Blue Sky Electronics

TDIG Rev F Engineering and User's Manual

Version 4.0a (2010.01.13)

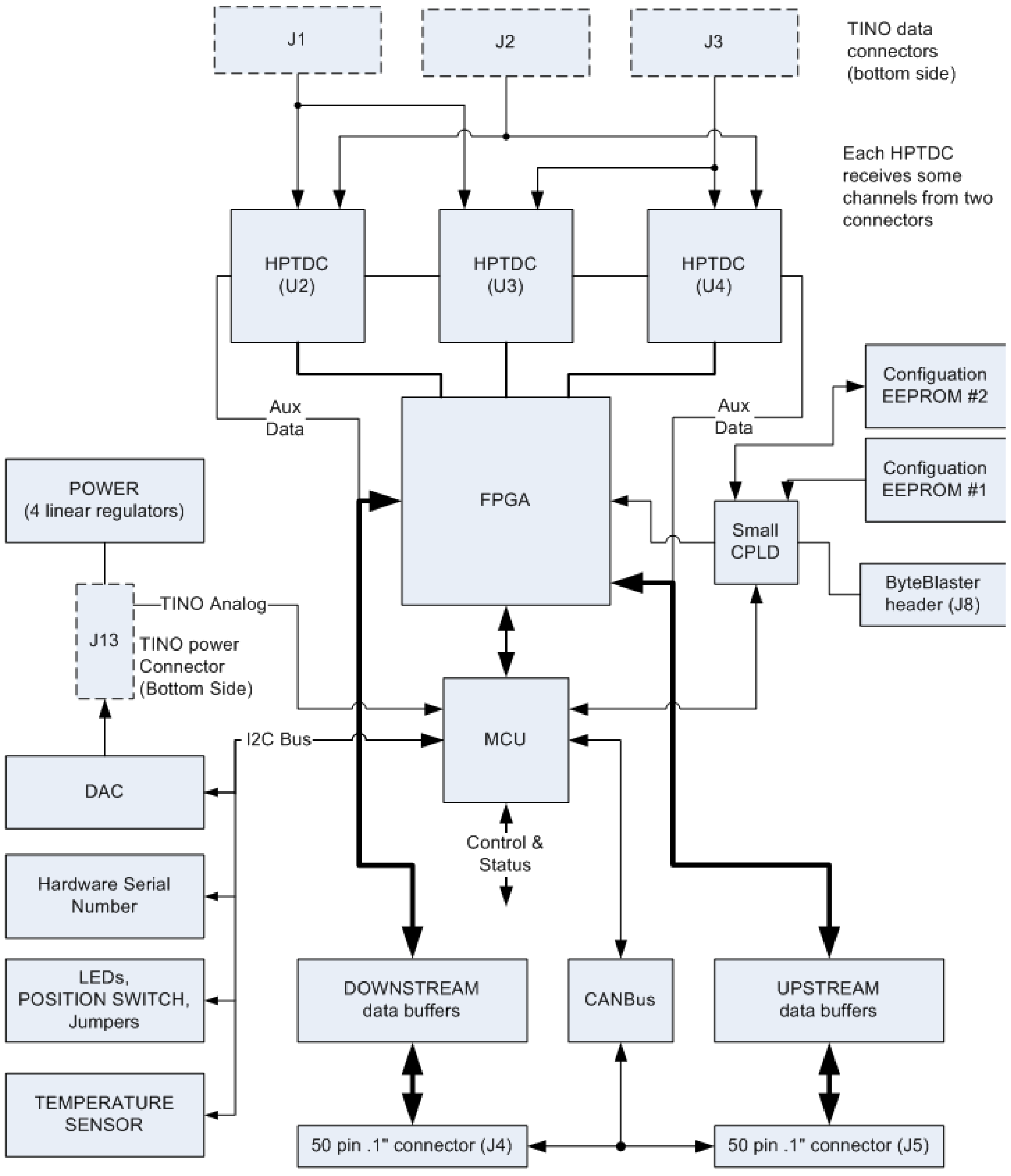
Lloyd Bridges
William Burton
Jo Schambach

TDIG Rev F Engineering and User Manual

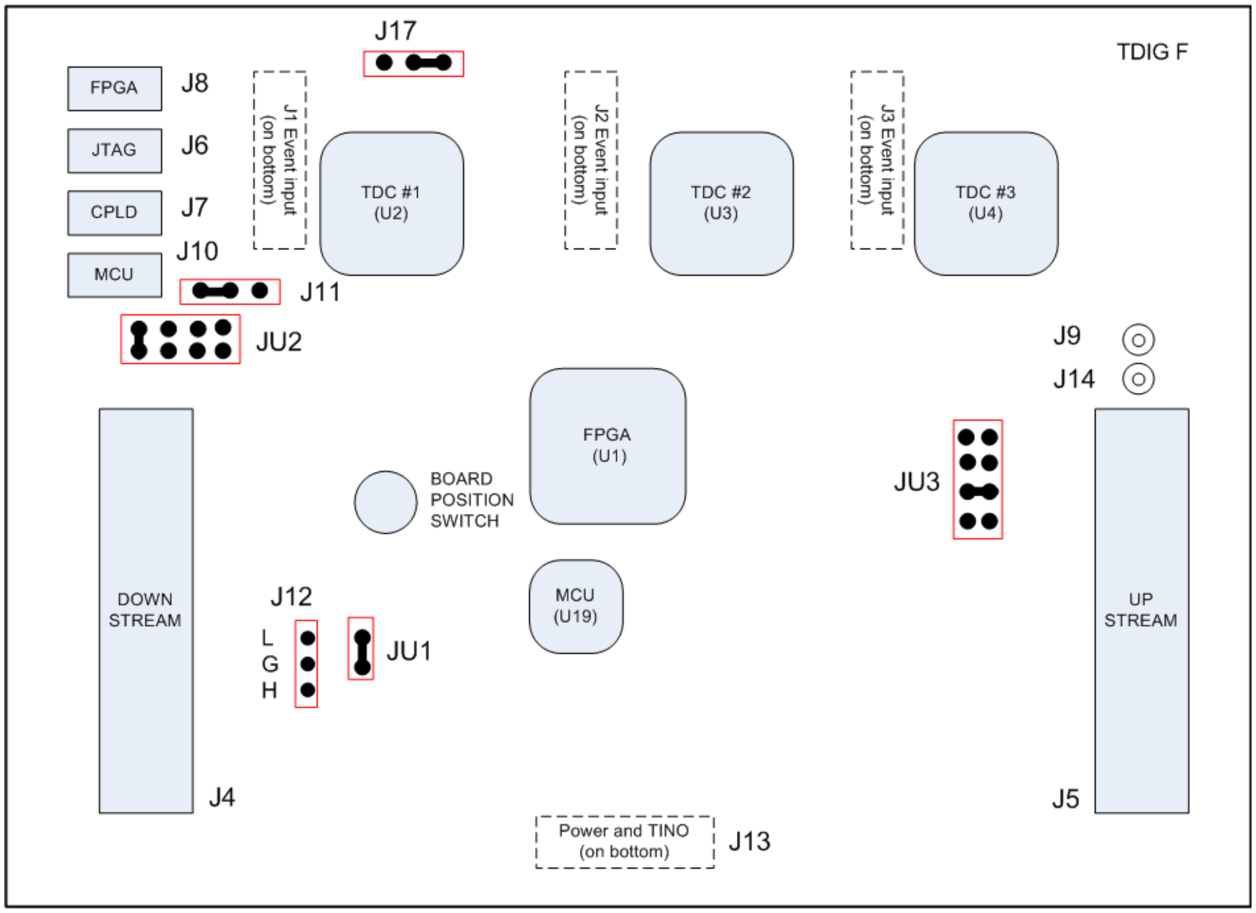
Version 4_x: September, 2009

Table of Contents

1. Block diagram



2. Board Top view



3. Connectors

3.1. Power and TINO I/O Connector (J13):

Samtec ASP-136468-01, custom version of Samtec IPS1-110-01-S-D. Located on bottom (secondary) side of PCB.
Mates to corresponding IPT-110- on TINO board.
This connector carries supply power and ground from TINO to TDIG plus additional inter-board signals: Analog output from DAC used to set TINO event threshold; Analog input voltages normally intended for TINO temperature monitoring; Digital output pulse intended for TINO test.
Input power to the board is limited by a self-resetting PTC fuse 3 Amperes (Littelfuse # SMD2920P300T)

3.2. Event Measurement Input Connectors (J1, J2, and J3):

Samtec QTE-020-03-F-D-A-K. Located on bottom (secondary) side of PCB.
Mates to corresponding QSE-040 on TINO board.
Each connector carries 8 differential pairs (LVDS level) event input signals and signal grounds. Routing of signals from connectors to HPTDC measurement chips is described in Appendix B: “TINO_N” – “TDIG-E” Mapping.

3.3. Upstream/Downstream Connectors (J5, J4)

Right angle 50-position 0.1” ribbon header with latches (3M #3433-5203).
Signal levels: LVDS pairs except Clock PECL.

3.3.1. Upstream / Downstream signals over ribbon cable

Signals discussed in this section are labeled as

- OUT (from TDIG to upstream) or
- IN (from upstream to TDIG)

Due to placement of the upstream ribbon cable connector on the top side of the circuit card, the ribbon cables invert the polarity of their differential signals.

Re-inversions take place as discussed below:

3.3.1.1. Re-Inversions at UPSTREAM only

The re-inversions take place on the UPSTREAM side of TDIG only.

- UCLK_40MHZ (IN)
- CAN (BIDIRECTIONAL)

3.3.1.2. Re-Inversions at Upstream Receiver

Some signals are re-inverted by swapping pin polarities at the upstream receiver device (U22):

- TRIGGER (IN)

- CLK_10MHZ (IN)
- BUNCH_RST (IN)

3.3.1.3. Re-Inverted by FPGA

The signals below are re-inverted at the UPSTREAM input/output of the TDIG FPGA. No inversion takes place on the DOWNSTREAM side.

- UDAISY_DATA (OUT)
- UDAISY_TOK_OUT (OUT)
- USTATUS0 (OUT)
- USTATUS1 (OUT)
- UDAISY_CLK (OUT)
- UDAISY_TOK_IN (IN)
- MCU_RESET?? (IN)
- FLEX_RESET_IN (IN)
- USPARE_IN1 (IN)
- UCONFIG_IN (IN)
- UMULT[3:0} (OUT)

3.3.2. Upstream Connector Signals (J5)

Cable Signal	+ pin	- pin	FPGA name (Pin)	Non-FPGA Name	Function	Logic	# of cable wires	Dir'n	Grp	Notes
GND	2	1					2			
UCLK_40MHZ	_N 4	_P 3	PLD_CLKIN1, PLD_CLKIN2	UCLK_40MHZ_P UCLK_40MHZ_N	External clock, note reversal	LVPECL	2	in	0	Buffered between US/DS connectors U45
GND	6	5					2			
ULV0	8	7	n/c USPARE_OUT0 (T22)	H3_SER_OUT		LVDS	2	out	7	U6-A1
ULV1	10	9	n/c USPARE_OUT1 (U18)	H3_TOKEN_OUT		LVDS	2	out	7	U6-A2
ULV2	12	11		H3_STROBE_OUT		LVDS	2	out	7	U6-A3
CAN_	_N 14	_P 13		CAN_P, CAN_N	CANBus, note reversal	LVDS	2	bidir	1	U23
ULV3	16	15	UDAISY_DATA (R21)		TDC daisy	LVDS	2	out	3	U6-B1; One daisy chain for 4 boards
ULV4	18	17	UDAISY_TOK_OUT (N21)		TDC daisy	LVDS	2	out	3	U6-B2
ULV5	20	19	USTATUS0 (N22)			LVDS	2	out	3	U6-B3; USTROBE_OUT in FPGA file
ULV6	22	21	USTATUS1 (T21)			LVDS	2	out	3	U6-B4; USTATUS in FPGA file
ULV7	24	23	UDAISY_CLK (P17)		TDC daisy	LVDS	2	in	4	U6-C1; Equals 12 TDCs in chain.
ULV8	26	25	UDAISY_TOK_IN (P18)		TDC daisy	LVDS	2	in	4	U6-C2
ULV9	28	27		MCU_RESET	Reset	LVDS	2	in	4	U6-C3; Common to all MCUs, thru routing; enabled by J11.2-3
ULV10	30	29	FLEX_RESET_IN (V20)		Reset	LVDS	2	in	4	U6-C4; Reset is daisy chained thru PLDs, then OR'd with MCU_RESET signal
ULV11	32	31		TRAY_TOKEN		LVDS	2	in	5	U6-D1
ULV12	34	33	USPARE_IN1 (V22)			LVDS	2	in	5	U6-D2
ULV13	36	35	BUNCH_RST (U21)	BUNCH_RST		LVDS	2	in	5	U22-4
ULV14	38	37		UCONFIG_IN	position chain	LVDS	2	in	6	U22-3; fixed direction, position signal from upstream MCU
ULV15	40	39	TRIGGER (T18)	TRIGGER	Upstr_Trigger	LVDS	2	in	6	U22-2; fixed direction, thru routing
ULV16	42	41	CLK_10MHZ (M22,R22)	CLK_10MHZ	Multiplicity clock	LVDS	2	in	6	U22-1; fixed direction, thru routing
ULV17	44	43	UMULT0 (R18)		Multiplicity	LVDS	2	out	2	U7-1; fixed direction
ULV18	46	45	UMULT1 (R19)		Multiplicity	LVDS	2	out	2	U7-2; fixed direction
ULV19	48	47	UMULT2 (M18)		Multiplicity	LVDS	2	out	2	U7-3; fixed direction
ULV20	50	49	UMULT3 (M19)		Multiplicity	LVDS	2	out	2	U7-4; fixed direction

