

Strip Telescope Interface Board Firmware

1 Organization

The STIB firmware is organized into several independent modules that perform specific operations but which are controlled and monitored by means of common bus. The bus architecture used is the MicroBlaze soft core external memory interface, which serves as the bus master, providing a 32-bit address space that is mapped into separate regions for each slave component.

2 IOBUS interface

The IOBUS interface is defined in a package that provides a configurable number of slave interfaces that connect to the MicroBlaze soft core, which serves as the bus master. The intent is that the core will configure and monitor the slave interfaces, but will not be responsible for any large volume or fast data transfers between slaves or between slaves and external interfaces. Instead, fast data transfers are handled by the slaves directly, but these can indicate to the core that data is available in internal buffers by means of an interrupt signal.

Each slave interface maintains its own separate address space and acknowledges address reads that lie within this range by asserting its SLAVE_READY signal. If no slaves assert the acknowledge signal, then the IOBUS interface itself asserts the acknowledge signal to the core and drives the pattern 0xf8f8f8 on the data bus.

3 Clock interface

The 66.667 MHz oscillator is scaled to provide a signal with a frequency of 1 Hz which increments a 32-bit counter which can be loaded by the bus master to correspond to the number of seconds since midnight on January 1, 1970.

3.1 Memory map and registers

The clock module provides a command/status register and the register containing the time in seconds. At this time, the functions of the bits in the CSR are not assigned.

address	name	description	access
0xc2000000	CLOCK_CSR	command/status register	r/w
0xc2000004	CLOCK_TIME	seconds counter	r/w

Table 1: Clock module memory map.

4 Gigabit Ethernet interface

Signals from the GbE physical interface drive several state machines that parse the received data stream and distribute data payloads to the appropriate modules that are responsible for responding to specific ethernet and IP protocols. Likewise, several state machines that drive serial data on the physical interface are responsible for formatting ethernet and IP headers and transmitting data payloads. The heirarcy of the components that comprise the GbE module are indicated below.

<code>gbe_interface</code>	IOBUS interface to GbE module
<code>ethernet</code>	receives ethernet frames
<code>arp</code>	processes ARP packets
<code>ip</code>	processes IP packets
<code>ipheader</code>	parses IP header
<code>icmp</code>	parses ICMP packets
<code>udp</code>	parses UDP packets
<code>txether</code>	transmits ethernet frames
<code>arprep</code>	transmits ARP reply message
<code>arpreq</code>	transmits ARP request message
<code>sendpkt</code>	transmits UDP packet
<code>arptab</code>	maintains list of ARP table entries

Table 2 describes the registers associated with the Gigabit Ethernet interface.

address	name	description	access
0xc1000000	GBE_CSR	command/status register	r/w
0xc1000004	FIFO_STATUS	spy FIFO status word	r
0xc1000010	RX_FIFO_DATA	receive FIFO data	r
0xc1000014	TX_FIFO_DATA	transmit FIFO data	r
0xc1000018	-	-	r
0xc1000020	IP_ADDR	interface IP address	r/w
0xc1000024	NETMASK	interface netmask	r/w
0xc1000028	GATEWAY	address of router to external networks	r/w
0xc100002c	IP_DADDR	destination IP address	r/w
0xc1000034	UDP_FIFO_DATA	UDP packet payload FIFO	r
0xc1000038	UDP_PORTS	UDP packet source and destination ports	r
0xc100003c	UDP_LENGTH	UDP packet length	r
0xc1000040	UDP_SADDR	UDP packet source address	r
0xc1000044	ARP_CSR	ARP request/reply command status register	r/w
0xc1000048	ARP_LOOKUP_IP	IP address to look up in ARP table	r/w
0xc100004c	ARP_TABLE_IP	actual IP address looked up in ARP table	r
0xc1000050	ARP_TABLE_HA_LOW	low 32 bits of retrieved hardware address	r
0xc1000054	ARP_TABLE_HA_HIGH	high 16 bits of retrieved hardware address	r
0xc1000058	ARP_TABLE_REFCOUNT	reference count for specified IP address in ARP table	r
0xc1000060	SENDPKT_CSR	UDP transmit packet command/status register	r/w
0xc1000064	SENDPKT_DADDR	UDP transmit packet destination address	r/w
0xc1000068	SENDPKT_PORTS	source/destination ports for UDP transmit packet	r/w
0xc1000100	RX_FRAME_COUNTER	received frames	r/w
0xc1000104	RX_ERROR_COUNTER	received frames with RXERR signal asserted	r/w
0xc1000108	RX_VALID_COUNTER	number of received frames with valid CRC	r/w
0xc100010c	RX_ARPREQ_COUNTER	number of ARP requests received	r/w
0xc1000110	RX_ARPREP_COUNTER	number of ARP replies received	r/w
0xc1000114	RX_IP_COUNTER	number of IP packets received	r/w
0xc1000118	RX_IP_BADCKSUM_COUNTER	number of IP packets with bad checksums	r/w
0xc100011c	RX_ARPREQ_ADDR_COUNTER	number of ARP requests matching hardware address	r/w
0xc1000120	RX_ICMPREQ_COUNTER	number of ICMP request packets received	r/w
0xc1000124	RX_ICMPREP_COUNTER	number of ICMP reply packets received	r/w
0xc1000128	TX_ICMPREP_COUNTER	number if ICMP reply packets transmitted	r/w
0xc100012c	RX_UDP_COUNTER	number of UDP packets received	r/w

Table 2: Gigabit Ethernet interface registers.

5 Strip interface

The interface to the strip sensor hybrids consists of a set of LVDS pads that drive and receive the following signals on each connector:

- MCLK_A and MCLK_B
- BCOCLK
- SHIFT, SCIN, SCOUT
- RESET
- HITOR
- OUT1[4..0], OUT2[4..0]

The clock signals are generated by a digital clock module (DCM) in the `stripclk` component, the reset signal is generated in the `stripreset` component and the slow controls signals are handled by the `stripsc` component. Serial data from a single chip are deserialized by the `chipserdes` component, which also provides status and error checking information. The STRIPANA module provides a means to accumulate statistics from the data received on a specified strip number.

5.1 Clock signals

The `stripclk` module generates the clock signals needed by the FSSR2 chips and throughout the design. The input signals are from the 66.667 MHz oscillator, the 5 MHz oscillator, and an external clock signal. The 5 MHz signal is used for the SPI interfaces that control the DAC's. The 66.667 MHz clock is used to derive the MCLK_A and MCLK_B signals as well as an internal 133.333 MHz clock that is phase aligned to MCLK_A for the readout of the DDR data interface from the FSSR2 chips. Finally, the 66.667 MHz clock is also used to synthesize a 200 MHz clock signal that drives the DAC's for charge injection.

