {A[11]]}

set_property SLEW FAST [get_ports {A[12]]}
set_property IOSTANDARD SSTL135 [get_ports {A[12]]}
set_property PACKAGE_PIN V8 [get_ports {A[12]]}

set_property SLEW FAST [get_ports {A[13]]}
set_property IOSTANDARD SSTL135 [get_ports {A[13]]}
set_property PACKAGE_PIN W11 [get_ports {A[13]]}

set_property SLEW FAST [get_ports {A[14]]}
set_property IOSTANDARD SSTL135 [get_ports {A[14]]}
set_property PACKAGE_PIN V11 [get_ports {A[14]]}

# DDR3 Bank Address (BA)
set_property SLEW FAST [get_ports {BA[0]]}
set_property IOSTANDARD SSTL135 [get_ports {BA[0]]}
set_property PACKAGE_PIN AC7 [get_ports {BA[0]]}

set_property SLEW FAST [get_ports {BA[1]]}
set_property IOSTANDARD SSTL135 [get_ports {BA[1]]}
set_property PACKAGE_PIN AB7 [get_ports {BA[1]]}

set_property SLEW FAST [get_ports {BA[2]]}
set_property IOSTANDARD SSTL135 [get_ports {BA[2]]}
set_property PACKAGE_PIN AD8 [get_ports {BA[2]]}

# DDR3 Row Address Strobe (RAS)
set_property SLEW FAST [get_ports RAS_n]
set_property IOSTANDARD SSTL135 [get_ports RAS_n]
set_property PACKAGE_PIN AA9 [get_ports RAS_n]

# DDR3 Column Address Strobe (CAS)
set_property SLEW FAST [get_ports CAS_n]
set_property IOSTANDARD SSTL135 [get_ports CAS_n]
set_property PACKAGE_PIN AB9 [get_ports CAS_n]

# DDR3 Write Enable (WE_n, active-low)
set_property SLEW FAST [get_ports WE_n]
set_property IOSTANDARD SSTL135 [get_ports WE_n]
set_property PACKAGE_PIN AC9 [get_ports WE_n]

# DDR3 Reset (RST_n, active-low)
set_property SLEW FAST [get_ports RST_n]
set_property IOSTANDARD SSTL135 [get_ports RST_n]
set_property PACKAGE_PIN W10 [get_ports RST_n]

# DDR3 Clock Enable (CKE)
set_property SLEW FAST [get_ports {CKE[0]]}
set_property IOSTANDARD SSTL135 [get_ports {CKE[0]]}
set_property PACKAGE_PIN AB12 [get_ports {CKE[0]]}

# DDR3 On-Die Termination (ODT)
set_property SLEW FAST [get_ports {ODT[0]]}
set_property IOSTANDARD SSTL135 [get_ports {ODT[0]]}
set_property PACKAGE_PIN AC12 [get_ports {ODT[0]]}

# DDR3 Chip Select (CS, active-low)
set_property SLEW FAST [get_ports {CS_n[0]]}
set_property IOSTANDARD SSTL135 [get_ports {CS_n[0]]}
set_property PACKAGE_PIN AA13 [get_ports {CS_n[0]]}

# DDR3 Data Mask (DM)
set_property SLEW FAST [get_ports {DM[0]]}
```

```

set_property IOSTANDARD SSTL135 [get_ports {DM[0]}]
set_property PACKAGE_PIN AE17 [get_ports {DM[0]}]

set_property SLEW FAST [get_ports {DM[1]}]
set_property IOSTANDARD SSTL135 [get_ports {DM[1]}]
set_property PACKAGE_PIN AA14 [get_ports {DM[1]}]

set_property SLEW FAST [get_ports {DM[2]}]
set_property IOSTANDARD SSTL135 [get_ports {DM[2]}]
set_property PACKAGE_PIN U6 [get_ports {DM[2]}]

set_property SLEW FAST [get_ports {DM[3]}]
set_property IOSTANDARD SSTL135 [get_ports {DM[3]}]
set_property PACKAGE_PIN Y3 [get_ports {DM[3]}]

# DDR3 Data Strokes (DQS)
set_property SLEW FAST [get_ports {DQS_P[0]}]
set_property IOSTANDARD DIFF_SSTL135_T_DCI [get_ports {DQS_P[0]}]

set_property SLEW FAST [get_ports {DQS_N[0]}]
set_property IOSTANDARD DIFF_SSTL135_T_DCI [get_ports {DQS_N[0]}]
set_property PACKAGE_PIN AF18 [get_ports {DQS_N[0]}]
set_property PACKAGE_PIN AE18 [get_ports {DQS_P[0]}]

set_property SLEW FAST [get_ports {DQS_P[1]}]
set_property IOSTANDARD DIFF_SSTL135_T_DCI [get_ports {DQS_P[1]}]

set_property SLEW FAST [get_ports {DQS_N[1]}]
set_property IOSTANDARD DIFF_SSTL135_T_DCI [get_ports {DQS_N[1]}]
set_property PACKAGE_PIN Y16 [get_ports {DQS_N[1]}]
set_property PACKAGE_PIN Y15 [get_ports {DQS_P[1]}]