3.3.3. Downstream Connector Signals (J4)

Cable Signal	+ pin	- pin	FPGA name (Pin)	Signal name	Function	Logic	# of cable wires	Direction	Group	Notes
GND	2	1					2			
DCLK_40MHZ	4	3		DCLK_40MHZ_		LVPECL	2	out	0	Buffered between US/DS connectors
GND	6	5					2			
DLV0	8	7		AUX_SERIN		LVDS	2	in	7	U5-A1, safe when open-circuit
DLV1	10	9		AUX_TOKEN_IN		LVDS	2	in	7	U5-A2
DLV2	12	11		DS_STROBE_IN		LVDS	2	in	7	U5-A3
CAN	14	13		CAN_P, CAN_N	CANBus	LVDS	2	bidir	1	
DLV3	16	15	DDAISY_DATA (N1)		TDC daisy	LVDS	2	in	3	U5-B1; One daisy chain for 4 boards
DLV4	18	17	DDAISY_TOK_OUT (N2)		TDC daisy	LVDS	2	in	3	U5-B2; One daisy chain for 4 boards
DLV5	20	19	DSTATUS0 (N3)			LVDS	2	in	3	U5-B3
DLV6	22	21	DSTATUS1 (N4)			LVDS	2	in	3	U5-B4
DLV7	24	23	DDAISY_CLK (P1)		TDC daisy	LVDS	2	out	4	U5-C1; One daisy chain for 4 boards
DLV8	26	25	DDAISY_TOK_IN (N6)		TDC daisy	LVDS	2	out	4	U5-C2; One daisy chain for 4 boards
DLV9	28	27		MCU_RESET_B	Reset	LVDS	2	out	4	U5-C3; Common to all MCUs, thru routing
DLV10	30	29	FLEX_RESET_OUT (R6)		Reset	LVDS	2	out	4	U5-C4; Reset is daisy chained thru PLDs, then OR'd with MCU_RESET signal
DLV11	32	31		TRAY_TOKEN		LVDS	2	out	5	U5-D1
DLV12	34	33	DSPARE_OUT1 (P6)			LVDS	2	out	5	U5-D2
DLV13	36	35		BUNCH_RST		LVDS	2	out	6	U32-1; fixed direction
DLV14	38	37		DCONFIG_OUT	position chain	LVDS	2	out	6	U32-2; fixed direction, position signal to downstream MCU
DLV15	40	39		TRIGGER	trigger for TDCs	LVDS	2	out	6	U32-3; fixed direction, thru routing
DLV16	42	41		CLK_10MHZ	Multiplicity clock	LVDS	2	out	6	U32-4; fixed direction, thru routing
DLV17	44	43	DMULT0 (V1)		Multiplicity	LVDS	2	in	2	U26-4; fixed direction
DLV18	46	45	DMULT1 (V2)		Multiplicity	LVDS	2	in	2	U26-3; fixed direction
DLV19	48	47	DMULT2 (V4)		Multiplicity	LVDS	2	in	2	U26-2; fixed direction
DLV20	50	49	DMULT3 (W1)		Multiplicity	LVDS	2	in	2	U26-1; fixed direction

3.4. Programming Connectors (J6, J7, J8, and J10)

Straight 10-position 0.1” ribbon headers with polarity key slot (3M #30310-6002HB).
J8: ByteBlaster Active Serial programming of configuration eeprom #1 (FPGA). See also section 7.2.
J6: ByteBlaster JTAG header for FPGA debug.
J7: ByteBlaster JTAG header for CPLD programming
J10: Header for MCU programming. This connector requires a special version of the Microchip ICD2 programming cable when used with the “hockey puck” ICD2 programmer/debugger (See section 7.2.3).

3.5. Clock Connectors (J9, J14)

SMB connectors are for input (J9) and output (J14) of clock signals. Used for testing and verification of local clock oscillator function.
Emerson Power (Johnson) # 131-3701-261.

3.6. CANBus Connector (J12)

Three-pin 0.1” header used for testing CANBus. This connector is not used in normal operation. CANBus signals are available on the Upstream and Downstream connectors.

3.7. Testpoints

Testpoints (loop connectors) are available as follows:

Testpoint Ref.Des.	Driven by	Used For
TP1	MCU U19.40	CLK_DIV2_OUT (MCU Clock) or RC15 (see LED D9 section 5.2.6)
TP2	MCU U19.1	MCU_TEST signal
TP3	Regulator U56	Vcc_1.2 (FPGA Core Voltage)
TP4	Regulator U51	Vcc, Vcc_3.3_A1, Vcc_3.3_A2B, Vcc_3.3A2, Vcc_3.3_A3 (MCU, FPGA I/O, buffers)
TP5	Regulator U52	Vcc_2.5V (HPTDC voltage, switched)
TP6	Regulator U53	Vcc_3.3V_TDC (HPTDC voltage, switched)
TP7	Ground	Ground

4. Switches and JUMPERS

4.1. Board Position Switch (SW4)

Eight-position rotary switch read by MCU and used to set board position in tray (0 thru 7). Board position is used to form part of the address for CANBus messages. All jumpers are 0.1” pitch gold plated 0.025” square pins.

4.2. JU1 – CANBus Termination Jumper

Pin Positions	**Function**	**Installed (Shorted)**	**NOT Installed (Open)**
1-2	CANBus Termination	Enables 120 ohm CANBus termination	No CANBus termination.

4.3. JU2 – General Purpose Jumpers

General-purpose MCU-readable jumper inputs

Pin Positions	**Function**	**Installed (Shorted)**	**NOT Installed (Open)**
1-2	Oscillator Selected by MCU	Selects external clock from upstream ribbon cable)	Selects internal clock based on JU3 settings for internal clock source
3-4	Not used		
5-6	HPTDC JTAG select	MSBit of select = 1	MSBit of select = 0
7-8	HPTDC JTAG Select	LSBit of select = 1	LSBit of select = 0

HPTDC JTAG selects are only used when JTAG to HPTDC is active.

HPTDC JTAG SELECT		
MSB CONFIG_1.1	LSB CONFIG_1.0	Selection
0 (open)	0 (open)	No active TDC
0 (open)	1	TDC #3 (U4)
1	0 (open)	TDC #2 (U3)
1	1	TDC #1 (U2)

The value read by the MCU during a polling routine from JU2 (5-6 & 7-8) is written as the JTAG select field to TDIG FPGA configuration register CONFIG_1(1:0) as shown in the table, and the 2 bits are decoded to enable TDC selection as shown.

4.3.1. JU3 – Clock and J9 Routing Jumpers

Pin Positions	**Function**	**Installed (Shorted)**	**NOT Installed (Open)**

1-2	Internal clock from J9	Local clock comes from J9 as input	
3-4	J9 as I/O	J9 is I/O with FPGA Pin AA20	
5-6	Local clock	Local clock comes from Local Oscillator	
7-8	Local clock	Local clock input grounded.	

4.3.2. J11 – MCU Reset Source Select Jumper

Pin Positions	**Function**	**Installed (Shorted)**	**NOT Installed (Open)**
1-2	Reset Select	Allows MPLAB-ICD2 programming of MCU (U19)	MPLAB cannot access MCU
2-3	Reset Select	MCU reset via pushbutton or TCPU-generated reset.	

5. Key Components

5.1. Microcontroller (MCU)

The TDIG board has a 16-bit Microcontroller (U19, Microchip PIC24HJ64GP506-I/PT). This MCU has embedded controllers for CANBus, I2C, 12-bit Analog-to-Digital converter (DAC), and general purpose input/output pins. The MCU manages overall board operation; CANBus message protocol handling; configuration and control of the HPTDC chips; download and reconfiguration of the FPGA. The MCU can also download and run a second set of code for alternate operational behavior.

5.1.1. Microcontroller Ports and Signals

CPU Pin	CPU Port	Schematic Signal	Initialize Condition
16	RB0	MCU_PLD_DATA0	Bidirectional, Data to and from FPGA
15	RB1	MCU_PLD_DATA1	
14	RB2	MCU_PLD_DATA2	
13	RB3	MCU_PLD_DATA3	
12	RB4	MCU_PLD_DATA4	
11	RB5	MCU_PLD_DATA5	
17	RB6	MCU_PLD_DATA6	
18	RB7	MCU_PLD_DATA7	
21	RB8	MCU_PLD_CTRL0	Output, Low, Register address to FPGA
22	RB9	MCU_PLD_CTRL1	
23	RB10	MCU_PLD_CTRL2	
24	RB11	MCU_PLD_CTRL3	
27	RB12	MCU_PLD_CTRL4	Output, Low, Write / ~Read to FPGA
28	RB13	MCU_PLD_SPARE0	JTAG to HPTDC Source Select H=ByteBlaster, L=MCU
29	RB14	MCU_PLD_SPARE1	Output, Low, Strobe to FPGA
30	RB15	MCU_PLD_SPARE2	
2	AN16	TINO_TEMP1	RC1/AN16 as Analog Input
3	AN17	TINO_TEMP2	RC2/AN17 as Analog Input
46	RD0	MCU_TDC_TDI	JTAG from HPTDC via FPGA
49	RD1	MCU_TDC_TDO	JTAG to HPTDC via FPGA
50	RD2	MCU_TDC_TCK	JTAG to HPTDC via FPGA
51	RD3	MCU_TDC_TMS	JTAG to HPTDC via FPGA
52	RD4	MCU_EE_DATA	Input
53	RD5	MCU_EE_DCLK	Output, Low, Programming clock to EEPROM
54	RD6	MCU_EE_ASDO	Output, Low, Programming data to EEPROM
55	RD7	MCU_EE_NCS	Output, High, EEPROM enable
42	RD8	MCU_SEL_EE2	Output, Low=configure from default EPROM (#1)
43	RD9	MCU_CONFIG_PLD	Output, High=configure not active, rising edge will start configuration
44	RD10	EXPANDER_INT	Interrupt from I2C expanders
45	RD11	FLEX_RESET_IN	Flexible reset from tray
34	RF2	UCONFIG_IN	Serial ID input (not functional)
33	RF3	DCONFIG_OUT	Serial ID output (not functional)
35	INT0	MCU_HEAT_ALERT	RF6/INT0 as Interrupt Input (from U37 temperature sensor)

CPU Pin	CPU Port	Schematic Signal	Initialize Condition
61	RG0	Future CAN2	
60	RG1	Future CAN2	
4	RG6	I2CA_RESETB	Output, High, pulse low, leave High to allow I2C expanders to function
5	RG7	MCU_EN_LOCAL_OSC	Output, High to enable local oscillator
6	RG8	MCU_SEL_LOCAL_OSC	Output, High to select local oscillator
8	RG9	MCU_SEL_TERM	Output, High to enable CAN termination
63	RG12	PLD_DEVOE	Output, High to enable FPGA outputs
64	RG13	USB_RESETB	Output, Low to inhibit USB
62	RG14	PLD_RESETB	Output, High (cycles low after Init_Done) to reset FPGA
1	RG15	MCU_TEST	Output, Toggles to blink add-on LED
37	SCL1	I2CA_SCL	RG2/SCL1 as I2C clock, initialized by OpenI2C()
36	SDA1	I2CA_SDA	RG3/SDA1 as I2C data, initialized by OpenI2C()
32	SER_TX	PLD_SERIN	
31	SER_RX	PLD_SEROUT	
47	PGD2	PROGRAM_DATA	RC13/PGD2 as programming data
48	PGC2	PROGRAM_CLOCK	RC14/PGC2 as programming clock
58	C1RX	CAN_RX	RF0/C1RX as CAN 1 receive
59	C1TX	CAN_TX	RF1/C1TX as CAN1 transmit
39	CLKIN	MCU_CLK	CLKIN/OSC1 as clock input
40	RC15	CLK_DIV2_OUT	RC15/OSC2 as clock divided out

5.2. I^{2}C Bus

The I^{2}C bus is mastered by the MCU and provides connection to various display, parallel I/O, and Status monitoring functions. Devices on the bus are:

Chip U-	Chip Addr	Device	Used for
60	A0x	DS28CM00	Silicon Serial Number
38	98x	DAC7571	DAC Threshold voltage to TINO, Vout = 3.3V x (D / 4096)
37	94x	MCP9801	Board Temperature and over temperature alarm
36	44x	MCP23008	PLD, etc. Control and Status (ECSR)
35	42x	MCP23008	Inputs, inverting, with Pull Up = Jumpers, Board Position Switch and pushbutton
34	40x	MCP23008	Outputs, All 1's = LEDs OFF, 0-bit = LED ON

5.2.1. Silicon Serial Number

This chip (U60, Maxim DS28CM00) provides a 7-bit-unique hardware serial number specific to each board. This number is distinct from the “board serial number” labeling.