The BCOCLK can also be driven from the 66.667 MHz input signal, but can also be driven from an external clock input. Setting bit 16 of the STRIP_CSR register selects the external clock source. The selected input drives a Virtex-4 DCM_ADV component with programmable multiplication and division factors that synthesizes the BCOCLK signal. The DCM_ADV component actually synthesizes a clock signal that has 8 times the frequency of the desired BCOCLK signal in order to provide 8 fine time bins within each BCOCLK period to record the times at which trigger signals are received. If it is not possible to generate a clock signal with this frequency, bit 17 in the STRIP_CSR register can be set, which generates a BCOCLK signal that is only 4 times the synthesized frequency, and provides only 4 bins for recording trigger timing information.

Table 3 lists the parameters that can be used to generate BCO clocks with various frequencies.

N	D	internal	external
		66.667 MHz	40.000 MHz
6	5	10.0	6.0
3	2	12.5	7.5
12	5	20.0	12.0
3	1	25.0	15.0
13	4	27.083	16.25

Table 3: Numerator and denominator values for synthesis of various BCO clock frequencies. Note that the FSSR2 chips may not operate correctly at BCO frequencies above about 20 MHz.

5.2 Memory map and registers

Table 4 lists the memory map for the strip sensor interface. Each readout chip has its own address space containing a spy FIFO, a link status register and error counters. These are indexed by sensor number 0..7 and chip number 1..5 as indicated in Table 4.

address	name	description	access
0xc4000000	STRIP_CSR	Command/status register	r/w
0xc4000004	STRIP_RESET	Chip reset	r/w
0xc4000008	STRIP_SC	Slow controls command register	r/w
0xc4000010	STRIP_SCI0	Slow controls input word 0	r/w
0xc4000014	STRIP_SCI1	Slow controls input word 1	r/w
0xc4000018	STRIP_SCI2	Slow controls input word 2	r/w
0xc400001c	STRIP_SCI3	Slow controls input word 3	r/w
0xc4000020	STRIP_SCO0	Slow controls output word 0	r/w
0xc4000024	STRIP_SCO1	Slow controls output word 1	r/w
0xc4000028	STRIP_SCO2	Slow controls output word 2	r/w
0xc400002c	STRIP_SCO3	Slow controls output word 3	r/w
0xc4000030	STRIP_FIFO_CSR	FIFO status register	r/w
0xc4000034	STRIP_FIFO_DATA	FIFO data	r
0xc4000038	STRIP_ANALYSIS_CSR	Strip analysis control	r/w
0xc400003c	STRIP_ANALYSIS_BCO_COUNTER	Analysis BCO counter	r
0xc4000y3c	STRIP_ANALYSIS_HIT _y _COUNTER	Analysis chip <i>y</i> hit counter	r
0xc400xy30	STRIP_STATUS	Strip status register	r
0xc4000040	STRIP_BCO_COUNTER_LOW	BCO counter (low 32 bits)	r/w
0xc4000044	STRIP_BCO_COUNTER_HIGH	BCO counter (upper 16 bits)	r/w
0xc4000048	STRIP_BCO_DCM	Digital Clock Manager configuration	r/w
0xc400004c	STRIP_STREAMER	Streamed data status	r
0xc4000050	STRIP_DAC_CSR	DAC constrol/status register	r/w
0xc4000054	STRIP_DAC_VALUE	DAC pulse amplitude	r/w
0xc4000058	STRIP_DAC_SPI	DAC SPI interface	r/w
0xc400005c	STRIP_WORDS_DUMPED	Number of words dropped	r
0xc4000060	STRIP_TRIG_CSR	Trigger control/status register	r/w
0xc4000064	STRIP_TRIG_FIFO_DATA	spy FIFO for trigger data words	r
0xc4000068	STRIP_TRIG_UNBIASED	Control register for periodic zero bias trigger	r/w
0xc4000070	STRIP_TRIG_INPUT_0	Controls trigger inputs 0,1	r/w
0xc4000074	STRIP_TRIG_INPUT_1	Controls trigger inputs 2,3	r/w
0xc4000078	STRIP_TRIG_INPUT_2	Controls trigger inputs 4,5	r/w
0xc400007c	STRIP_TRIG_INPUT_3	Controls trigger inputs 6,7	r/w

Table 4: Strip interface memory map.

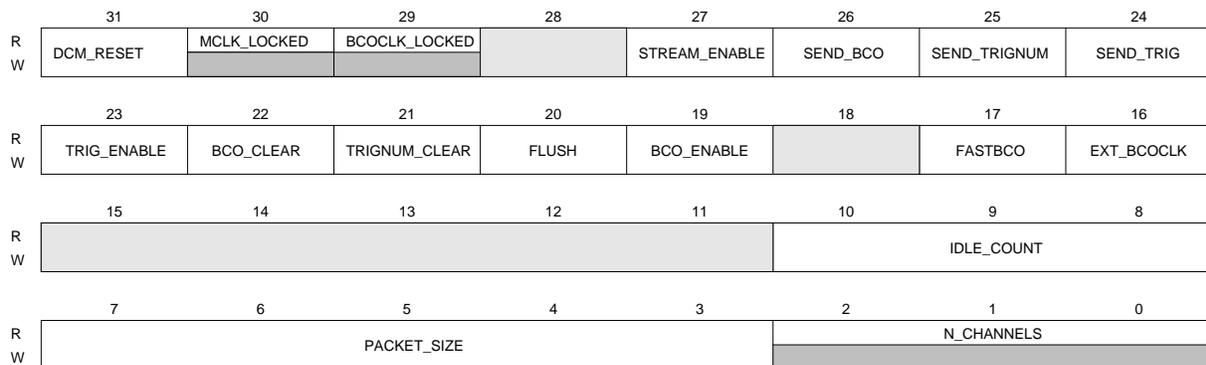
bits	name	access	description
31	DCM_RESET	r/w	Assert RESET signal to the clock generator module
30	MCLK_LOCKED	r	MCLK frequency locked
29	BCO_LOCKED	r	BCO frequency locked
28	—		unused
27	STREAM_ENABLE	r/w	enables streaming hit data over Ethernet
26	SEND_BCO	r/w	Include BCO number in hit data stream
25	SEND_TRIGNUM	r/w	Include trigger number in hit data stream
24	SEND_TRIG	r/w	Include trigger data in hit data stream
23	TRIG_ENABLE	r/w	Require trigger in order to read out data
22	BCO_CLEAR	r/w	resets BCO counter to zero
21	TRIGNUM_CLEAR	r/w	resets trigger counter to zero
20	FLUSH	r/w	flushes partially filled buffers
19	BCO_ENABLE	r/w	enables BCO counter increment
17	FASTBCO	r/w	doubles maximum possible BCO frequency
16	EXT_BCOCLK	r/w	selects external BCO clock source
15–11	—		unused
10–8	IDLE_COUNT	r/w	Time to wait before dumping idle packet
7–3	PACKET_SIZE	r/w	Number of 32-bit hit words per packet
2–0	N_CHANNELS	r	Number of channels defined in firmware

Table 5: STRIP_CSR - address 0xc4000000.