set_property SLEW FAST [get_ports {DQS_P[2]}]
set_property IOSTANDARD DIFF_SSTL135_T_DCI [get_ports {DQS_P[2]}]

set_property SLEW FAST [get_ports {DQS_N[2]}]
set_property IOSTANDARD DIFF_SSTL135_T_DCI [get_ports {DQS_N[2]}]
set_property PACKAGE_PIN W5 [get_ports {DQS_N[2]}]
set_property PACKAGE_PIN W6 [get_ports {DQS_P[2]}]

set_property SLEW FAST [get_ports {DQS_P[3]}]
set_property IOSTANDARD DIFF_SSTL135_T_DCI [get_ports {DQS_P[3]}]

set_property SLEW FAST [get_ports {DQS_N[3]}]
set_property IOSTANDARD DIFF_SSTL135_T_DCI [get_ports {DQS_N[3]}]
set_property PACKAGE_PIN AC1 [get_ports {DQS_N[3]}]
set_property PACKAGE_PIN AB1 [get_ports {DQS_P[3]}]

## #####
## Section: ADCs (LTC2323)
## #####

# ADC #0
set_property SLEW FAST [get_ports {ADC_CNV_n[0]}]
set_property IOSTANDARD LVCMOS25 [get_ports {ADC_CNV_n[0]}]
set_property PACKAGE_PIN AB26 [get_ports {ADC_CNV_n[0]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_P[0]}]
# External Termination

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_N[0]}]
# External Termination
set_property PACKAGE_PIN AC22 [get_ports {ADC_SCK_N[0]}]
set_property PACKAGE_PIN AB22 [get_ports {ADC_SCK_P[0]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_P[0]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_P[0]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_N[0]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_N[0]}]
set_property PACKAGE_PIN AA22 [get_ports {ADC_SCKOUT_N[0]}]
set_property PACKAGE_PIN Y22 [get_ports {ADC_SCKOUT_P[0]}]

```

```
set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_P[0]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_P[0]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_N[0]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_N[0]}]
set_property PACKAGE_PIN AC21 [get_ports {ADC_SDO1_N[0]}]
set_property PACKAGE_PIN AB21 [get_ports {ADC_SDO1_P[0]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_P[0]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_P[0]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_N[0]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_N[0]}]
set_property PACKAGE_PIN AD24 [get_ports {ADC_SDO2_N[0]}]
set_property PACKAGE_PIN AD23 [get_ports {ADC_SDO2_P[0]}]

# ADC #1
set_property SLEW FAST [get_ports {ADC_CNV_n[1]}]
set_property IOSTANDARD LVCMOS25 [get_ports {ADC_CNV_n[1]}]
set_property PACKAGE_PIN AC26 [get_ports {ADC_CNV_n[1]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_P[1]}]
# External Termination

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_N[1]}]
# External Termination
set_property PACKAGE_PIN AF25 [get_ports {ADC_SCK_N[1]}]
set_property PACKAGE_PIN AF24 [get_ports {ADC_SCK_P[1]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_P[1]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_P[1]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_N[1]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_N[1]}]
set_property PACKAGE_PIN AC24 [get_ports {ADC_SCKOUT_N[1]}]
set_property PACKAGE_PIN AC23 [get_ports {ADC_SCKOUT_P[1]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_P[1]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_P[1]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_N[1]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_N[1]}]
set_property PACKAGE_PIN AE26 [get_ports {ADC_SDO1_N[1]}]
set_property PACKAGE_PIN AD26 [get_ports {ADC_SDO1_P[1]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_P[1]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_P[1]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_N[1]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_N[1]}]
set_property PACKAGE_PIN AE25 [get_ports {ADC_SDO2_N[1]}]
set_property PACKAGE_PIN AD25 [get_ports {ADC_SDO2_P[1]}]

# ADC #2
set_property SLEW FAST [get_ports {ADC_CNV_n[2]}]
set_property IOSTANDARD LVCMOS25 [get_ports {ADC_CNV_n[2]}]
set_property PACKAGE_PIN W24 [get_ports {ADC_CNV_n[2]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_P[2]}]
# External Termination

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_N[2]}]
# External Termination
set_property PACKAGE_PIN V26 [get_ports {ADC_SCK_N[2]}]
set_property PACKAGE_PIN U26 [get_ports {ADC_SCK_P[2]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_P[2]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_P[2]}]
```

```
set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_N[2]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_N[2]}]
set_property PACKAGE_PIN AA24 [get_ports {ADC_SCKOUT_N[2]}]
set_property PACKAGE_PIN Y23 [get_ports {ADC_SCKOUT_P[2]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_P[2]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_P[2]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_N[2]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_N[2]}]
set_property PACKAGE_PIN V22 [get_ports {ADC_SDO1_N[2]}]
set_property PACKAGE_PIN U22 [get_ports {ADC_SDO1_P[2]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_P[2]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_P[2]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_N[2]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_N[2]}]
set_property PACKAGE_PIN U25 [get_ports {ADC_SDO2_N[2]}]
set_property PACKAGE_PIN U24 [get_ports {ADC_SDO2_P[2]}]