Since the lowermost (0th) byte is always 0x70 it is not transmitted in response to the CANBus read-hardware serial number message.

5.2.2. Threshold-Setting DAC

The TDIG board has a 12-bit Digital-to-Analog converter (DAC) as part of the I^2C bus (U38, TI DAC7571IDBVT). The DAC is set via the MCU as part of start up and in response to CANBus messages. The MCU initializes the DAC to a nominal Vout of 2500 mV during start-up. This start up value is defined in the TDIG_BOARD.H file.

The DAC word is related to the threshold voltage sent to the TDC-TINO connector (J13) by the following (idealized) equation:

$V_{out} = 3.3V * (D / 4095.)$

The maximum threshold setting is approximately 3300 mV, DAC word = 0xFFF. The minimum threshold is approximately 0mV, DAC word = 0x000.

Example CANBus messages addressed to TDIG board 0 are shown below:

DAC output	**Send**			**Receive**		
Volts (nominal)	Message	Length	Payload	Message	Length	Payload
0	0x102	3	0x08 0x00 0x00	0x103	2	0x08 0x00
3.3	0x102	3	0x08 0xFF 0x0F	0x103	2	0x08 0x00
1.	0x102	3	0x08 0xD9 0x04	0x103	2	0x08 0x00

5.2.3. Board Temperature and Alarm

This chip (U37, TI DAC7571) provides a means for the MCU to determine board temperature near the center of the board. The part is also capable of generating an MCU interrupt in over-temperature conditions (implemented in TDIG MCU code version x.x or later).

5.2.4. Extended Control and Status Register (ECSR)

This chip (U36, Microchip MCP23008) provides for controlling and monitoring additional status lines on the board. These lines provide for the following:

Bit (MSB to LSB)	**Signal Name**	**Direction**	**Used For**
7	SPARE_PLD	Input	Extra bit to FPGA
6	TINO_TEST_MCU	Output	Provide programmable test pulse output to TINO
5	ENABLE_TDC_POWER	Output	High level turns on power to HPTDC. Initialized with power disabled
4	TDC_POWER_ERROR_B	Input	Returns power-OK signal after HPTDC power-on unless a fault has occurred
3	PLD_nSTATUS	Input	Returns FPGA status
2	PLD_CRC_ERROR	Input	Changes state if the current FPGA configuration experiences a bit-upset (SEU) if the FPGA code supports that operation. State change causes Interrupt
1	PLD_INIT_DONE	Input	Returns high level when FPGA has initialized
0	PLD_CONFIG_DONE	Input	Returns high level when FPGA has successfully configured.

5.2.5. Switch and Jumper Register

This chip (U35, Microchip MCP23008) provides for reading additional, relatively static information about the status of switches and jumpers. These lines are initialized with internal pull-up resistors and inverting inputs to thus provide a high level signal (1 bit) when a pull-down jumper is installed or switch is activated:

Bit (MSB to LSB)	**Signal Name**	**Direction**	**Used For**
7		Input	Jumper JU2 pins 1-2, 1==jumper installed
6		Input	Jumper JU2 pins 3-4, 1==jumper installed
5		Input	Jumper JU2 pins 5-6, 1==jumper installed
4		Input	Jumper JU2 pins 7-8, 1==jumper installed
3		Input	Board Position SW4 bit 1
2		Input	Board Position SW4 bit 2
1		Input	Board Position SW4 bit 4.
0		Input	Pushbutton Status 1==pressed

Note that due to board layout, board position bits 1 and 4 are interchanged from the usual weighting; this is taken into account by the MCU firmware.

5.2.6. LED Indicators

For TDIG- MCU firmware version 11D or later (Release 5 or later) any or all LEDs may illuminate briefly during MCU firmware initialization and start-up. LEDs D9 and D11 are connected to specific hardware signals as described below. LEDs D0 through D7 are controlled through the MCU I^{2}C bus I/O expander port (U34, Microchip MCP23008). Writing a “0” bit to the port turns the corresponding LED on.

Once the MCU program has initialized LEDs are used as follows *unless* overridden by a CANBus message (write_led, 0x1?4, length 2, payload ‘0x0A’ <byte>).

Reading from top to bottom:

D0 (green) – This LED reflects the current state of jumper position JU2 pins 7-8.
Jumper installed results in LED-ON. This jumper position is used in conjunction with JU2 pins 5-6 to select an HPTDC for manual JTAG access. Jumper installed = Bit 2^0 selected.

D1 (green) – This LED reflects the current state of jumper position JU2 pins 5-6.
Jumper installed results in LED-ON. This jumper position is used in conjunction with JU2 pins 7-8 to select an HPTDC for manual JTAG access. Jumper installed = Bit 2^1 selected.

D2 (green) – This LED reflects the current state of jumper position JU2 pins 3-4.
Jumper installed results in LED-ON. This jumper position is not currently used.

D3 (green) – This LED reflects the current state of jumper position JU2 pins 1-2.
Jumper installed results in LED-ON and selects “Tray Clock” from connector J5 as the board clock source.

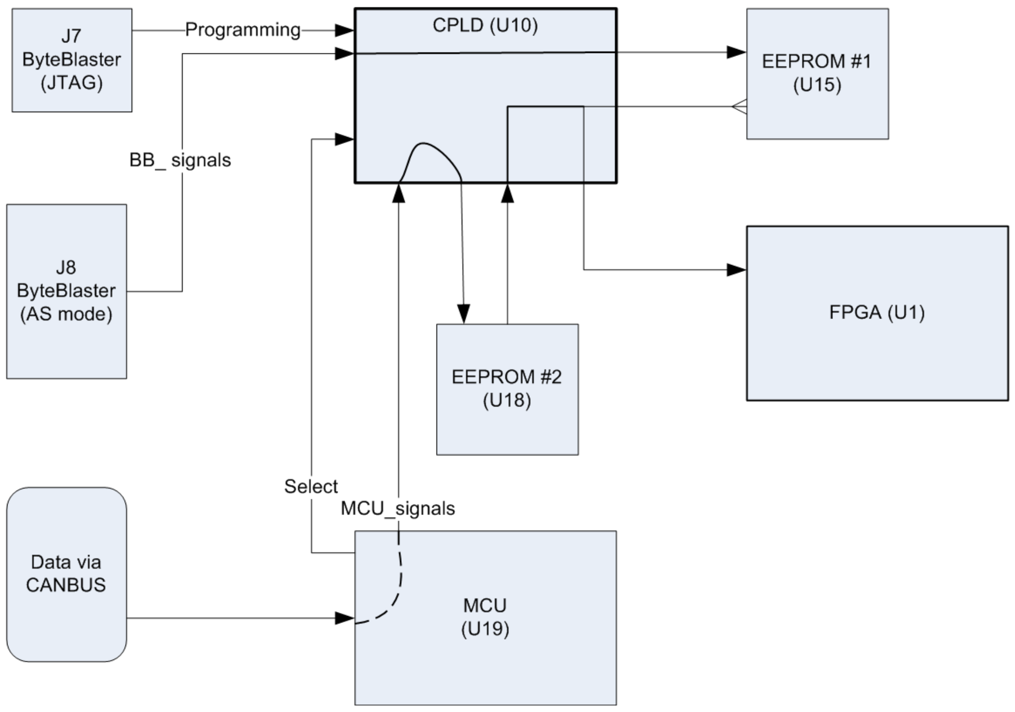
D4 (green) – This LED lit indicates an error condition. It will be illuminated if the configuration (647-bit setup string) read-back from one or more of the HPTDC chips differs from what was programmed.

D5 (green) – This LED is not used at present.
D6 (green) – This LED will be lit when the HPTDCs have been configured.
D7 (amber) – This LED is illuminated when the MCU requests the HPTDC power regulator "ON" condition.
D9 (green) – This indicator is hard-wired to the MCU "G"-Register bit 15 (MCU_TEST) and test-point TP2. A low-level at TP2 corresponds to LED-on. There is a 50% duty-cycle square wave generated by the MCU to this test point when the MCU program is running in the "idle" loop after all initializations are done. The LED will appear slightly "dim" relative to the others when the idle loop is functioning.
D11 (green) – This indicator is hard-wired to the FPGA. If the FPGA has not been programmed it will be off. After the FPGA initializes, it will normally be lit. Pressing SW3 will cause the LED to turn off.

The normal idling condition of a TDIG board will be:
D11, D9, D7, D6, D3 = ON; the remaining LEDs = OFF

5.3. Small PLD (CPLD)

The CPLD (U10, Altera EPM3064ALC44) multiplexes sources of configuration data to the FPGA (U1) and EEPROM#1 and provides a path from the MCU to EEPROM#2 for download programming of FPGA configurations. The sources are: ByteBlaster connector active serial via J8 into EEPROM#1; EEPROM#1 (U15); and EEPROM#2. Programming of the CPLD is through J7 ByteBlaster connector running in JTAG mode. Selection of which EEPROM to use for configuration is made by the MCU.



6. Configuration and Operation Reference

6.1. Board and TDC numbering

There are 8 TDIG boards per tray. Board positions are set using SW4 such that boards are numbered consecutively 0 – 7. The board physically farthest from the TCPU is set to be position 0.

For TDIG MCU firmware release 6.5 and greater TDCs will be automatically configured so that their device ID field (included in each TDC data word) will be set according to the table below.

Board #	TDC #s		Board #	TDC #s
0	0,1,2		4	0,1,2
1	4,5,6		5	4,5,6
2	8,9,10		6	8,9,10
3	12,13,14		7	12,13,14

6.2. Power Sequencing

The MCU (Microchip PIC24HJ64GP506) runs entirely from 3.3 Volts at 40 MHz. The MCU internal core generates its own 2.5 Volt supply. After it's internally controlled Power-on reset, the MCU can control and monitor the remaining voltages in the system.

The remainder of power on sequencing is controlled by the MCU. The MCU enables dedicated regulators for the TDC I/O and core voltages. Upon enable from MCU ("ENABLE_TDC_POWER" valid), the TDC I/O regulator is enabled. That regulator's output voltage then enables the TDC core regulator.

To avoid excessive inrush current during power-up, the MCU has a programmed delay of 10-seconds PLUS 2-seconds times the board-position setting before enabling TDC power. This delay is effective at power-up reset and during processing of a CANBus Reset-into-code-space-1 command.

6.3. FPGA Configuration

The FPGA (U1, Altera EP2C20F484C8) is configurable from two non-volatile sources. One source, EEPROM #1 (U15, Altera EPCS4SI8N), is programmed via the ByteBlaster header (J8, Active Serial Mode) using either Altera Quartus, or Altera Stand-Alone Programmer software. EEPROM#1 is not accessible for reprogramming except via the ByteBlaster header. The second source is EEPROM#2 (U18, ST Microelectronics M25P40-VMN6P), which is programmed by the MCU using a downloaded compressed binary file transmitted over the CANBus. The FPGA is configured at start-up from one of the two EEPROMs. The FPGA can be reconfigured from either EEPROM #1 or EEPROM #2 using CANBus commands. If reconfiguration using EEPROM #2 fails (times-out), EEPROM #1 will be used instead.

6.4. Resets

6.4.1. MCU Reset

Power on reset occurs internally to the MCU as a result of power-up. The rise of input voltage from 0 to 3.3 Volts must be accomplished within xxx seconds. Too slow a rise-time may cause the MCU to hang. After initialization, the MCU resets the TDCs, initiates PLD configuration, resets the PLD and configures the TDCs. TDC reset is independent of PLD and HPTDC resets.

6.4.2. HPTDC Reset

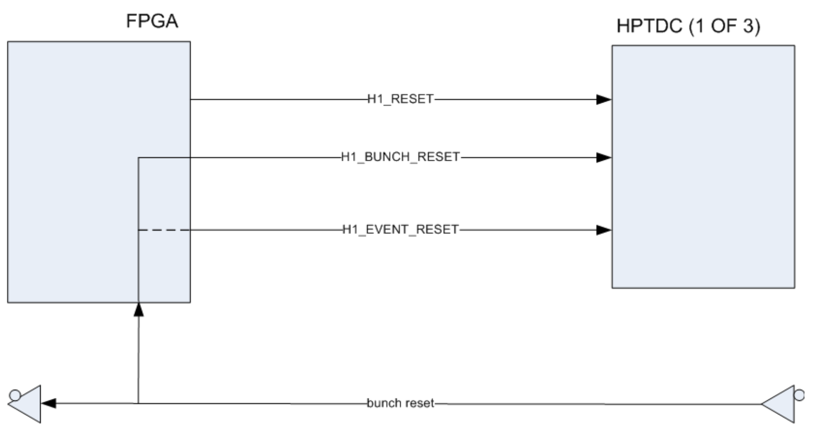
In addition to the hardware reset signal from the FPGA to the HPTDCs, the MCU provides a sequence of control words via the JTAG data path which perform the remainder of the required HPTDC Reset and PLL initialization sequence.
The sequence of control words sent to each HPTDC during the reset sequence is:
0xFFFFFFFF, 0x3FFFFFFF, 0x9FFFFFFF, 0x1FFFFFFF, 0x9FFFFFFF
This is followed by the default final control word.