5.2.1 Strip command/status register

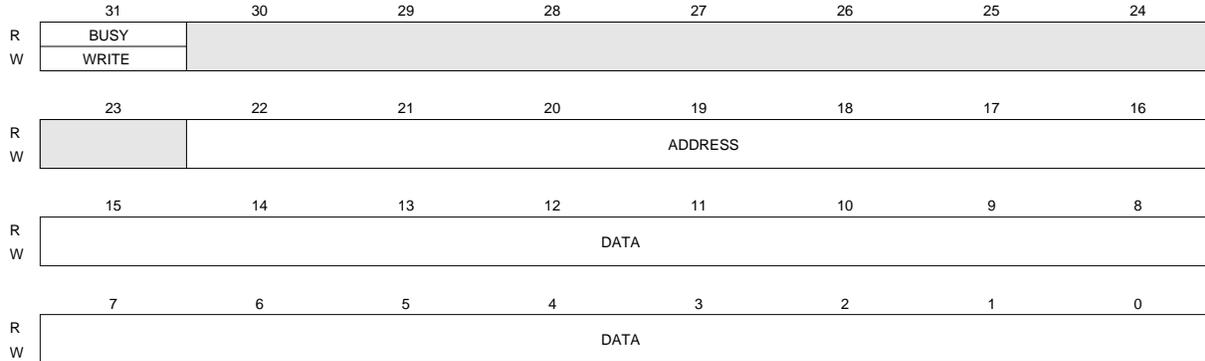
The main strip CSR controls the configuration of clocks and the flow of data sent via the GbE or optical serial links.

STRIP_CSR
Base address: 0xc400
Offset: 0x0000



5.2.2 Digital Clock Manager configuration register

STRIP_BCO_DCM
 Base address: 0xc400
 Offset: 0x0048



The DCM configuration register (STRIP_BCO_DCM) is used to control the DCM that synthesizes both the BCO clock signal and a faster clock that is $4\times$ or $8\times$ the BCO clock frequency. The source of this clock can either be the 66.667 MHz internal oscillator, or an external clock source. Bit 16 of the STRIP_CSR register selects between these two clock sources and setting bit 17 specifies that the faster clock will be only $4\times$ the BCO clock frequency. When writing to this register, if the high bit is set, the 16-bit data is written to the register specified in the address field. If the high bit is clear, then the data contained in the addressed register is returned in the low 16-bit data field on a subsequent read. The high bit will read back as being set if the data has not yet been retrieved from the DCM component.

The only two registers that are relevant for changing the clock frequency are at addresses 0x50 and 0x52, and specify the numerator and the denominator, respectively, of the frequency multiplication factor. If the numerator and denominator are N and D , then the 5-bit data field is written $N - 1$ or $D - 1$. These specify the frequency of the fast clock from which the BCO clock is derived after dividing by 4 or 8 depending on the setting of bit 17 in the STRIP_CSR register.

For example, to set a slow BCO clock of 12.5 MHz, derived from the internal clock source, write 5 to register 0x50 and 3 to register 0x52 so that $(66.667 \text{ MHz}) \times 6/4/8 = 12.50 \text{ MHz}$.

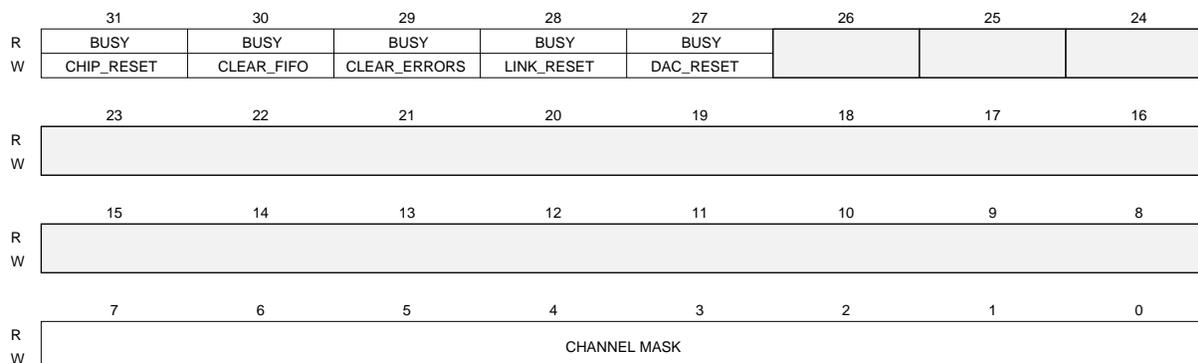
bits	name	access	description
31	CHIP_RESET	w r	Assert RESET signal on selected channels Indicates that RESET signal is asserted
30	CLEAR_FIFO	w r	Clears link spy FIFO's Istatus of FIFO clear signal
29	CLEAR_ERRORS	w r	Clears link error counters Status of link error counter reset signal
28	LINK_RESET	w r	Causes the selected links to realign and synchronize Status of link reset signal
27	DAC_RESET	w r	Asserts the reset* signal to the DAC's (not masked) Status of the DAC reset signal
7..0	CHANNEL_MASK	r/w	Reset signals are sent to all channels with bits set in the CHANNEL_MASK field

Table 6: STRIP_RESET - address 0xc400004.

5.2.3 Strip reset register

Reset signals are asserted by setting the appropriate bits in the STRIP_RESET register. These cause the specified reset signal to be asserted for a fixed number of BCO clock cycles (currently 7), and are self clearing. While the reset signals are asserted, their status is reflected by reading the appropriate bits in the register. The CHANNEL_MASK bits select the channels to which the reset signals are simultaneously sent. Note that asserting CHIP_RESET does not automatically assert LINK_RESET. Normally CHIP_RESET and LINK_RESET should be asserted simultaneously to cause the link to be resynchronized after the chip comes out of reset.

STRIP_RESET
Base address: 0xc400
Offset: 0x0004



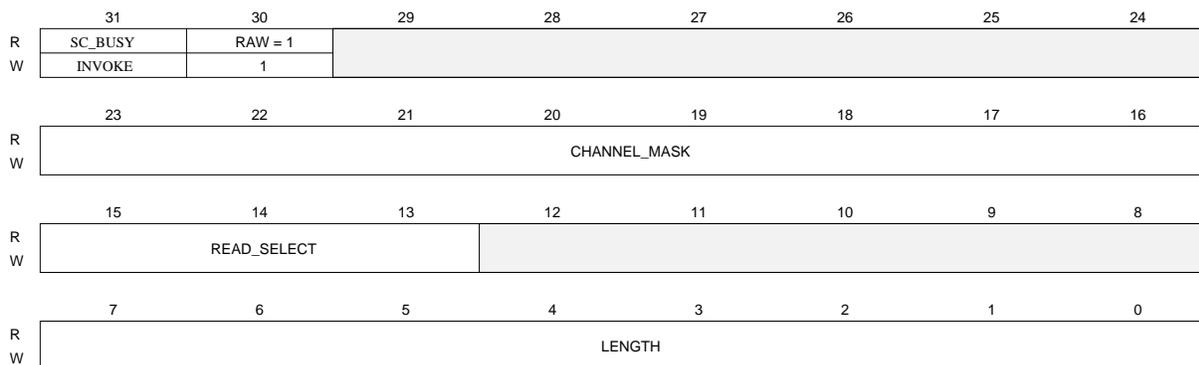
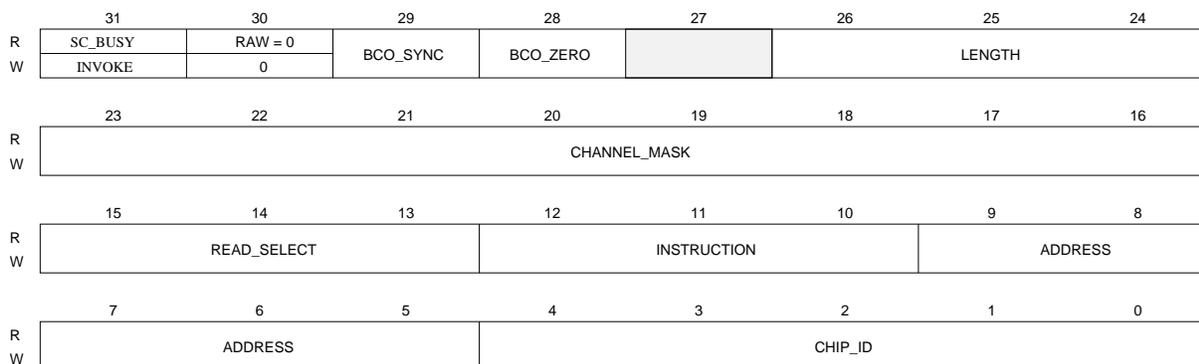
value	number of bits
0	0
1	1
2	2
3	8
4	128