# ADC #3
set_property SLEW_FAST [get_ports {ADC_CNV_n[3]}]
set_property IOSTANDARD LVCMOS25 [get_ports {ADC_CNV_n[3]}]
set_property PACKAGE_PIN W23 [get_ports {ADC_CNV_n[3]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_P[3]}]
# External Termination

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_N[3]}]
# External Termination
set_property PACKAGE_PIN AB25 [get_ports {ADC_SCK_N[3]}]
set_property PACKAGE_PIN AA25 [get_ports {ADC_SCK_P[3]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_P[3]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_P[3]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_N[3]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_N[3]}]
set_property PACKAGE_PIN AB24 [get_ports {ADC_SCKOUT_N[3]}]
set_property PACKAGE_PIN AA23 [get_ports {ADC_SCKOUT_P[3]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_P[3]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_P[3]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_N[3]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_N[3]}]
set_property PACKAGE_PIN V24 [get_ports {ADC_SDO1_N[3]}]
set_property PACKAGE_PIN V23 [get_ports {ADC_SDO1_P[3]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_P[3]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_P[3]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_N[3]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_N[3]}]
set_property PACKAGE_PIN W21 [get_ports {ADC_SDO2_N[3]}]
set_property PACKAGE_PIN V21 [get_ports {ADC_SDO2_P[3]}]

# ADC #4
set_property SLEW_FAST [get_ports {ADC_CNV_n[4]}]
set_property IOSTANDARD LVCMOS25 [get_ports {ADC_CNV_n[4]}]
set_property PACKAGE_PIN N16 [get_ports {ADC_CNV_n[4]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_P[4]}]
# External Termination

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_N[4]}]
# External Termination
set_property PACKAGE_PIN P18 [get_ports {ADC_SCK_N[4]}]
set_property PACKAGE_PIN R18 [get_ports {ADC_SCK_P[4]}]
```



```
set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_P[4]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_P[4]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_N[4]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_N[4]}]
set_property PACKAGE_PIN R23 [get_ports {ADC_SCKOUT_N[4]}]
set_property PACKAGE_PIN R22 [get_ports {ADC_SCKOUT_P[4]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_P[4]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_P[4]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_N[4]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_N[4]}]
set_property PACKAGE_PIN T23 [get_ports {ADC_SDO1_N[4]}]
set_property PACKAGE_PIN T22 [get_ports {ADC_SDO1_P[4]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_P[4]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_P[4]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_N[4]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_N[4]}]
set_property PACKAGE_PIN P26 [get_ports {ADC_SDO2_N[4]}]
set_property PACKAGE_PIN R26 [get_ports {ADC_SDO2_P[4]}]

# ADC #5
set_property SLEW_FAST [get_ports {ADC_CNV_n[5]}]
set_property IOSTANDARD LVCMOS25 [get_ports {ADC_CNV_n[5]}]
set_property PACKAGE_PIN U16 [get_ports {ADC_CNV_n[5]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_P[5]}]
# External Termination

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_N[5]}]
# External Termination
set_property PACKAGE_PIN T17 [get_ports {ADC_SCK_N[5]}]
set_property PACKAGE_PIN U17 [get_ports {ADC_SCK_P[5]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_P[5]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_P[5]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_N[5]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_N[5]}]
set_property PACKAGE_PIN P21 [get_ports {ADC_SCKOUT_N[5]}]
set_property PACKAGE_PIN R21 [get_ports {ADC_SCKOUT_P[5]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_P[5]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_P[5]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_N[5]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_N[5]}]
set_property PACKAGE_PIN R20 [get_ports {ADC_SDO1_N[5]}]
set_property PACKAGE_PIN T20 [get_ports {ADC_SDO1_P[5]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_P[5]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_P[5]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_N[5]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_N[5]}]
set_property PACKAGE_PIN U20 [get_ports {ADC_SDO2_N[5]}]
set_property PACKAGE_PIN U19 [get_ports {ADC_SDO2_P[5]}]

# ADC #6
set_property SLEW_FAST [get_ports {ADC_CNV_n[6]}]
set_property IOSTANDARD LVCMOS25 [get_ports {ADC_CNV_n[6]}]
set_property PACKAGE_PIN M21 [get_ports {ADC_CNV_n[6]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_P[6]}]
# External Termination
```

```

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_N[6]}]
# External Termination
set_property PACKAGE_PIN K26 [get_ports {ADC_SCK_N[6]}]
set_property PACKAGE_PIN K25 [get_ports {ADC_SCK_P[6]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_P[6]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_P[6]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_N[6]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_N[6]}]
set_property PACKAGE_PIN N22 [get_ports {ADC_SCKOUT_N[6]}]
set_property PACKAGE_PIN N21 [get_ports {ADC_SCKOUT_P[6]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_P[6]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_P[6]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_N[6]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_N[6]}]
set_property PACKAGE_PIN N24 [get_ports {ADC_SDO1_N[6]}]
set_property PACKAGE_PIN P24 [get_ports {ADC_SDO1_P[6]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_P[6]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_P[6]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_N[6]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_N[6]}]
set_property PACKAGE_PIN L25 [get_ports {ADC_SDO2_N[6]}]
set_property PACKAGE_PIN M25 [get_ports {ADC_SDO2_P[6]}]