6.4.3. Bunch Reset

The Bunch Reset condition is provided to each of the HPTDC chips as diagrammed below. To use the Bunch Reset as a driver to HPTDC Master Reset the FPGA firmware can:

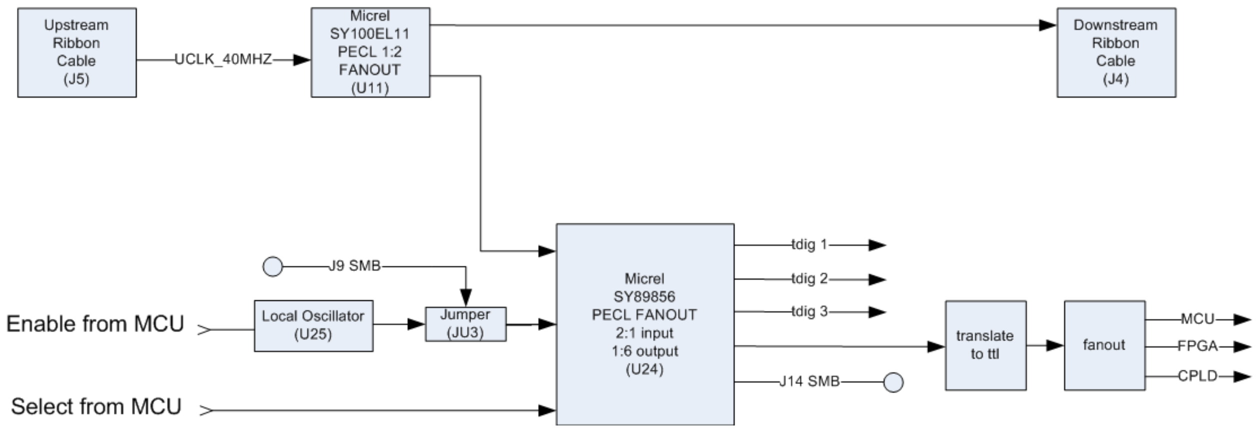
1. Connect Event Reset from the FPGA to the existing Bunch Reset signal
2. Set HPTDC configuration bit 152: ‘enable master reset on event reset’
3. Set HPTDC configuration bit 156: ‘enable direct event reset’

As an alternative, the Master Reset input bit from the MCU can be OR’d with the Bunch Reset input.



6.5. Clock Distribution

A simplified drawing of board clock distribution paths is shown below:


Upstream Ribbon Cable (J5)
UCLK_40MHZ
Micrel SY100EL11 PECL 1:2 FANOUT (U11)
Downstream Ribbon Cable (J4)
J9 SMB
Enable from MCU
Local Oscillator (U25)
Jumper (JU3)
Select from MCU
Micrel SY89856 PECL FANOUT 2:1 input 1:6 output (U24)
tdig 1
tdig 2
tdig 3
J14 SMB
translate to ttl
fanout
MCU
FPGA
CPLD

The Upstream input clock UCLK_40MHZ is buffered and split with U11, one output leg goes directly to the downstream connector. With this arrangement, a failure in the clock switching control will not affect operation of downstream boards.

The TDIG system on-board clock has 2 possible sources: a) the second copy of the upstream input clock originating from TCPU; or b) the on-board oscillator (U25).

Source selection is made by the MCU through the 2:1-in / 5-out PECL fanout buffer (U24). The attenuated (zero jitter PECL to LVDS conversion) clocks go to the TDCs, plus a conversion to TTL and subsequent fanout to the MCU, CPLD and FPGA.

The local oscillator (U25) can be disabled by the MCU when not in use.

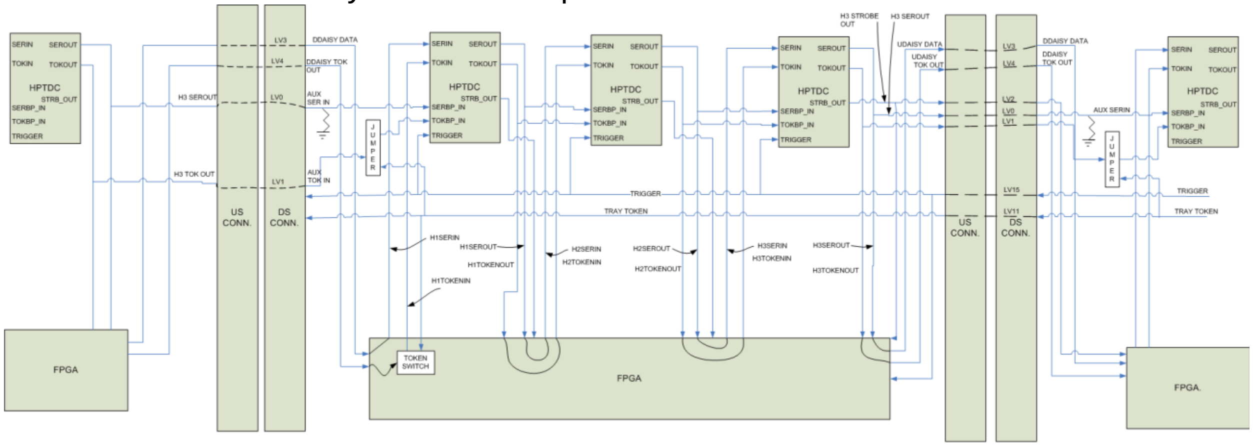
A test clock input from an RF connector (J9) can be used as a local clock when selected via jumper JU3.

During start-up and reset initialization, the MCU starts-up using its own internal oscillator; reads the jumper setting JU2 and continues with the selected local or external oscillator. Under CAN Bus commands the clock source can be changed. If the external clock fails in operation, the MCU will automatically switch to its own internal oscillator and execute a trap. The trap code will switch back to the local oscillator. An external clock failure does not hang the MCU. The MCU may hang if a local oscillator is selected but not operating.

The local oscillator is also routed to the FPGA for use as an asynchronous source for testing, including the generation of the tray test pulse that is sent to TINO.

6.6. Normal and Auxiliary Data Readout Paths

The Normal and Auxiliary data readout paths are shown below:



The Normal readout path is managed through the FPGA. The Auxiliary readout path does not use the FPGA and can be initiated using the following sequence:

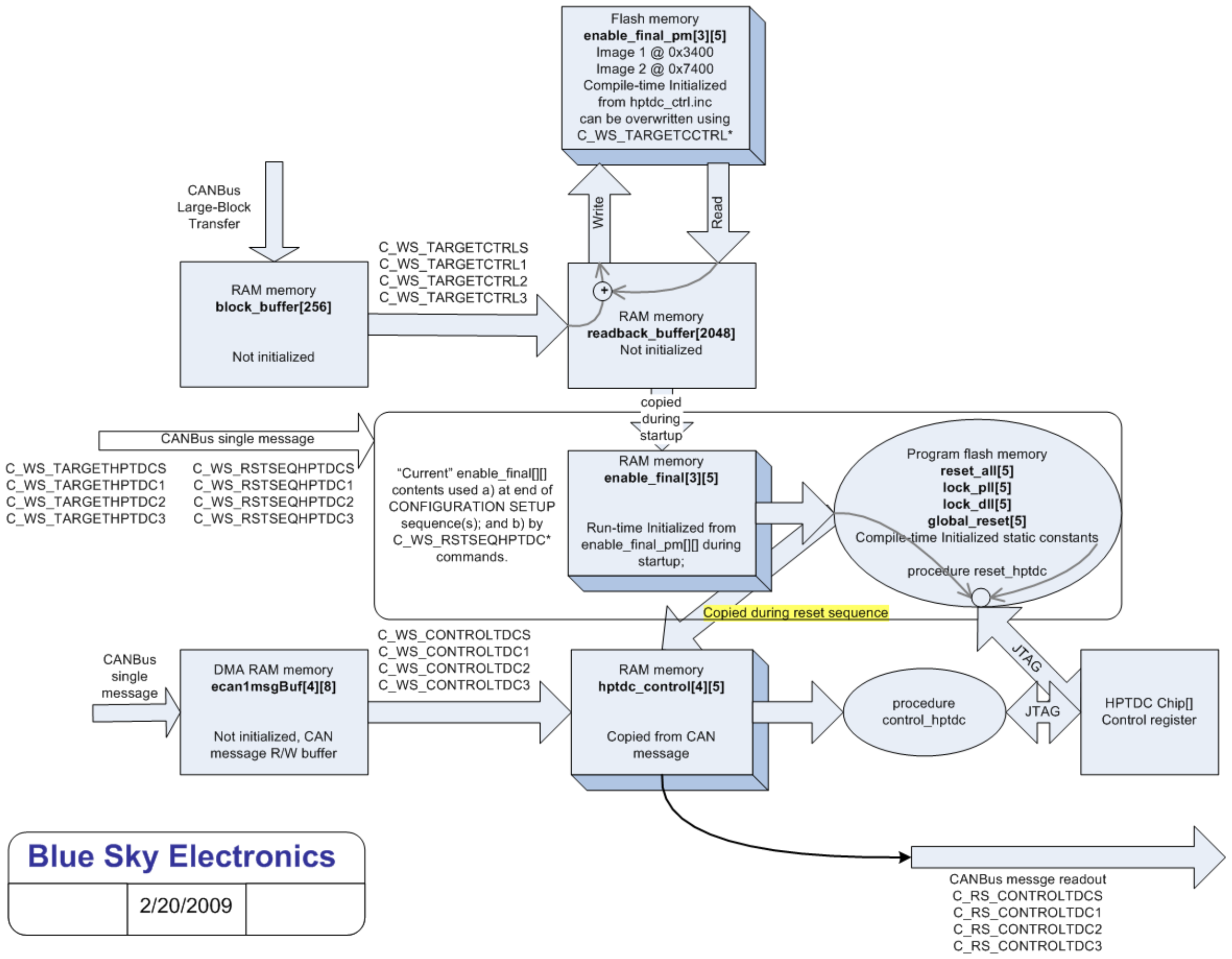
1. Set jumper J17 properly on each TDIG in the readout chain for AUX data path token routing.
   1.1. Jumper 2-3 if board is farthest from TCPU;
   1.2. Otherwise jumper 1-2.
2. Set HPTDC configuration bit #44 “SELECT BYPASS INPUTS” to “1” in all TDCs in the readout chain.
3. Set TCPU FPGA configuration register bit CONFIG_14.7 = “1”
4. Unless power is cycled, a reset to the TCPU state machines, and possibly the HPTDCs, will probably be necessary when switching between modes (toggle CONFIG_1.0 HI then LO using a CANBus Write-Register command such as (“002 5 0E 01 00 01 00”).