Table 7: Encoded length parameter values in the STRIP_SC register.

5.2.4 Slow controls command register

There are two modes of operation for the STRIP_SC register which depend on the value of the RAW bit (bit 30). When RAW=0, the different bit fields specify the slow controls register operation to be performed. When RAW=1, bits contained in the STRIP_SCI[0..3] registers are shifted out without interpretation. The slow controls bits are shifted on the SHIFT/SCIN lines of all channels selected by the CHANNEL_MASK field and the SCOUT data received from the single channel selected by the READ_CHANNEL is stored in the STRIP_SCO[0..3] registers. When RAW=0, the LENGTH field encodes the number of data bits to be sent or received according to Table 7.

STRIP_SC
Base address: 0xc400
Offset: 0x0008



When the BCO_SYNC and BCO_ZERO bits are not set, SHIFT is asserted on the next falling edge of BCOCLK and is deasserted on the falling edge of BCOCLK after all bits have been shifted.

bits	name	access	description
31	SC_BUSY	r	Set while a slow controls transaction is in progress
	INVOKE	w	Initiate the slow controls transaction
30	RAW	r/w	Set when raw data is being shifted out
29	BCO_SYNC	r/w	Synchronize rising edge of SHIFT with BCO zero
28	BCO_ZERO	r/w	Synchronize falling edge of SHIFT with BCO zero
26-24	LENGTH	r/w	Encodes the number of bits to send
23-16	CHANNEL_MASK	r/w	Set bits enable the channels to receive SCIN
15-13	READ_SELECT	r/w	Select channel from which to read SCOUT
12-10	INSTRUCTION	r/w	Slow-controls instruction
9-5	ADDRESS	r/w	Slow-controls register address
4-0	CHIP_ID	r/w	Target chip ID

Table 8: STRIP_SC - address 0xc4000008.

When BCO_SYNC is set, the transaction is delayed until the internal BCO counter has a value of 240 and SHIFT is deasserted on the first rising edge of BCOCLK after all bits have been shifted. This allows the AqBCO command to be synchronized in such a way that it will latch the bunch counters internal to the chips when they equal zero and cause the AqBCO status bit to be asserted. If the BCO counter internal to the chips is not equal to zero, then the AqBCO status bit will be cleared. Figure 1 shows the timing when BCO_SYNC is set.

When BCO_ZERO is set, SHIFT will be held high until the falling edge of BCOCLK one cycle before the BCO=0. In this way, the SCR command can be timed so as to synchronize the BCO counters internal to the chips with the counter implemented in the STIB firmware. After this synchronization has been performed, AqBCO commands sent with the BCO_SYNC set will always latch BCO=0 into the AqBCO output register and set the AqBCO status bit, unless bunch counter synchronization has been lost.

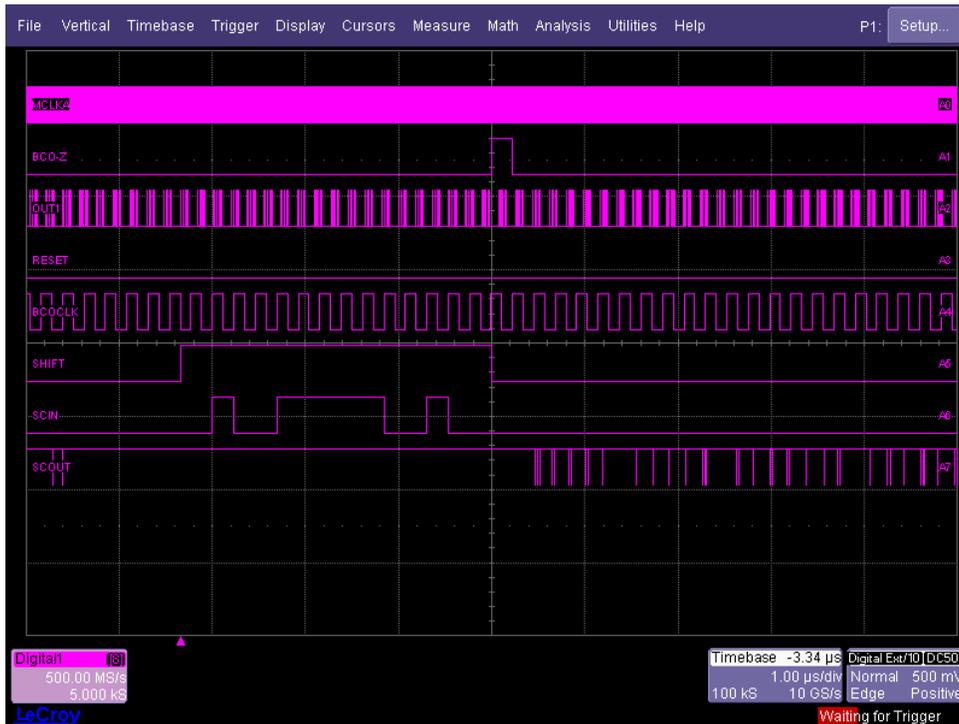


Figure 1: Example of the (AqBCO,Set) command when BCO_SYNC=1.