# ADC #7
set_property SLEW_FAST [get_ports {ADC_CNV_n[7]}]
set_property IOSTANDARD LVCMOS25 [get_ports {ADC_CNV_n[7]}]
set_property PACKAGE_PIN M22 [get_ports {ADC_CNV_n[7]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_P[7]}]
# External Termination

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCK_N[7]}]
# External Termination
set_property PACKAGE_PIN T25 [get_ports {ADC_SCK_N[7]}]
set_property PACKAGE_PIN T24 [get_ports {ADC_SCK_P[7]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_P[7]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_P[7]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SCKOUT_N[7]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SCKOUT_N[7]}]
set_property PACKAGE_PIN N23 [get_ports {ADC_SCKOUT_N[7]}]
set_property PACKAGE_PIN P23 [get_ports {ADC_SCKOUT_P[7]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_P[7]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_P[7]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO1_N[7]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO1_N[7]}]
set_property PACKAGE_PIN M26 [get_ports {ADC_SDO1_N[7]}]
set_property PACKAGE_PIN N26 [get_ports {ADC_SDO1_P[7]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_P[7]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_P[7]}]

set_property IOSTANDARD LVDS_25 [get_ports {ADC_SDO2_N[7]}]
set_property DIFF_TERM TRUE [get_ports {ADC_SDO2_N[7]}]
set_property PACKAGE_PIN P25 [get_ports {ADC_SDO2_N[7]}]
set_property PACKAGE_PIN R25 [get_ports {ADC_SDO2_P[7]}]

## #####
## Section: DACs (AD5547)
## #####

```

```
## Data BUS
set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[0]}]
set_property SLEW SLOW [get_ports {DAC_D[0]}]
set_property DRIVE 8 [get_ports {DAC_D[0]}]
set_property PACKAGE_PIN D20 [get_ports {DAC_D[0]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[1]}]
set_property SLEW SLOW [get_ports {DAC_D[1]}]
set_property DRIVE 8 [get_ports {DAC_D[1]}]
set_property PACKAGE_PIN E20 [get_ports {DAC_D[1]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[2]}]
set_property SLEW SLOW [get_ports {DAC_D[2]}]
set_property DRIVE 8 [get_ports {DAC_D[2]}]
set_property PACKAGE_PIN H17 [get_ports {DAC_D[2]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[3]}]
set_property SLEW SLOW [get_ports {DAC_D[3]}]
set_property DRIVE 8 [get_ports {DAC_D[3]}]
set_property PACKAGE_PIN D19 [get_ports {DAC_D[3]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[4]}]
set_property SLEW SLOW [get_ports {DAC_D[4]}]
set_property DRIVE 8 [get_ports {DAC_D[4]}]
set_property PACKAGE_PIN G16 [get_ports {DAC_D[4]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[5]}]
set_property SLEW SLOW [get_ports {DAC_D[5]}]
set_property DRIVE 8 [get_ports {DAC_D[5]}]
set_property PACKAGE_PIN F19 [get_ports {DAC_D[5]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[6]}]
set_property SLEW SLOW [get_ports {DAC_D[6]}]
set_property DRIVE 8 [get_ports {DAC_D[6]}]
set_property PACKAGE_PIN F20 [get_ports {DAC_D[6]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[7]}]
set_property SLEW SLOW [get_ports {DAC_D[7]}]
set_property DRIVE 8 [get_ports {DAC_D[7]}]
set_property PACKAGE_PIN E17 [get_ports {DAC_D[7]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[8]}]
set_property SLEW SLOW [get_ports {DAC_D[8]}]
set_property DRIVE 8 [get_ports {DAC_D[8]}]
set_property PACKAGE_PIN E18 [get_ports {DAC_D[8]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[9]}]
set_property SLEW SLOW [get_ports {DAC_D[9]}]
set_property DRIVE 8 [get_ports {DAC_D[9]}]
set_property PACKAGE_PIN D15 [get_ports {DAC_D[9]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[10]}]
set_property SLEW SLOW [get_ports {DAC_D[10]}]
set_property DRIVE 8 [get_ports {DAC_D[10]}]
set_property PACKAGE_PIN D16 [get_ports {DAC_D[10]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[11]}]
set_property SLEW SLOW [get_ports {DAC_D[11]}]
set_property DRIVE 8 [get_ports {DAC_D[11]}]
set_property PACKAGE_PIN C16 [get_ports {DAC_D[11]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[12]}]
set_property SLEW SLOW [get_ports {DAC_D[12]}]
set_property DRIVE 8 [get_ports {DAC_D[12]}]
set_property PACKAGE_PIN C18 [get_ports {DAC_D[12]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[13]}]
set_property SLEW SLOW [get_ports {DAC_D[13]}]
set_property DRIVE 8 [get_ports {DAC_D[13]}]
set_property PACKAGE_PIN B16 [get_ports {DAC_D[13]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[14]}]
```