6.7. HPTDC Control and Configuration

6.7.1. HPTDC Control Words

Control words (40 bits) are written and status is read back using the MCU-JTAG data path. Complete HPTDC device reset includes multiple writes of the control words with different patterns to toggle control bits and examine lock status of the PLL and DLL. Refer to the HPTDC chip device datasheet/user manual for details of bit meanings. Data flow for HPTDC Control words is shown below:

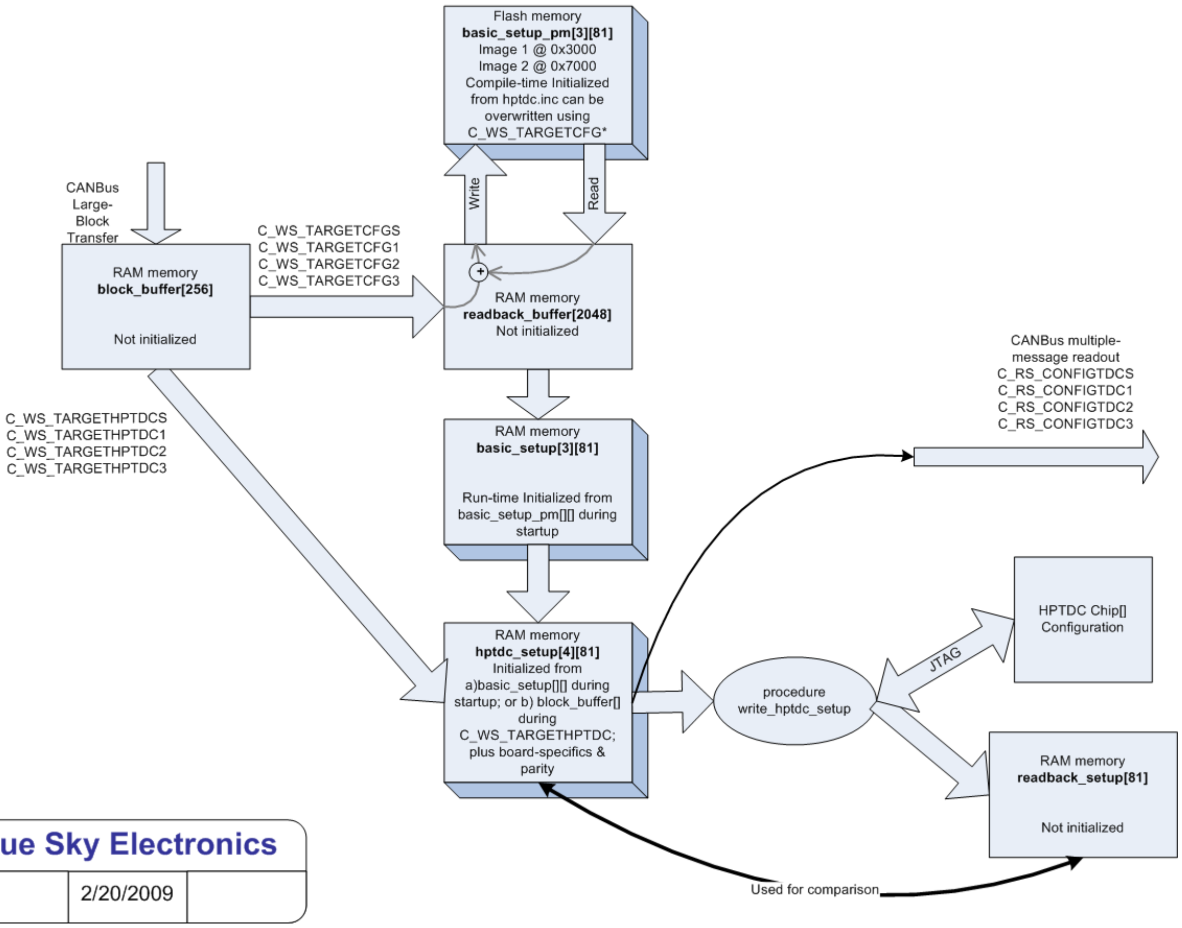
TDIG MCU Data Flow for HPTDC CONTROL Word (40 bits)



6.7.2. HPTDC Configuration Bits

HPTDC Configuration bits (647 bits) are used to control HPTDC device options and calibration. Refer to the HPTDC chip device datasheet/user manual for details of bit meanings. Data flow for HPTDC Configuration bit string processing is shown below:

TDIG MCU Data Flow for HPTDC CONFIGURATION SETUP (647 bits)



Blue Sky Electronics		
	2/20/2009	

7. Option Reference

7.1. Board Firmware Identifiers

Beginning at TDIG firmware version 11D CANBus messages are available which enable reading the “Firmware Identifiers” for the MCU and FPGA components of boards.

Method:

1. Issue the CANBus command: “Read board”, length 1, message 0xB1.
2. The response will be a “Read reply”, length 4, message 0xB1 <2-byte MCU firmware ID><1-byte FPGA ID>.

Example:

1. Send: 0x104, length 1, 0xB1
2. Receive: 0x105, length 4, 0xB1 0x0D 0x11 0x55

Details: IDs are built into the firmware components as follows:

1. The MCU firmware ID is defined through bytes FIRMWARE_ID_0 and FIRMWARE_ID_1 in the main program source file “TDIG-D_VER...C”.
2. The FPGA firmware ID is set through definition of the contents returned to “data7x” when the MCU reads FPGA register 7 (currently “CONSTANT five_five_byte...”). This is defined in module TDIG_pins1.vhd in the TDIG Altera source directory.

7.2. Hardware Programming Using Pre-Compiled Files

7.2.1. CPLD Programming

Note that the CPLD must be programmed prior to attempting to program the FPGA.

a. Start Quartus; select “run with existing license” (Or start Quartus Programmer stand-alone program and skip to step c.)
b. Select “Tools”; select “Programmer”, a blank programming screen opens titled “QuartusII – [chain1.cdf].
c. Select “File”; select “Open”; select “Files of type: Programming Files...”
d. Browse to the location containing “SmallCPLD.CDF”; click “Open”.
e. Select “Open a new Programmer window listing this file”; click “OK”.
f. The chain will open.
g. The display will show “Mode JTAG”; “Progress 0%”.
h. The release programming filename; “Device EPM3064AL44”; Checksum as documented in the release document; “Usercode” all F’s should display.
i. The two boxes labeled “Program/Configure” and “Verify” must be checked.
j. Connect ByteBlaster cable to board header J7; apply power to board.
k. Click the button labeled “Start”. The program/verify sequence will run and programming progress will be indicated in the “Progress” bar.

7.2.2. FPGA Programming

Note that the CPLD must be programmed prior to attempting to program the FPGA.

a. Start Quartus; select “run with existing license” (Or start Quartus Programmer stand-alone program and skip to step c.)
b. Select “Tools”; select “Programmer”, a blank programming screen opens titled “QuartusII – [chain1.cdf].
c. Select “File”; select “Open”; select “Files of type: Programming Files...”
d. Browse to the location containing “TDIG_pins1.CDF”; click “Open”.
e. Select “Open a new Programmer window listing this file”; click “OK”.
f. The chain will open.
g. The display will show “Mode Active Serial Programming”; “Progress 0%”.
h. The release programming filename; “Device EPCS4”; Checksum as documented in the release document; “Usercode” all 0’s should display.
i. The boxes labeled “Program/Configure” and “Verify” must be checked for both the EPCS4 and the sub-device “Page 0” (4 check-marks total).
j. Connect ByteBlaster cable to board header J8; apply power to board.
k. Click the button labeled “Start”. The program/verify sequence will run and programming progress will be indicated in the “Progress” bar.

7.2.3. MCU Programming

a. Start MPLAB
b. Select “Programmer”;
c. Select “MPLAB ICD2”. If the MPLAB module is connected to the board several messages will appear, ending with “MPLAB ICD2 Ready”.
d. Select “File”; select “Import”.
e. Browse to the location containing the release filename.hex
f. Select the filename and click “open”. The message “Loaded should appear in the MPLAB “Output” window.
g. Connect Blue Sky Electronics programming cable from MPLAB ICD2 to board header J10; apply power to board.
h. Select “Programmer”; then select “Program”. MPLAB will erase, program, and verify the MCU code image. The message “Programming succeeded will appear in the “Output” window. (The yellow “Program target device” icon performs the same action as the “programmer/program” selection sequence).

7.3. Reading HPTDC Data Using JTAG

1. Set up data to TDCs:
   a. Mount front end test fixture to TDIG
   b. Set up data input to front end test fixture

OR

2. Enable test data input to chan1 on TDCs using FPGA configuration registers

3. Attach ByteBlaster to JTAG readout cable and test header

4. Select or edit the “tdig.jam” file so the config_basic command gives the appropriate TDC configuration.

5. Set the TDC ID field. Edit “data_file” command to give the desired number of words in the output file.

6. Open a DOS window and issue the following commands:

   a. CONFIGURE THE HPTDCs and LOCK HPTDC PLLs
   ```
   jam_tdc -aconfig_basic tdig.jam
   jam_tdc -alock tdig.jam
   ```

   b. TURN OFF THE DATA SOURCE

   c. RESET THE HPTDCs:
   ```
   jam_tdc -areset tdig.jam
   ```

   d. CHECK THAT THE READOUT FIFO IS EMPTY
   ```
   jam_tdc -astatus tdig.jam
   ```

   e. TURN ON THE DATA SOURCE

   f. CHECK THAT THE READOUT FIFO IS FULL
   ```
   jam_tdc -astatus tdig.jam
   ```

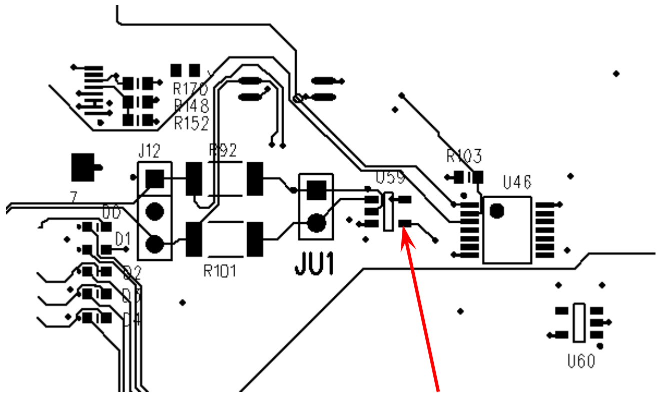
   g. READ SINGLE DATA WORDS – Check for correct format and TDC ID.
   ```
   jam_tdc -adata tdig.jam
   ```

   h. READ DATA to FILE (after editing JAM code to control the number of words)
   ```
   jam_tdc -adata_file -otesta.dat tdig.jam
   ```

7.4. Checking the State of "Automatic" CANBus Termination

Some versions of MCU code (before the version identified TDIG-D11E) had the CANBus set to be terminated by default. This leads to incorrect bus operation due to too-many termination points when multiple TDIG boards are connected to the same CANBus. Board versions TDIG-D, TDIG-E, TDIG-F rev 0, and TDIG-F rev 1 can exhibit this behavior if built with U59 populated.

Checking the state of the "automatic" termination feature is straightforward.
The figure below shows a portion of the top side of the PCB in the area adjacent to the CANBus manual termination jumper JU1.



1. IF the chip location U59 is *un-populated* (bare pads), the MCU-controlled switch is *not present* and automatic termination cannot take place. This is the as-built condition of TDIG-F boards.
2. IF the switch chip at U59 is *populated* (chip is present), the MCU-controlled switch is present and the automatic termination can occur. If this is the case, the actual state of the control line can be observed by careful probing of pin-4 of chip U59. The arrow in the above figure shows the probe-point. Probing can be done with either an Oscilloscope or a Digital Volt Meter (DVM). With TDIG board power-on and the MCU running (observe the normal pattern of LEDs) observe the signal voltage at pin 4 of U59 relative to ground (testpoint TP7).
    a. IF the voltage is a "high" level (greater than 1.4 Volts DC), the terminator switch is ON. The board has termination equivalent to the jumper JU1 being present.
    b. IF the voltage is a "low" level (less than 0.8 Volts DC), the terminator switch is OFF. This is equivalent to the jumper JU1 being absent and the board has no termination.
3. If automatic termination is absent or turned off, termination can be applied by installing a jumper across JU1.

7.5. MCU Reprogramming via CANBus

0. Introduction:
Some considerations in section 2 relating to the alternate vector table are subject to change when alternate download code is developed.
This note describes programming and linking considerations and the download procedure for programming a second code image into the PIC24HJxxx processors used on the Blue Sky Electronics TDIG-D or later and TCPU-B or later boards.

1. Initial programming:
Presuming the MCU has been programmed (using MPLAB or via some other means) with MCU code version 11d or later for TDIG, version 1F or later for TCPU.

2. Considerations for the design and coding of the "second image":
The PIC24HJxxx series processors have the ability to re-program while running. The download procedure takes advantage of this ability and the presence of an "alternate" interrupt vector table. The ability to return to the "original image" code necessitates some explicit coding which might not otherwise be needed.

a. Any setup of CPU registers and configurations must be done explicitly in the second-image code.
b. It is not possible, except through explicit reprogramming, to alter the CPU identification fields. ID fields, Clock select fields, and other special "one-time" fields are not generally reprogrammed by the download procedure.
c. The second-image is entered from the first image via a Jump to location 0x4000 with interrupts disabled and the Alternate Interrupt Vectors selected in the CPU INTCON register. The user should readjust interrupt priorities during start-up of the second-image code but should NOT select "standard" interrupt vector mapping.

d. The first- and second- images share the DMA and RAM space.
e. Upon startup, the RAM space is cleared by the C-Language-provided startup routines CRT0() which is entered automatically at startup and reset.

3. Considerations for linking "second image" code:
Stand-alone ("First" images) will use the normal Microchip provided linker script file for the processor. Downloadable "Second" images MUST be linked using the modified linker script file (download-P24HJ....GLD). The special linking control file provides the following:

a. Reassignment of the "origin" to a location above the end of the first-image code. Currently location 0x4000 is defined as the start location for second-image code.
b. Reduction in the amount of available program space to allow for the first image to be retained.
c. Remapping the "Standard" interrupt vector table into the space normally occupied by the "Alternate" interrupt vector table. The user does NOT use the "Alt..." form of the definition for interrupt service routines. This simplifies the development and debugging of code through MPLAB as though it were the "first image". It is possible to use the Alt... form of the interrupt vectors; in which case the remapping/reassignment provided by the .GLD file is not needed and should be removed.
d. The remainder of the usual link options remain; however certain of the fields will not be downloaded, specifically:
    1) The RESET vector at location 0
    2) The Oscillator select/control registers FOSCSEL, FOSC locations 0xF80006 and F80008.
    3) The Watchdog and Power-On Reset registers FWDT and FPOR at locations 0xF8000A and 0xF8000C
    4) The Processor ID fields (FUID0...3) at locations 0xf80010 through 0xF80016.
e. After linking, examine (using "Notepad" or other text editor) the linker output map (filename.MAP). It is important to verify that the "__resetPRI" external symbol references location 0x4000. This insures that the correct startup code will be executed (as though a hardware RESET had occurred) when control transfers after the download.