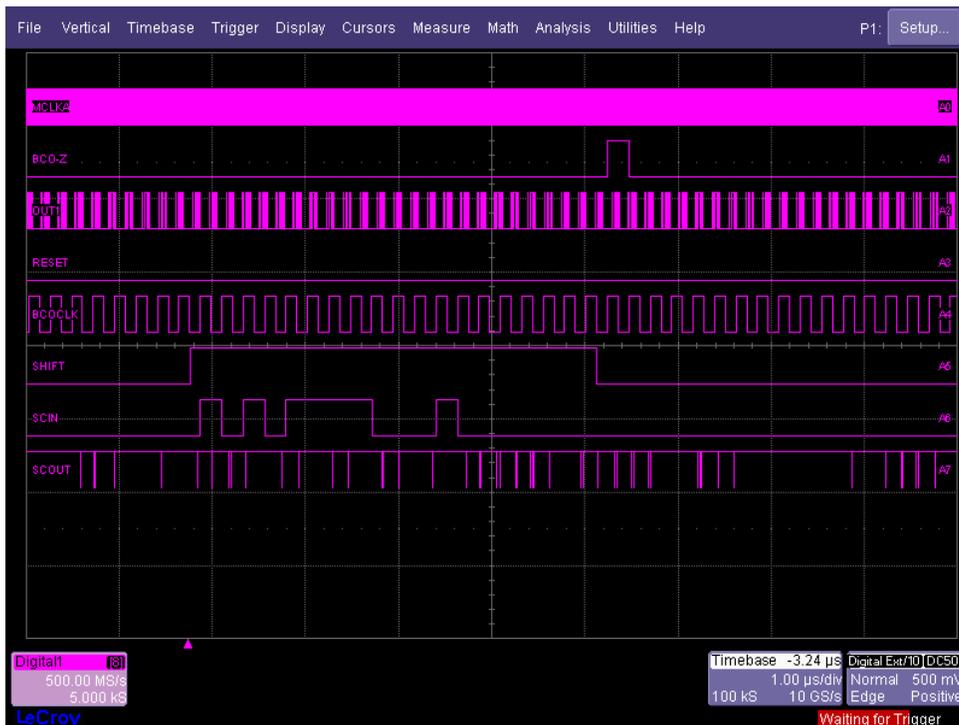


Figure 2: Example of the (SCR,Set) command when BCO_ZERO=1.

bits	name	access	description
31	ALIGNED	r	Set when the sync word is successfully aligned
30	SYNC_ERROR	r	Set for one OUTCLK cycle when a single sync error detected
29	SEND_DATA	r	SendData - copied from sync word status bits
28	REJECT_HITS	r	RejectHits - copied from sync word status bits
27	ALINES_LSB	r	Alines LSB - copied from sync word status bits
26	BCO_MISMATCH	r	Acquire BCO = 1 if number \neq 0 - copied from sync word status bits
25	PULSING_ACTIVE	r	copied from sync word status bits
24	DATA_VALID	r	Valid data on FIFO input
23-20	INTERNAL_STATE	r	internal state of link serdes FSM
19-14	MISSING_SYNC_COUNT	r	Number of consecutive data words received with no intervening sync words
13-8	DELAY_TAP_COUNT	r	Tap count applied to IDELAY
7-0	LINK_SYNC_COUNT	r	Number of OUTCLK cycles needed to synchronize link

Table 9: STRIP_SERDES_STATUS - address 0xc400xy04.

5.2.5 Link status register

STRIP_SERDES_STATUS
Base address: 0xc400
Offset: 0xXY04

R	31	30	29	28	27	26	25	24
W	ALIGNED	SYNC_ERROR	SEND_DATA	REJECT_HITS	ALINES_LSB	BCO_MISMATCH	PULSING_ACTIVE	DATA_VALID
R	23	22	21	20	19	18	17	16
W	INTERNAL STATE				MISSING SYNC COUNT			
R	15	14	13	12	11	10	9	8
W	MISSING SYNC COUNT		DELAY TAP COUNT					
R	7	6	5	4	3	2	1	0
W	LINK SYNC COUNT							

Strip No.	Code	Strip No.	Code
0	0101	4	1010
1	0111	5	1011
2	0110	6	1001
3	1110	7	1101

Table 10: Strip number encoding.

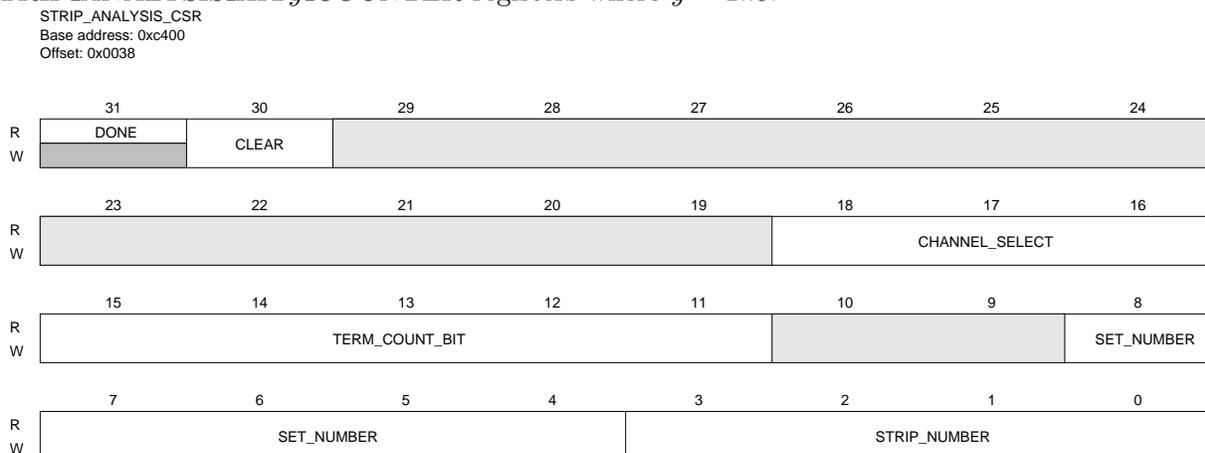
Set No.	Code	Set No.	Code
0	01010	8	10100
1	01011	9	10101
2	01111	10	10111
3	01110	11	10110
4	01100	12	10010
5	01101	13	10011
6	11101	14	11011
7	11100	15	11010

Table 11: Set number encoding.

5.2.6 Serial data analysis

The `stripana` module accumulates statistics about hit data retrieved by spying on the input to the channel FIFO. Only data with `STRIP_NUMBER` and `SET_NUMBER` matching the fields in the `STRIP_ANALYSIS_CSR` are analyzed. Mapping between set/strip number and strip channel number is provided in Tables 10 and 11.

There are six 32-bit counters that record the number of BCO clock cycles and the number of hits received from each chip. When `CLEAR` is asserted, all counters are reset to zero. When `CLEAR` is deasserted, statistics are accumulated until the BCO counter reaches a terminal count value specified by the `TERM_COUNT_BIT` field. For example, setting `TERM_COUNT_BIT = 26` terminates the accumulation of statistics when the BCO count reaches `0x04000000`. During and after accumulation, the counters can be read via the `STRIP_ANALYSIS_BCO_COUNTER` and `STRIP_ANALYSIS_HITy_COUNTER` registers where $y = 1..5$.



bits	name	access	description
31	DONE	r	Indicates that the terminal BCO count has been reached
30	CLEAR	r/w	Resets all counters - not self clearing
18..16	CHANNEL_SELECT	r/w	Selects strip sensor channel for analysis
15..11	TERM_COUNT_BIT	r/w	\log_2 of BCO count at which to stop analysis
8..4	SET_NUMBER	r/w	Set number to match in data stream
3..0	STRIP_NUMBER	r/w	Strip number to match in data stream

Table 12: STRIP_ANALYSIS_CSR - address 0xc4000038.