```
set_property SLEW SLOW [get_ports {DAC_D[14]]}
set_property DRIVE 8 [get_ports {DAC_D[14]]}
set_property PACKAGE_PIN B19 [get_ports {DAC_D[14]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[15]]}
set_property SLEW SLOW [get_ports {DAC_D[15]]}
set_property DRIVE 8 [get_ports {DAC_D[15]]}
set_property PACKAGE_PIN A17 [get_ports {DAC_D[15]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[16]]}
set_property SLEW SLOW [get_ports {DAC_D[16]]}
set_property DRIVE 8 [get_ports {DAC_D[16]]}
set_property PACKAGE_PIN G15 [get_ports {DAC_D[16]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[17]]}
set_property SLEW SLOW [get_ports {DAC_D[17]]}
set_property DRIVE 8 [get_ports {DAC_D[17]]}
set_property PACKAGE_PIN H18 [get_ports {DAC_D[17]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[18]]}
set_property SLEW SLOW [get_ports {DAC_D[18]]}
set_property DRIVE 8 [get_ports {DAC_D[18]]}
set_property PACKAGE_PIN K15 [get_ports {DAC_D[18]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[19]]}
set_property SLEW SLOW [get_ports {DAC_D[19]]}
set_property DRIVE 8 [get_ports {DAC_D[19]]}
set_property PACKAGE_PIN K20 [get_ports {DAC_D[19]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[20]]}
set_property SLEW SLOW [get_ports {DAC_D[20]]}
set_property DRIVE 8 [get_ports {DAC_D[20]]}
set_property PACKAGE_PIN J18 [get_ports {DAC_D[20]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[21]]}
set_property SLEW SLOW [get_ports {DAC_D[21]]}
set_property DRIVE 8 [get_ports {DAC_D[21]]}
set_property PACKAGE_PIN J19 [get_ports {DAC_D[21]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[22]]}
set_property SLEW SLOW [get_ports {DAC_D[22]]}
set_property DRIVE 8 [get_ports {DAC_D[22]]}
set_property PACKAGE_PIN J20 [get_ports {DAC_D[22]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[23]]}
set_property SLEW SLOW [get_ports {DAC_D[23]]}
set_property DRIVE 8 [get_ports {DAC_D[23]]}
set_property PACKAGE_PIN A19 [get_ports {DAC_D[23]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[24]]}
set_property SLEW SLOW [get_ports {DAC_D[24]]}
set_property DRIVE 8 [get_ports {DAC_D[24]]}
set_property PACKAGE_PIN B17 [get_ports {DAC_D[24]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[25]]}
set_property SLEW SLOW [get_ports {DAC_D[25]]}
set_property DRIVE 8 [get_ports {DAC_D[25]]}
set_property PACKAGE_PIN C17 [get_ports {DAC_D[25]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[26]]}
set_property SLEW SLOW [get_ports {DAC_D[26]]}
set_property DRIVE 8 [get_ports {DAC_D[26]]}
set_property PACKAGE_PIN L20 [get_ports {DAC_D[26]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[27]]}
set_property SLEW SLOW [get_ports {DAC_D[27]]}
set_property DRIVE 8 [get_ports {DAC_D[27]]}
set_property PACKAGE_PIN H16 [get_ports {DAC_D[27]]}

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[28]]}
set_property SLEW SLOW [get_ports {DAC_D[28]]}
set_property DRIVE 8 [get_ports {DAC_D[28]]}
set_property PACKAGE_PIN G20 [get_ports {DAC_D[28]]}
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[29]}]
set_property SLEW SLOW [get_ports {DAC_D[29]}]
set_property DRIVE 8 [get_ports {DAC_D[29]}]
set_property PACKAGE_PIN A18 [get_ports {DAC_D[29]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[30]}]
set_property SLEW SLOW [get_ports {DAC_D[30]}]
set_property DRIVE 8 [get_ports {DAC_D[30]}]
set_property PACKAGE_PIN G17 [get_ports {DAC_D[30]}]

set_property IOSTANDARD LVCMOS33 [get_ports {DAC_D[31]}]
set_property SLEW SLOW [get_ports {DAC_D[31]}]
set_property DRIVE 8 [get_ports {DAC_D[31]}]
set_property PACKAGE_PIN D18 [get_ports {DAC_D[31]}]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_ADR0]
set_property SLEW SLOW [get_ports DAC_ADR0]
set_property DRIVE 8 [get_ports DAC_ADR0]
set_property PACKAGE_PIN E16 [get_ports DAC_ADR0]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_ADR1]
set_property SLEW SLOW [get_ports DAC_ADR1]
set_property DRIVE 8 [get_ports DAC_ADR1]
set_property PACKAGE_PIN G19 [get_ports DAC_ADR1]

set_property IOSTANDARD LVCMOS25 [get_ports DAC_MSB]
set_property SLEW SLOW [get_ports DAC_MSB]
set_property DRIVE 8 [get_ports DAC_MSB]
set_property PACKAGE_PIN W26 [get_ports DAC_MSB]