4. Downloading the second image to the MCU:
The result of the linking procedure is a file with the .hex extension. This file describes the in-memory image of the completed program. This image is parsed and downloaded using program MCU2.EXE via a series of CANBus block-transfer messages to the "first-image" code running in the MCU. The MCU transfers the image to MCU program (flash) memory. Once the download is complete, a final CANBus message triggers transfer of control to the new image. The download protocol as implemented by the program MCU2.exe proceeds as follows (MCU2.cpp):

a. MCU2 is invoked from the command line with arguments describing the board-id and hex filename to be downloaded. (e.g. MCU2 0x10 download-test.hex) The program proceeds as:
    1) The CANBus support .DLLs are loaded.

2) The HEX file is opened.
3) The HEX file is parsed and records interpreted:
    a) "Address" type records are not transmitted; they update address variables within the MCU2 program controlling the block-target-MCU protocol messages.
    b) "Data" type records are transmitted using the CANBus HLP3 block transfer protocol if they are destined for "allowed" addresses. In order to preserve the original interrupt vector table and first-image code, the memory areas allocated to those are NOT written. Only addresses 0x100 through 0x200 (Alternate vector table) and 0x4000 through 0x8FFF are allowed for download. Other addresses are skipped by MCU2.
    c) As blocks are completed, they are "written" by the MCU to the flash memory. Because the first-image low-memory area (interrupt vector tables) needs to be preserved, a special type of "erase" command, ERASE_PRESERVE is used in this memory region.

b. When the MCU2 program completes, files and .DLLs are closed.
c. The MCU2 program reports its operation to the .DOS window which may be saved for examination.

5. Starting the second image:
After downloading the second image, the first image is still operating. To begin execution of the second image, a CANBus HLP message sequence is issued:
a. board|WRITE; length 5; payload: 0x8D 0x69 0x96 0xa5 0x5a is issued.
b. This causes the MCU code to reply with: board|WRITE_REPLY; length 2; payload: 0x8D 0x0.
c. The MCU locks-out interrupts, sets the interrupt vector table selector to "alternate" and does a Jump to address 0x4000. The startup code CRT0 within the second image begins executing to initialize the C-language environment, followed by "main()" being invoked.

6. Restarting the first image:
a. From the second image: A call to the C-routines exit(); or reset(); or execution of the assembly instruction "reset" will cause a jump through the original reset vector (which was not changed during download) and cause the first image code to be restarted.
b. From within the first image: If the second image was not started: then issuing the CANBus HLP command board|WRITE length 5; payload: 0x8F 0x69 0x98 0xa5 0x5a will cause the equivalent of a power-on reset to take place which restarts the first image.

7.6. FPGA Reprogramming via CANBus

Refer to the *CANBus High-Level-Protocol* document for more details on the proper sequence of CANBus download commands.

The Altera FPGA (U1) on boards TDIG-D, TDIG-E, and TDIG-F can be reprogrammed and reconfigured via CANBus using a download procedure and FPGA reconfiguration commands as follows assuming the CANBus has been configured and operates properly:

1. The TDIG-D, TDIG-E, and TDIG-F board(s) must be running MCU code version 11C or later
2. Using Altera Quartus software, compile the new FPGA code.   Be sure to use "compressed bitstreams" and "Generate RBF" file options in Quartus as shown below. The resulting .RBF file will be used to download to the auxiliary configuration EEPROM.

```
In Quartus under:
<Assignments>
  <Device>
    <Device & Pin Options>
      <Configuration tab>
        Enable (check) the box "Generate compressed bitstreams".
      <Programming Files tab>
        Enable (check) the box "Raw Binary File (.RBF)"
      <General tab>
        Enable (check) the boxes "Enable INIT_DONE output" and
          "Enable device-wide reset DEV_CLRn"
```

3. Close the PcanView utility. (Only one attachment to the PEAK CANBus DLL files is allowed).

4. From a command line interface ("DOS box"), invoke the download utility eeprom2.exe using the command line format:

```
        eeprom2  <brd#> <filename.rbf>
```

Where: `eeprom2` is the name of the utility eeprom2.exe
`<brd#>` is the board number of the target board, set by the "board position switch(es)".
`<filename.rbf>` is the name of the .RBF file generated in step 2.

This download procedure may take several minutes and generates numerous diagnostic and progress messages. If an abnormal condition occurs, the download process will stop and display an "error" message. When a successful download is complete a success message will appear. At that point it is safe to close the DOS window and return to the PcanView (or other CAN monitor/control) program.

5. At the conclusion of step 4, the FPGA is still running from the original (unmodifyable) configuration. The final step is to cause the FPGA to reconfigure using the newly downloaded image. This may be done using the PcanView program by issuing a Reconfiguration message (Write Command) to the MCU:

Message ID	Length	Payload Contents (hex):
<brd#>2	**5**	**8A 69 96 a5 5a**

e.g. "`012 5 8A 69 96 a5 5a`" will cause reconfiguration from EEPROM #2 of the board ID'd 1.

The message reply will consist of the following message: `<brd#>3   3   8A   00   r7`

Where: `<brd#>3` indicates a "Write Reply" from <brd#>

`3` is the length of the reply

`8A` is the target of the write (this value will be `89` for reconfiguration from EEPROM#1)

`00` is the status of the write (`00` is normal return)

`r7` is the contents of the "ID Register-7" following reconfiguration.

6. Following reconfiguration, the FPGA is reset and the default register contents are loaded.

7. Operation from the EEPROM #1 configuration can be restored by sending a message with contents:

Message ID	Length	Payload Contents (hex):				
<brd#>2	**5**	**8A**	**69**	**96**	**a5**	**5a**

8. Quick Start

For a quick start of a TDIG board to be used for testing, the following
JU1 – CAN bus termination. Short 1-2 enables termination.

JU2 – General purpose MCU jumper inputs.
  Short 1-2 to select external oscillator (40 MHz clock from TCPU)
  Open 1-2 to select on-board oscillator
  3-4 not used
  5-6 msb of JTAG tdc select (short = 1) (only used with JTAG TDC I/O)
  7-8 lsb of JTAG tdc select (short = 1) (only used with JTAG TDC I/O)

JU3 – Clock and J9 routing
  1-2 short allows local clock from J9
  3-4 short allows J9 as I/O with FPGA /pin AA20
  5-6 short allows local clock from on-board oscillator
  7-8 short grounds local clock input

Programming header placement: 4 connectors in upper left hand side of board.
J8 in corner
J6 next below
J7 next below
J10 next below

J11: MCU reset select
  Short 1-2 to allow ICD2 programming
  Short 2-3 to allow hardware reset from SW3 or TCPU reset signal
  Default: Leave jumper in **short 1-2** position

J12: CAN bus header.
  J12/pin1 = CAN_L
  J12/pin2 = GND
  J12/pin3 = CAN_H

J14: Board clock out. 40 MHz, PECL levels. Clock that goes to FPGA, HPTDCs, after all clock switching.

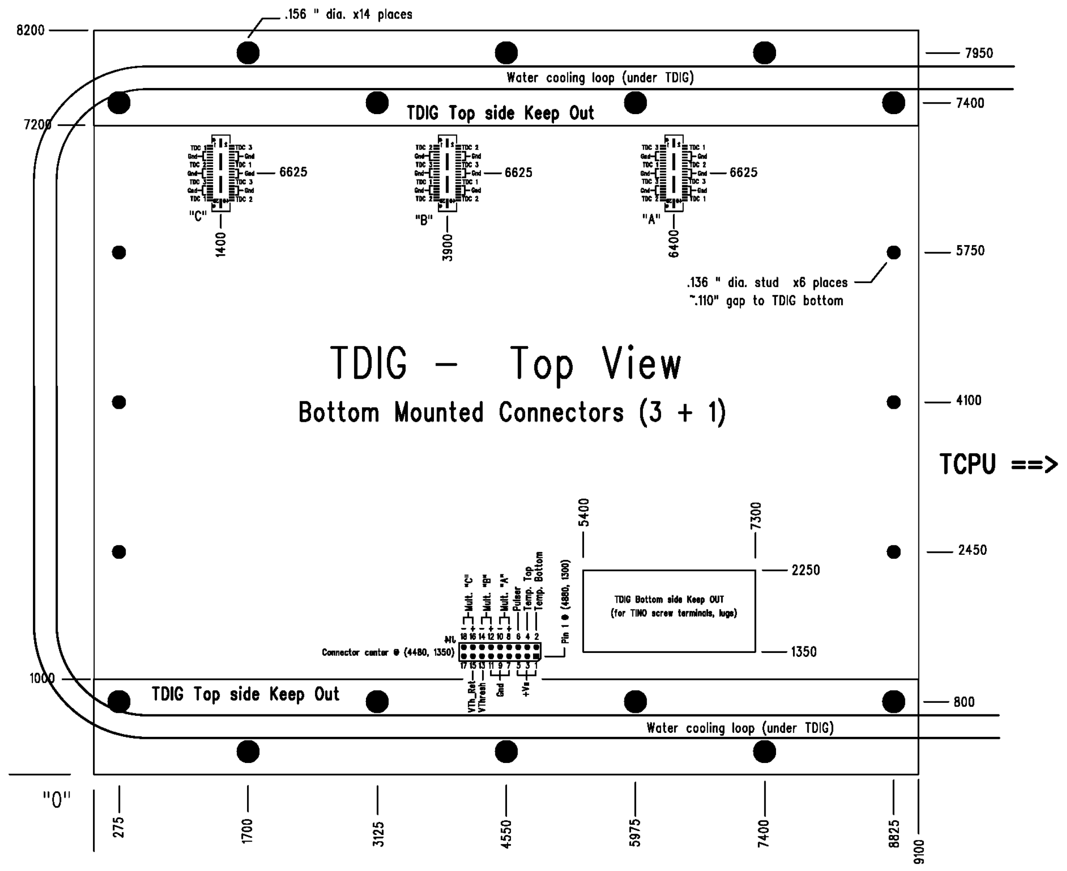
J17: Aux data path token routing – jumper 2-3 if board is farthest from TCPU; otherwise jumper 1-2.

SW4: board position switch. Selects board position in tray, position 0 to 7.

9. Physical Parameters

9.1. TDIG-to-TINO PCB Layout

The diagram below describes critical layout locations (whole number locations without decimals are "mils" 0.001 inches):


.156 " dia. x14 places
8200
7950
7400
7200
Water cooling loop (under TDIG)
TDIG Top side Keep Out
6625
"C"
"B"
"A"
1400
3900
6400
5750
.136 " dia. stud x6 places
~.110" gap to TDIG bottom
TDIG - Top View
Bottom Mounted Connectors (3 + 1)
4100
TCPU ==>
5400
7300
2450
2250
TDIG Bottom side Keep OUT (for TINO screw terminals, lugs)
1350
Connector center @ (4480, 1350)
Pin 1 @ (4880, 1300)
1000
800
"0"
275
1700
3125
4550
5975
7400
8825
9100

9.2. PCB Layer Stackup and Specifications

Overall board size: 8.2 x 9.5 inches (Connectors J4, J5 overhang edge of board)
Board material: FR4 nominal thickness 0.093 inches 8 conductive layers.
Surface finish: Immersion Silver.
Layer Stackup (Primary-to-Secondary):

Conductive Layer		Laminate Layer
0.5 oz Copper primary	Over	3.45 mils FR-406
1 oz Copper	Over	6.0 mils FR-406
1 oz Copper	Over	7.55 mils FR-406
1 oz Copper	Over	47 mils FR-406
1 oz Copper	Over	7.55 mils FR-406
1 oz Copper	Over	6.0 mils FR-406
1 oz Copper	Over	3.45 mils FR-406
0.5 oz Copper secondary		

10. Change History

Version 4_0a: Modifications of registers for BNL boards added (JS)
Version 4_x: Extensive restructuring and revision for TDIG-F.
Version 3_2:
    Added Board and TDC numbering
Version 3_1: October xx, 2007
    Appendix B: TINO interface specification
Version 3_0: October 1, 2007
    Formatting changes
    Settings for Aux data path readout, including J17 settings and TDC configuration
    J14 description (TDC clock monitor connector)
    LED indicator description
    Appendix A: TDIG FPGA configuration registers

Appendix A: TDIG FPGA configuration registers

"TDIG FPGA – MCU IF registers ver 1.xls" Definition of R/W registers in FPGA that provide configuration / status communication with MCU.