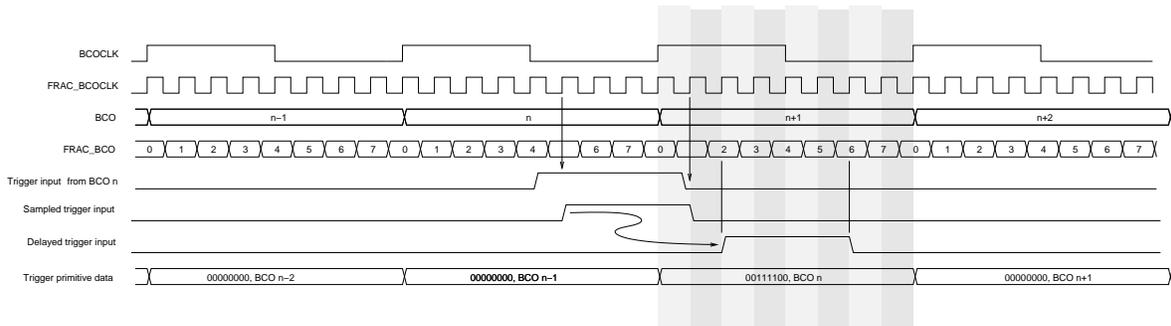


Figure 3: Generation of trigger primitive input data words. In this case, FBCO_TRIM = 5 and BCO_TRIM = -1.

5.3 Trigger interface

The design samples up to 8 trigger inputs, records the BCO number of the event that asserted them, and records the times within each BCO clock cycle at which the input signals were asserted. An asserted trigger input signal causes a configurable range of BCO numbers to be marked for readout and when bit 23 in STRIP_CSR is set, event data with BCO numbers that did not have an accompanying trigger will be dropped.

The operation of the trigger system is shown in Figure 3 and 4.

Input signals are sampled on the rising edge of the FBCO clock and delayed by up to 8 FBCO clock cycles. On the rising edge of the BCO clock, the delayed pattern is sampled and a trigger primitive is generated if the pattern is not zero. The BCO number assigned to the trigger primitive for subsequent readout is the BCO number in which it is sampled, plus a signed offset specified in the corresponding trigger input register. If the pattern is non-zero, an 8-bit mask is OR'ed with the bits in a shift register that shifts left on each BCO clock cycle. The high bit of the shift register assigned to each of the two trigger primitives is sampled and forms the trigger output signal. At this time, the output is the logical OR of the two bits, but a coincidence trigger can be formed by requiring both bits to be set.

Proposal: use the high bit in the STRIP_TRIG_INPUT_x register to control whether to AND or OR these two bits.

The four trigger outputs are written to a 4-bit wide \times 256-deep dual-port RAM, with the write address calculated from the current BCO number minus the value of BCO_OFFSET stored in the STRIP_TRIG_CSR register. When the vetoing of un-triggered events is enabled by setting bit 23

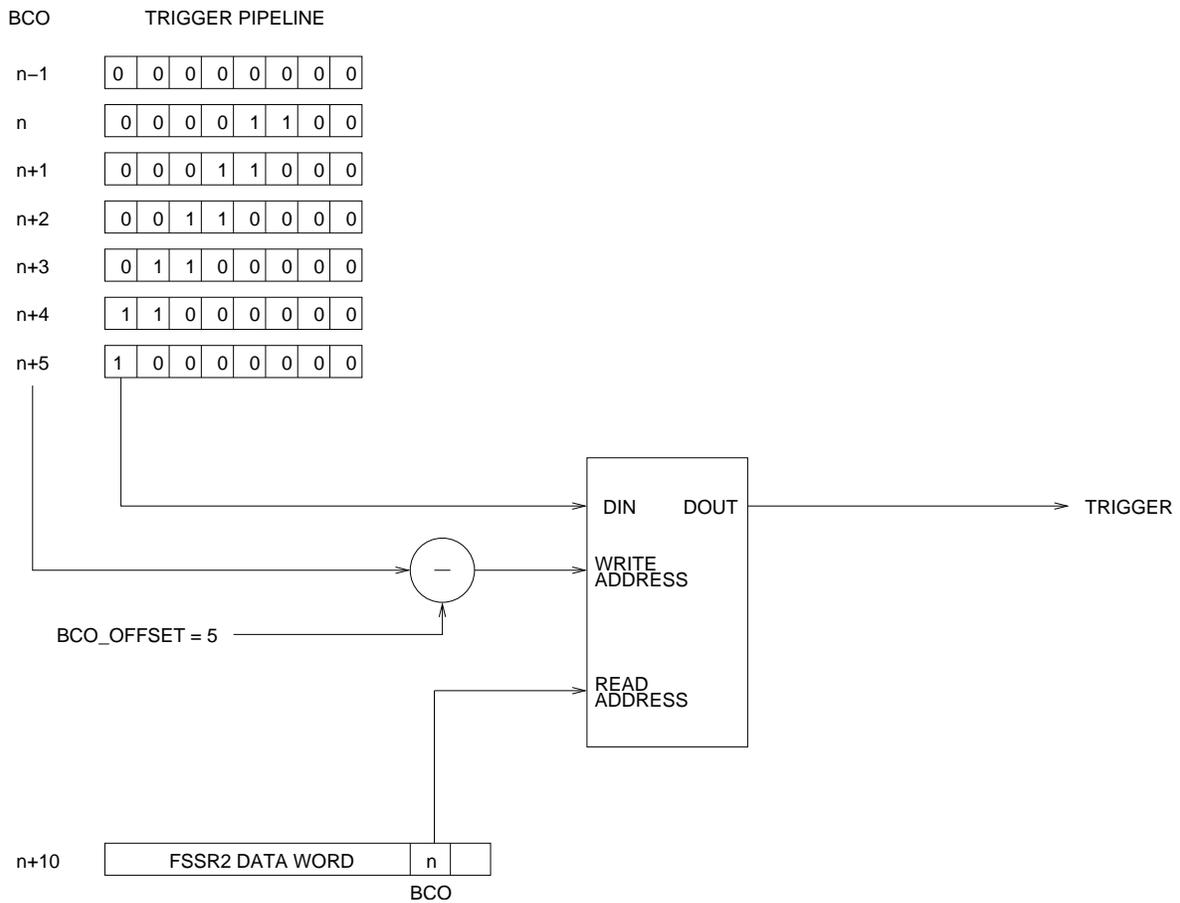


Figure 4: Example showing the trigger pipeline and dual-port RAM. Bits shifted out of the trigger pipeline are written to the dual port RAM at an address equal to the current BCO number minus a specified offset. The BCO field in the readout data is used as a read address to determine whether the event in question satisfied the trigger criteria.