set_property IOSTANDARD LVCMOS25 [get_ports DAC_RS_N]
set_property SLEW SLOW [get_ports DAC_RS_N]
set_property DRIVE 8 [get_ports DAC_RS_N]
set_property PACKAGE_PIN W25 [get_ports DAC_RS_N]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_WR00_01_N]
set_property SLEW SLOW [get_ports DAC_WR00_01_N]
set_property DRIVE 8 [get_ports DAC_WR00_01_N]
set_property PACKAGE_PIN F15 [get_ports DAC_WR00_01_N]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_WR02_03_N]
set_property SLEW SLOW [get_ports DAC_WR02_03_N]
set_property DRIVE 8 [get_ports DAC_WR02_03_N]
set_property PACKAGE_PIN C19 [get_ports DAC_WR02_03_N]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_WR04_05_N]
set_property SLEW SLOW [get_ports DAC_WR04_05_N]
set_property DRIVE 8 [get_ports DAC_WR04_05_N]
set_property PACKAGE_PIN J16 [get_ports DAC_WR04_05_N]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_WR06_07_N]
set_property SLEW SLOW [get_ports DAC_WR06_07_N]
set_property DRIVE 8 [get_ports DAC_WR06_07_N]
set_property PACKAGE_PIN E15 [get_ports DAC_WR06_07_N]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_WR08_09_N]
set_property SLEW SLOW [get_ports DAC_WR08_09_N]
set_property DRIVE 8 [get_ports DAC_WR08_09_N]
set_property PACKAGE_PIN L19 [get_ports DAC_WR08_09_N]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_WR10_11_N]
set_property SLEW SLOW [get_ports DAC_WR10_11_N]
set_property DRIVE 8 [get_ports DAC_WR10_11_N]
set_property PACKAGE_PIN H19 [get_ports DAC_WR10_11_N]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_WR12_13_N]
set_property SLEW SLOW [get_ports DAC_WR12_13_N]
set_property DRIVE 8 [get_ports DAC_WR12_13_N]
set_property PACKAGE_PIN K16 [get_ports DAC_WR12_13_N]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_WR14_15_N]
set_property SLEW SLOW [get_ports DAC_WR14_15_N]
```

```

set_property DRIVE 8 [get_ports DAC_WR14_15_N]
set_property PACKAGE_PIN K17 [get_ports DAC_WR14_15_N]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_LDAC00_01]
set_property SLEW SLOW [get_ports DAC_LDAC00_01]
set_property DRIVE 8 [get_ports DAC_LDAC00_01]
set_property PACKAGE_PIN H26 [get_ports DAC_LDAC00_01]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_LDAC02_03]
set_property SLEW SLOW [get_ports DAC_LDAC02_03]
set_property DRIVE 8 [get_ports DAC_LDAC02_03]
set_property PACKAGE_PIN H21 [get_ports DAC_LDAC02_03]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_LDAC04_05]
set_property SLEW SLOW [get_ports DAC_LDAC04_05]
set_property DRIVE 8 [get_ports DAC_LDAC04_05]
set_property PACKAGE_PIN G21 [get_ports DAC_LDAC04_05]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_LDAC06_07]
set_property SLEW SLOW [get_ports DAC_LDAC06_07]
set_property DRIVE 8 [get_ports DAC_LDAC06_07]
set_property PACKAGE_PIN H23 [get_ports DAC_LDAC06_07]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_LDAC08_09]
set_property SLEW SLOW [get_ports DAC_LDAC08_09]
set_property DRIVE 8 [get_ports DAC_LDAC08_09]
set_property PACKAGE_PIN H24 [get_ports DAC_LDAC08_09]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_LDAC10_11]
set_property SLEW SLOW [get_ports DAC_LDAC10_11]
set_property DRIVE 8 [get_ports DAC_LDAC10_11]
set_property PACKAGE_PIN J21 [get_ports DAC_LDAC10_11]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_LDAC12_13]
set_property SLEW SLOW [get_ports DAC_LDAC12_13]
set_property DRIVE 8 [get_ports DAC_LDAC12_13]
set_property PACKAGE_PIN H22 [get_ports DAC_LDAC12_13]

set_property IOSTANDARD LVCMOS33 [get_ports DAC_LDAC14_15]
set_property SLEW SLOW [get_ports DAC_LDAC14_15]
set_property DRIVE 8 [get_ports DAC_LDAC14_15]
set_property PACKAGE_PIN J24 [get_ports DAC_LDAC14_15]

## IOB
set_property IOB TRUE [get_ports DAC_WR*]
set_property IOB TRUE [get_ports DAC_D*]
set_property IOB TRUE [get_ports DAC_ADR*]
set_property IOB TRUE [get_ports DAC_WR*]
set_property IOB TRUE [get_ports DAC_LDAC*]