MCU writes TO byte Address	**Register name**	**Bit Address** (7 is msb)	**Description** (Active High unless indicated)	**Notes**
0	**CONFIG_0**		**Configuration 0**	
		0	Select test input to MCU FIFO	DEFAULT = 0 (Not in BNL boards)
		1	Select this board as first board in readout chain. Token to first TDC comes from upstream (or MCU in test mode) rather than from downstream TDIG.	DEFAULT = 0 (Not in BNL boards)
		2	Select test mode for serial readout: FPGA uses Strobe 0 as signal to issue token and do serial readout from TDCs	DEFAULT = 0 (Not in BNL boards)
		3	Select test mode for TDC data: FPGA uses STROBE_1 as signal to issue one pulse on Channel 1 to each TDC.	DEFAULT = 0 (Not in BNL boards)
		4	Select test mode for TDC trigger: FPGA uses STROBE_2 as signal to issue one trigger pulse to each TDC.	DEFAULT = 0 (Not in BNL boards)
		5	Select test mode for Bunch Reset	DEFAULT = 0 (bunch reset from upstream) (Not in BNL boards)
		6	Select test mode for Event Reset	DEFAULT = 0 (event reset from upstream) (Not in BNL boards)
		7		
1	**CONFIG_1**		**Configuration 1**	

MCU writes TO byte Address	**Register name**	**Bit Address** (7 is msb)	**Description** (Active High unless indicated)	**Notes**
		1:0	Select TDC for JTAG communication (msb) "JTAG_MODE" msb =JU2 / 5,6 lsb = JU2 / 7,8 msb | lsb 00 : no TDC selected 01 : TDC1 (U2) selected 10 : TDC2 (U3) selected 11 : TDC3 (U4) selected	For user communication w/ TDCs using the ByteBlaster connected to the Test Header, MCU reads jumper settings to select TDC. Default board configuration should have these jumpers open (unstuffed), so that MCU will write 00.
		2	Select MCU as source for TDC configuration. "JTAG_SEL" If not selected, then JTAG input from test header is selected.	DEFAULT = 0
		4:3	Not currently implemented. TDC readout selection: table index = bits 765 000 : 3 TDCs 001 : TDC U2 010 : TDC U3 100 : TDC U4	DEFAULT = 000
		5		
		6		
		7		
2	**CONFIG_2**		**Configuration 2**	
		0	TDC hardware reset	Active HI; DEFAULT = 0
		1	Bunch reset test mode	0 = shift register source, 1 = MCU strobe 7, default = 0 (Not in BNL boards)
		2		
		3		
		4		
		5		
		6		
		7		
3	**CONFIG_3**			

MCU writes TO byte Address	Register name	Bit Address (7 is msb)	Description (Active High unless indicated)	Notes
4	STROBE_4	N/A	**Generate test token.** Upon receipt of this signal, the serial readout controller will issue token and do serial readout from TDCs. Enabled by CONFIG_0.2.	Value "written" does not matter. (Not in BNL boards)
5	STROBE_5	N/A	**Generate test data.** FPGA will generate 1 test pulse for each TDC for each STROBE_1 input. Enabled by CONFIG_0.3.	Value "written" does not matter. (Not in BNL boards)
6	STROBE_6	N/A	**Generate test trigger.** FPGA will generate 1 trigger pulse for each TDC for each STROBE_2 input. Enabled by CONFIG_0.4.	Value "written" does not matter. (Not in BNL boards)
7	STROBE_7	N/A	**Generate test bunch reset.** FPGA will generate 1 bunch reset pulse for each TDC for each STROBE_3 input. Enabled by CONFIG_0.5	Value "written" does not matter.
8	STROBE_8	N/A	**Generate test event reset.** FPGA will generate 1 event reset pulse for each TDC for each STROBE_3 input. Enabled by CONFIG_0.5	Value "written" does not matter.
9	STROBE_9	N/A	**Reset readout.** Resets local serial readout state machine.	Value "written" does not matter.
10	STROBE_10	N/A	**Reset MCU FIFO**	Value "written" does not matter.
11	STROBE_11	N/A	Clocks test data counter and MCU fifo input	Value "written" does not matter.
12	CONFIG12	2:00	Position switch (2:0) right justified i.e. 3 lsbs	
13	CONFIG13	3:0	Multiplicity gate width	(1/16th of a RHICstrobe)
14	CONFIG14	0	SELECT LVDS TEST	
15				

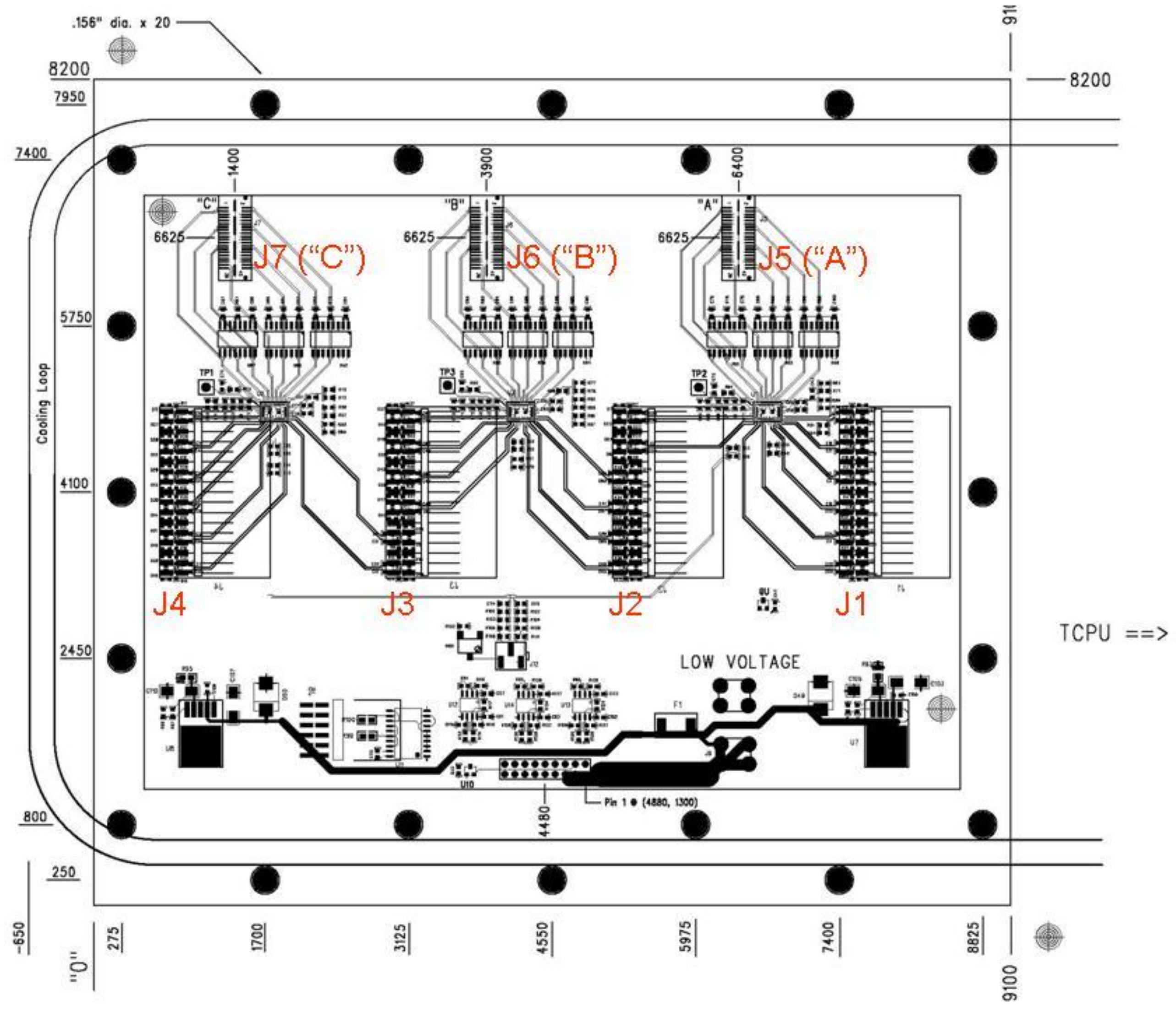
MCU reads FROM byte Address	**Register name**	**Bit Address** (7 is msb)	**Description** (Active High unless indicated)	**Notes**
0	**CONFIG_0**		**Configuration 0 readback**	
1	**CONFIG_1**		**Configuration 1 readback**	
2	**CONFIG_2**		**Configuration 2 readback**	
3	**CONFIG_3**		**Configuration 3 readback**	
		0	TDC 1 Error Bit	
		1	TDC 2 Error Bit	
		2	TDC 3 Error Bit	
4				
5				
6				
7	**Version ID**		**FPGA code version ID**	Constant value identifying FPGA code version
8	**Status 0**			
		0		
		1		
		2		
		3		
		4		
		5		
		6		
		7		
9	**Status 1**			
		0		
		1		
		2		
		3		
		4		
		5		
		6		
		7		
10				
11	**FIFO (7:0)**		**LS byte**	
12	**FIFO (15:8)**			
13	**FIFO (23:16)**			
14	**FIFO (31: 24)**		**MS byte** Reads should be from LSbyte (read first) to MSbyte (read last). Final read from MS byte generates read_clock_enable	

MCU reads FROM byte Address	Register name	Bit Address (7 is msb)	Description (Active High unless indicated)	Notes
			signal to FIFO.	
15	**FIFO status**			
		4:0	MCU FIFO OCCUPANCY (# of data words in FIFO)	
		5	MCU FIFO PARITY	
		6	MCU FIFO FULL	
		7	MCU FIFO EMPTY	

Appendix B: “TINO_N” – “TDIG-E” Mapping

Version: 6, February 14, 2007

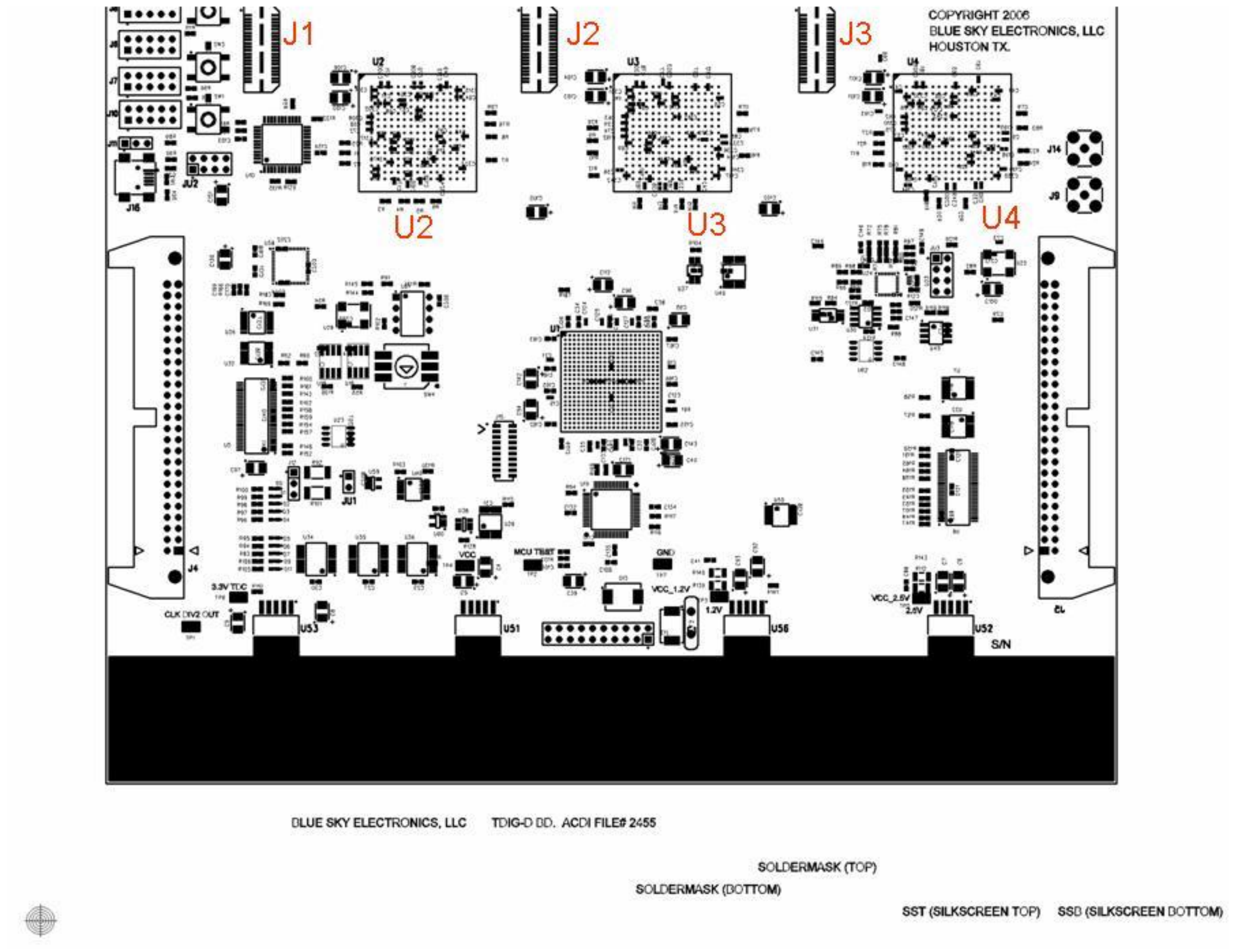
This document is based on the alternative mapping suggested by G. Eppley, which was finally implemented on TDIG_E.
TINO_N has four connectors on the bottom of the board (J1 – J4) that connect to four MRPCs. On the top of the board are three signal connectors that mate with corresponding connectors on TDIG plus one connector for the power. The following picture shows the layout of these connectors as seen when looking at TINO_N from the top.