input	register	description
0	STRIP_TRIG_INPUT_0	HITOR output from channel 0
1	STRIP_TRIG_INPUT_0	HITOR output from channel 1
2	STRIP_TRIG_INPUT_1	HITOR output from channel 2
3	STRIP_TRIG_INPUT_1	HITOR output from channel 3
4	STRIP_TRIG_INPUT_2	HITOR output from channel 4
5	STRIP_TRIG_INPUT_2	HITOR output from channel 5
6	STRIP_TRIG_INPUT_3	External trigger input
7	STRIP_TRIG_INPUT_3	Unbiased periodic trigger

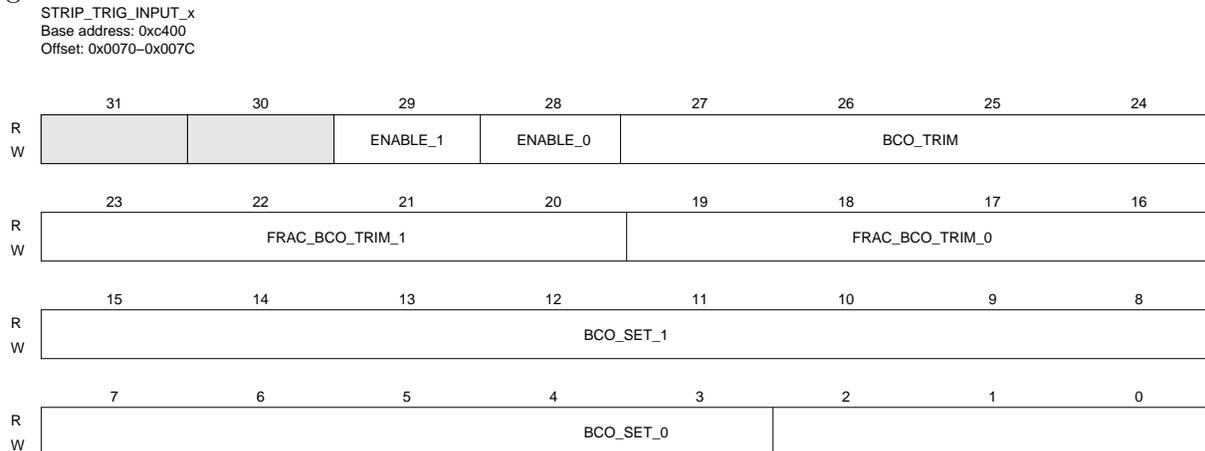
Table 13: Trigger input signal assignment.

bits	name	access	description
29	ENABLE_1	r/w	Enables sampling of input 1
28	ENABLE_0	r/w	Enables sampling of input 0
27-24	BCO_TRIM	r/w	signed offset relative to sampled BCO number
23-20	FRAC_BCO_TRIM_1	r/w	input 1 fractional BCO time slice delay
19-16	FRAC_BCO_TRIM_0	r/w	input 0 fractional BCO time slice delay
15-8	BCO_SET_1	r/w	BCO pattern set by trigger input 1
7-0	BCO_SET_0	r/w	BCO pattern set by trigger input 0

Table 14: Trigger input register description. Each of the four trigger input registers configures two inputs.

in the STRIP_CSR register, the recorded BCO numbers in the input data stream are used as the read address for the dual-port RAM. Data are sent over Ethernet when the bit pattern read from the dual-port RAM is non-zero.

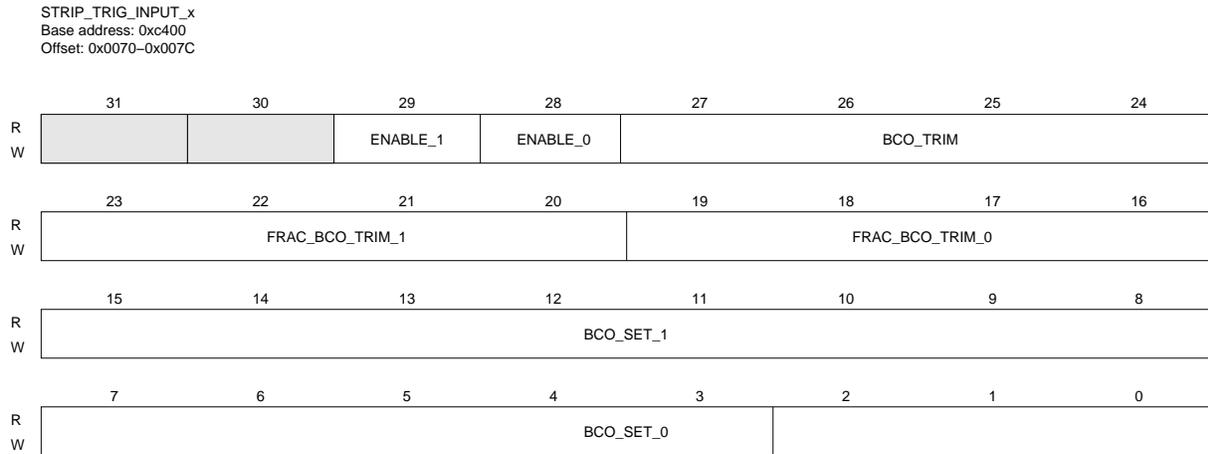
The masking and de-skewing of the trigger input signals is controlled by registers STRIP_TRIG_INPUT_ x where $x = 0..3$. The nominal signal assignment is shown in Table 13. The format of the STRIP_TRIG_INPUT_ x register is shown below and is described in Table 14.



The format of the STRIP_TRIG_CSR register is shown in Table 15.

bits	name	access	description
31-26	—	—	unused
25	TRIGNUM_FIFO_EMPTY	r	Empty flag for trigger number spy FIFO
24	TRIGNUM_FIFO_FULL	r	Full flag for trigger number spy FIFO
23	INPUT_3_FIFO_EMPTY	r	Empty flag for trigger input 3 spy FIFO
22	INPUT_3_FIFO_FULL	r	Full flag for trigger input 3 spy FIFO
21	INPUT_2_FIFO_EMPTY	r	Empty flag for trigger input 2 spy FIFO
20	INPUT_2_FIFO_FULL	r	Full flag for trigger input 2 spy FIFO
19	INPUT_1_FIFO_EMPTY	r	Empty flag for trigger input 1 spy FIFO
18	INPUT_1_FIFO_FULL	r	Full flag for trigger input 1 spy FIFO
17	INPUT_0_FIFO_EMPTY	r	Empty flag for trigger input 0 spy FIFO
16	INPUT_0_FIFO_FULL	r	Full flag for trigger input 0 spy FIFO
15-8	—	—	unused
7	HALT	r/w	Inhibits writing of dual port RAM
6-4	SPY_FIFO_SELECT	r/w	Selects trigger spy FIFO for reading
3-0	BCO_OFFSET	r/w	BCO offset to be subtracted to form write address

Table 15: Trigger control/status register description.



5.4 Streamed data interface

Hit data, bunch counter number, and trigger data is collected from all channels into a single fifo and sent as UDP packets over the Ethernet link. The destination address is stored in the STREAMPKT_DADDR (0xc10000a4) register and the source and destination ports are stored in the STREAMPKT_PORTS (0xc10000a8) register.