## #####
## Section: Digital I/O to Firefly
## #####

set_property IOSTANDARD LVDS_25 [get_ports DIG_IO_00_P]
set_property PACKAGE_PIN M24 [get_ports DIG_IO_00_P]

set_property IOSTANDARD LVDS_25 [get_ports DIG_IO_00_N]
set_property PACKAGE_PIN L24 [get_ports DIG_IO_00_N]

set_property IOSTANDARD LVDS_25 [get_ports DIG_IO_01_P]
set_property PACKAGE_PIN E10 [get_ports DIG_IO_01_P]

set_property IOSTANDARD LVDS_25 [get_ports DIG_IO_01_N]
set_property PACKAGE_PIN D10 [get_ports DIG_IO_01_N]

set_property IOSTANDARD LVDS_25 [get_ports DIG_IO_02_P]
set_property PACKAGE_PIN N19 [get_ports DIG_IO_02_P]

set_property IOSTANDARD LVDS_25 [get_ports DIG_IO_02_N]
set_property PACKAGE_PIN M20 [get_ports DIG_IO_02_N]

set_property IOSTANDARD LVDS_25 [get_ports DIG_IO_03_P]

```

```
set_property PACKAGE_PIN Y25 [get_ports DIG_IO_03_P]

set_property IOSTANDARD LVDS_25 [get_ports DIG_IO_03_N]
set_property PACKAGE_PIN Y26 [get_ports DIG_IO_03_N]

## #####
## Section: BACK IO
## #####

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO0_P]
set_property PACKAGE_PIN AE23 [get_ports BACK_IO0_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO0_N]
set_property PACKAGE_PIN AF23 [get_ports BACK_IO0_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO1_P]
set_property PACKAGE_PIN AE22 [get_ports BACK_IO1_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO1_N]
set_property PACKAGE_PIN AF22 [get_ports BACK_IO1_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO2_P]
set_property PACKAGE_PIN AD21 [get_ports BACK_IO2_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO2_N]
set_property PACKAGE_PIN AE21 [get_ports BACK_IO2_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO3_P]
set_property PACKAGE_PIN W20 [get_ports BACK_IO3_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO3_N]
set_property PACKAGE_PIN Y21 [get_ports BACK_IO3_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO4_P]
set_property PACKAGE_PIN T18 [get_ports BACK_IO4_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO4_N]
set_property PACKAGE_PIN T19 [get_ports BACK_IO4_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO5_P]
set_property PACKAGE_PIN R16 [get_ports BACK_IO5_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO5_N]
set_property PACKAGE_PIN R17 [get_ports BACK_IO5_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO6_P]
set_property PACKAGE_PIN N18 [get_ports BACK_IO6_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO6_N]
set_property PACKAGE_PIN M19 [get_ports BACK_IO6_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO7_P]
set_property PACKAGE_PIN P16 [get_ports BACK_IO7_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO7_N]
set_property PACKAGE_PIN N17 [get_ports BACK_IO7_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO8_P]
set_property PACKAGE_PIN J13 [get_ports BACK_IO8_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO8_N]
set_property PACKAGE_PIN H13 [get_ports BACK_IO8_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO9_P]
set_property PACKAGE_PIN H14 [get_ports BACK_IO9_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO9_N]
set_property PACKAGE_PIN G14 [get_ports BACK_IO9_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO10_P]
set_property PACKAGE_PIN J11 [get_ports BACK_IO10_P]
```

```
set_property IOSTANDARD LVDS_25 [get_ports BACK_IO10_N]
set_property PACKAGE_PIN J10 [get_ports BACK_IO10_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO11_P]
set_property PACKAGE_PIN H12 [get_ports BACK_IO11_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO11_N]
set_property PACKAGE_PIN H11 [get_ports BACK_IO11_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO12_P]
set_property PACKAGE_PIN G12 [get_ports BACK_IO12_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO12_N]
set_property PACKAGE_PIN F12 [get_ports BACK_IO12_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO13_P]
set_property PACKAGE_PIN H9 [get_ports BACK_IO13_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO13_N]
set_property PACKAGE_PIN H8 [get_ports BACK_IO13_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO14_P]
set_property PACKAGE_PIN F14 [get_ports BACK_IO14_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO14_N]
set_property PACKAGE_PIN F13 [get_ports BACK_IO14_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO15_P]
set_property PACKAGE_PIN G10 [get_ports BACK_IO15_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO15_N]
set_property PACKAGE_PIN G9 [get_ports BACK_IO15_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO16_P]
set_property PACKAGE_PIN E13 [get_ports BACK_IO16_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO16_N]
set_property PACKAGE_PIN E12 [get_ports BACK_IO16_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO17_P]
set_property PACKAGE_PIN F9 [get_ports BACK_IO17_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO17_N]
set_property PACKAGE_PIN F8 [get_ports BACK_IO17_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO18_P]
set_property PACKAGE_PIN P19 [get_ports BACK_IO18_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO18_N]
set_property PACKAGE_PIN P20 [get_ports BACK_IO18_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO19_P]
set_property PACKAGE_PIN C14 [get_ports BACK_IO19_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO19_N]
set_property PACKAGE_PIN C13 [get_ports BACK_IO19_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO20_P]
set_property PACKAGE_PIN D9 [get_ports BACK_IO20_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO20_N]
set_property PACKAGE_PIN D8 [get_ports BACK_IO20_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO21_P]
set_property PACKAGE_PIN B15 [get_ports BACK_IO21_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO21_N]
set_property PACKAGE_PIN A15 [get_ports BACK_IO21_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO22_P]
set_property PACKAGE_PIN B12 [get_ports BACK_IO22_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO22_N]
```