The MRPC connectors on the bottom of TINO_N are labeled “J1”, “J2”, “J3”, and “J4”. “J4” is the connector closest to eta = 0 (away from TCPU), “J1” is the connector closest to eta = 1 (towards TCPU).

The signal connectors are labeled J5 (“A”), J6 (“B”), and J7 (“C”). J7 (“C”) is the connector closest to eta = 0 (away from TCPU), while J5 (“A”) is the connector closest to eta = 1 (towards TCPU).

TDIG-E has 3 signal connectors on the bottom side of the board (J1, J2, J3) corresponding to the signal connectors on the top side of TINO_N, and a power connector on the bottom of the board (J13) corresponding to the power connector on top of TINO_N. The layout of these connectors as seen when looking at TDIG-E from the top is shown in the following picture.


J1
J2
J3
COPYRIGHT 2006
BLUE SKY ELECTRONICS, LLC
HOUSTON TX.
U2
U3
U4
BLUE SKY ELECTRONICS, LLC   TDIG-D BD. ACDI FILE# 2455
SOLDERMASK (TOP)
SOLDERMASK (BOTTOM)
SST (SILKSCREEN TOP)   SSB (SILKSCREEN BOTTOM)

The signal connectors on TDIG-E are labeled “J1”, “J2”, and “J3”. “J1” is the connector closest to eta = 0 (away from TCPU), while “J3” is the connector closest to eta = 1 (towards TCPU). “J1” on TDIG-E mates with J7 (“C”) on TINO_N, “J2” on TDIG-E with J6 (“B”) on TINO_N and “J3” on TDIG-E with J5 (“A”) on TINO_N. TDIG-E has three HPTDC chips on top, labeled “U2”, U3”, and “U4”. “U2” is the chip closest to eta = 0, while “U4” is the chip closest to eta = 1.

The signal path is then intended to be as follows:

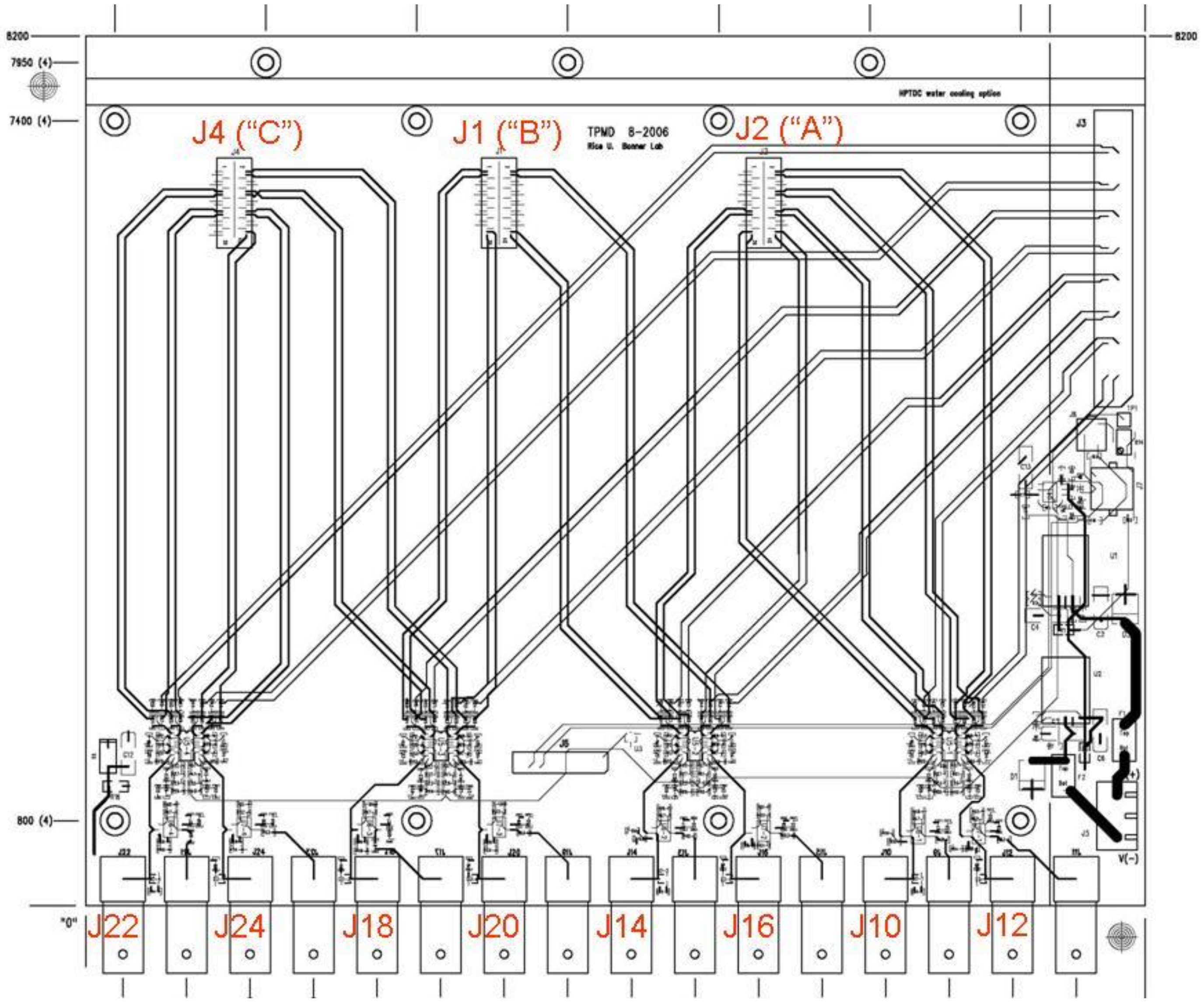
Pad #	PAD #	TINO Bottom connector	Pins	TINO top connector	Pins	TDIG Connector	Pins	TDC	Ch
MRPC 1	1	J4	1,2,3,4	J7 ("C")	1,3	J1	1,3	U2	7
	2	J4	7,8,9,10	J7 ("C")	13,15	J1	13,15	U3	7
	3	J4	13,14,15,16	J7 ("C")	25,27	J1	25,27	U2	0
	4	J4	19,20,21,22	J7 ("C")	37,39	J1	37,39	U3	2
	5	J4	25,26,27,28	J7 ("C")	38,40	J1	38,40	U2	5
	6	J4	31,32,33,34	J7 ("C")	26,28	J1	26,28	U3	6
MRPC 2	1	J3	1,2,3,4	J6 ("B")	1,3	J2	1,3	U4	7
	2	J3	7,8,9,10	J6 ("B")	13,15	J2	13,15	U2	4
	3	J3	13,14,15,16	J6 ("B")	25,27	J2	25,27	U4	4
	4	J3	19,20,21,22	J6 ("B")	37,39	J2	37,39	U2	2
	5	J3	25,26,27,28	J7 ("C")	2,4	J1	2,4	U3	3
	6	J3	31,32,33,34	J7 ("C")	14,16	J1	14,16	U2	6
MRPC 3	1	J2	1,2,3,4	J5 ("A")	1,3	J3	1,3	U3	0
	2	J2	7,8,9,10	J5 ("A")	13,15	J3	13,15	U4	2
	3	J2	13,14,15,16	J6 ("B")	2,4	J2	2,4	U2	3
	4	J2	19,20,21,22	J6 ("B")	14,16	J2	14,16	U4	3
	5	J2	25,26,27,28	J6 ("B")	26,28	J2	26,28	U2	1
	6	J2	31,32,33,34	J6 ("B")	38,40	J2	38,40	U4	6
MRPC 4	1	J1	1,2,3,4	J5 ("A")	2,4	J3	2,4	U4	0
	2	J1	7,8,9,10	J5 ("A")	14,16	J3	14,16	U3	5
	3	J1	13,14,15,16	J5 ("A")	26,28	J3	26,28	U4	1
	4	J1	19,20,21,22	J5 ("A")	38,40	J3	38,40	U3	4
	5	J1	25,26,27,28	J5 ("A")	37,39	J3	37,39	U4	5
	6	J1	31,32,33,34	J5 ("A")	25,27	J3	25,27	U3	1

This results in the following connections between the signal connectors on TDIG-E and the HPTDC’s on TDIG-E:

TDIG Connector	TDC	Ch	TINO Top connector	Pins	TINO Bottom connector	Pins	MRPC	PAD #
J1	U2	7	J7 ("C")	1,3	J4	1,2,3,4	MRPC 1	1
J1	U3	3	J7 ("C")	2,4	J3	25,26,27,28	MRPC 2	5
J1	U3	7	J7 ("C")	13,15	J4	7,8,9,10	MRPC 1	2
J1	U2	6	J7 ("C")	14,16	J3	31,32,33,34	MRPC 2	6
J1	U2	0	J7 ("C")	25,27	J4	13,14,15,16	MRPC 1	3
J1	U3	6	J7 ("C")	26,28	J4	31,32,33,34	MRPC 1	6
J1	U3	2	J7 ("C")	37,39	J4	19,20,21,22	MRPC 1	4
J1	U2	5	J7 ("C")	38,40	J4	25,26,27,28	MRPC 1	5
J2	U4	7	J6 ("B")	1,3	J3	1,2,3,4	MRPC 2	1
J2	U2	3	J6 ("B")	2,4	J2	13,14,15,16	MRPC 3	3
J2	U2	4	J6 ("B")	13,15	J3	7,8,9,10	MRPC 2	2
J2	U4	3	J6 ("B")	14,16	J2	19,20,21,22	MRPC 3	4
J2	U4	4	J6 ("B")	25,27	J3	13,14,15,16	MRPC 2	3
J2	U2	1	J6 ("B")	26,28	J2	25,26,27,28	MRPC 3	5
J2	U2	2	J6 ("B")	37,39	J3	19,20,21,22	MRPC 2	4
J2	U4	6	J6 ("B")	38,40	J2	31,32,33,34	MRPC 3	6
J3	U3	0	J5 ("A")	1,3	J2	1,2,3,4	MRPC 3	1
J3	U4	0	J5 ("A")	2,4	J1	1,2,3,4	MRPC 4	1
J3	U4	2	J5 ("A")	13,15	J2	7,8,9,10	MRPC 3	2
J3	U3	5	J5 ("A")	14,16	J1	7,8,9,10	MRPC 4	2
J3	U3	1	J5 ("A")	25,27	J1	31,32,33,34	MRPC 4	6
J3	U4	1	J5 ("A")	26,28	J1	13,14,15,16	MRPC 4	3
J3	U4	5	J5 ("A")	37,39	J1	25,26,27,28	MRPC 4	5
J3	U3	4	J5 ("A")	38,40	J1	19,20,21,22	MRPC 4	4

Appendix C: TPMD Mapping

TPMD has 16 BNC connectors mounted on top and on the bottom of the board. The top mounted BNC connectors are labeled “J10”, “J12”, “J14”, “J16”, “J18”, “J20”, “J22”, and “J24”. There are three signal connectors on top of TPMD labeled “J2” (“A”), “J1” (“B”), and “J4” (“C”). The layout of these connectors as seen when looking at TPMD from the top is shown in the following picture:



The resulting mapping from the BNC connectors on TPMD to the TDIG connectors and TDIG’s HPTDC chips is shown in the following table. There are a total of 8 leading edge mappings (“L.E.”) and 8 trailing edge mappings (“T.E.”) for each input BNC connector. The leading edge signals are mapped to two HPTDC with 5 channels on U2 and 3 channels on U4, while the trailing edge signals are mapped to one HPTDC (U3) with 8 channels:

BNC in	TPMD top connector	Pins	TDIG connector	L.E. TDC	Ch	TPM top connector	Pins	TDIG connector	T.E. TDC	Ch
J10	J2, "A"	26,28	J3	U4	1	J2, "A"	37,39	J3	U4	5
J12	J2, "A"	2,4	J3	U4	0	J2, "A"	14,16	J3	U3	5
J14	J1, "B"	38,40	J2	U4	6	J2, "A"	25,27	J3	U3	1
J16	J1, "B"	2,4	J2	U2	3	J2, "A"	38,40	J3	U3	4
J18	J4, "C"	14,16	J1	U2	6	J1, "B"	1,3	J2	U4	7
J20	J1, "B"	37,39	J2	U2	2	J4, "C"	2,4	J1	U3	3
J22	J4, "C"	25,27	J1	U2	0	J4, "C"	13,15	J1	U3	7
J24	J4, "C"	38,40	J1	U2	5	J4, "C"	26,28	J1	U3	6