```
set_property PACKAGE_PIN B11 [get_ports BACK_IO22_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO23_P]
set_property PACKAGE_PIN B14 [get_ports BACK_IO23_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO23_N]
set_property PACKAGE_PIN A14 [get_ports BACK_IO23_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO24_P]
set_property PACKAGE_PIN C9 [get_ports BACK_IO24_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO24_N]
set_property PACKAGE_PIN B9 [get_ports BACK_IO24_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO25_P]
set_property PACKAGE_PIN A13 [get_ports BACK_IO25_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO25_N]
set_property PACKAGE_PIN A12 [get_ports BACK_IO25_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO26_P]
set_property PACKAGE_PIN B10 [get_ports BACK_IO26_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO26_N]
set_property PACKAGE_PIN A10 [get_ports BACK_IO26_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO27_P]
set_property PACKAGE_PIN A9 [get_ports BACK_IO27_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO27_N]
set_property PACKAGE_PIN A8 [get_ports BACK_IO27_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO28_P]
set_property PACKAGE_PIN G11 [get_ports BACK_IO28_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO28_N]
set_property PACKAGE_PIN F10 [get_ports BACK_IO28_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO29_P]
set_property PACKAGE_PIN D14 [get_ports BACK_IO29_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO29_N]
set_property PACKAGE_PIN D13 [get_ports BACK_IO29_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO30_P]
set_property PACKAGE_PIN E11 [get_ports BACK_IO30_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO30_N]
set_property PACKAGE_PIN D11 [get_ports BACK_IO30_N]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO31_P]
set_property PACKAGE_PIN C12 [get_ports BACK_IO31_P]

set_property IOSTANDARD LVDS_25 [get_ports BACK_IO31_N]
set_property PACKAGE_PIN C11 [get_ports BACK_IO31_N]

## #####
## Section: Miscellaneous
## #####

# MCB_CLK
set_property IOSTANDARD DIFF_SSTL135_DCI [get_ports MCB_CLK_P]
set_property PACKAGE_PIN AB11 [get_ports MCB_CLK_P]
set_property IOSTANDARD DIFF_SSTL135_DCI [get_ports MCB_CLK_N]
set_property PACKAGE_PIN AC11 [get_ports MCB_CLK_N]

# REF_CLK
set_property IOSTANDARD DIFF_SSTL135_DCI [get_ports REF_CLK_P]
set_property PACKAGE_PIN AA10 [get_ports REF_CLK_P]
set_property IOSTANDARD DIFF_SSTL135_DCI [get_ports REF_CLK_N]
set_property PACKAGE_PIN AB10 [get_ports REF_CLK_N]

# USER_CLKA
```

```
set_property IOSTANDARD LVCMOS33 [get_ports USER_CLKA]
set_property PACKAGE_PIN F22 [get_ports USER_CLKA]

# SI514_CLK
set_property IOSTANDARD LVDS_25 [get_ports SI514_CLK_P]
set_property PACKAGE_PIN G22 [get_ports SI514_CLK_P]
set_property IOSTANDARD LVDS_25 [get_ports SI514_CLK_N]
set_property PACKAGE_PIN F23 [get_ports SI514_CLK_N]

# K7_EMCCLK
set_property IOSTANDARD LVCMOS33 [get_ports K7_EMCCLK]
set_property PACKAGE_PIN B26 [get_ports K7_EMCCLK]

# User: I2C Interface (inter-device communication)
set_property SLEW SLOW [get_ports USER_SDA]
set_property DRIVE 4 [get_ports USER_SDA]
set_property IOSTANDARD LVCMOS33 [get_ports USER_SDA]
set_property PACKAGE_PIN G26 [get_ports USER_SDA]

set_property SLEW SLOW [get_ports USER_SCL]
set_property DRIVE 4 [get_ports USER_SCL]
set_property IOSTANDARD LVCMOS33 [get_ports USER_SCL]
set_property PACKAGE_PIN F25 [get_ports USER_SCL]

# BCC: I2C Interface (BCC communication)
set_property SLEW SLOW [get_ports FPGA_SDA]
set_property DRIVE 4 [get_ports FPGA_SDA]
set_property IOSTANDARD LVCMOS33 [get_ports FPGA_SDA]
set_property PACKAGE_PIN M17 [get_ports FPGA_SDA]

set_property SLEW SLOW [get_ports FPGA_SCL]
set_property DRIVE 4 [get_ports FPGA_SCL]
set_property IOSTANDARD LVCMOS33 [get_ports FPGA_SCL]
set_property PACKAGE_PIN L18 [get_ports FPGA_SCL]

# LEDs
set_property SLEW FAST [get_ports {USER_LED[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {USER_LED[0]}]
set_property PACKAGE_PIN J26 [get_ports {USER_LED[0]}]

set_property SLEW FAST [get_ports {USER_LED[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {USER_LED[1]}]
set_property PACKAGE_PIN E26 [get_ports {USER_LED[1]}]
```