Four bits in the STRIP_CSR register can be set to enable the transmission of each of the four types of data. Data is buffered until the number of words written equals the number encoded in the PACKET_SIZE field of the STRIP_CSR register. The number of 32 bit words per packet is given by

$$n = 16 \times (\text{PACKET_SIZE} + 1) \quad (1)$$

or $n = 350$ when $\text{PACKET_SIZE} = 0x1f$. Except for this case, packet sizes must be smaller than the

pattern	data type	figure number
xxx1	FSSR2 hit data	6
xx10	PSI46 hit data	–
1000	STIB status/data	5

Table 16: Data types indicated by the low 4-bits in the 32-bit streamed data words.

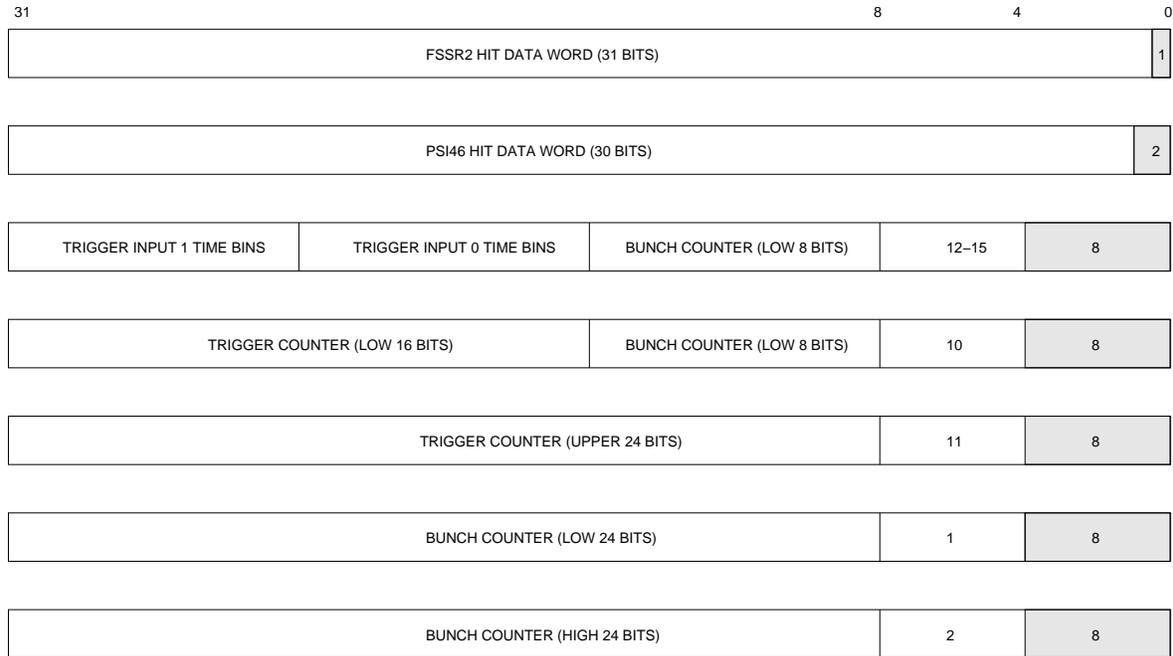


Figure 5: Format of the streamed STIB status/data words.

maximum transfer unit of approximately 1500 bytes on typical networks. The choice of packet size is a tradeoff between latency and the effective use of the network bandwidth for data rather than frame/packet header information. If hit generation has been disabled, the data that remains in the FIFO can be flushed by setting the FLUSH bit in the STRIP_CSR register (not yet implemented).

Streamed data consist of 32-bit words in which low bits to define their data type. Hit data from the FSSR2 chips has the lowest bit set to 1 (this is a sync bit), so any data word with the lowest order bit set to zero indicates some other data type. In particular, a status word is indicated by the lowest 4-bits set to "1000", and hit data from a detector that uses another type of readout chip could be indicated by other bit patterns, provided they don't conflict with these ones. Table 16 lists the bit patterns that are currently defined or envisioned.

Status words have the low 4-bits of the streamed data word set to "1000", with bits 7.4 specifying the type of status data provided in the upper 24-bits. Figure 5 shows the format of the status words that will be sent.

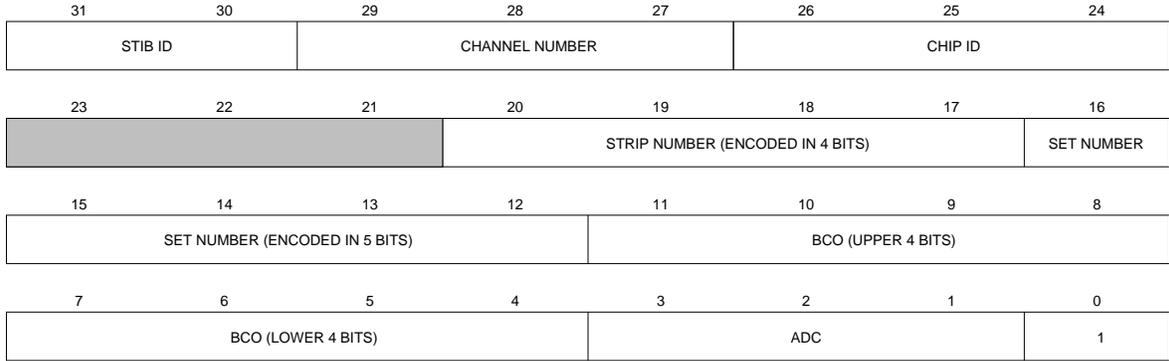


Figure 6: Format of the streamed FSSR2 hit data word.

5.4.1 FSSR2 data

Figure 6 defines the data word for FSSR2 strip hits.

5.4.2 PSI46 data

Data words from pixel hits read out using the PSI46 ROC may be packed into 30 bits in a format to be defined elsewhere.

5.4.3 STIB trigger primitive data

Provision have been made for up to 8 trigger inputs that can be sampled at the $BCOCLK \times 8$ frequency. The time bins during which the trigger signal was asserted are recorded in 8-bit trigger data fields and written as trigger input data words (types 12-15). The low 8-bits of the BCO counter during which the trigger signals were asserted is stored along with time patterns for two of the trigger inputs.

5.4.4 Trigger counter data

Events for which a trigger accept signal is generated are assigned sequential trigger numbers. A 40-bit trigger number counter is stored in two status words along with the low 8-bits of the bunch counter of the triggered event.

5.4.5 Bunch counter data

Bunch counter words are sent when bit 23 of the STRIP_CSR word is set. Bunch counter information is only transmitted immediately before a valid hit if no bunch counter words have been previously sent within the past 256 BCO clock cycles. In this way, the all information needed to determine the full 48-bit BCO counter is available as soon as the hit data words are received. For low rates, 2/3 of the data bandwidth is used by the BCO counter words, but this fraction is reduced to less than 1% of the data bandwidth at high rates.

STRIP_DAC_SPI
 Base address: 0xc400
 Offset: 0x0058

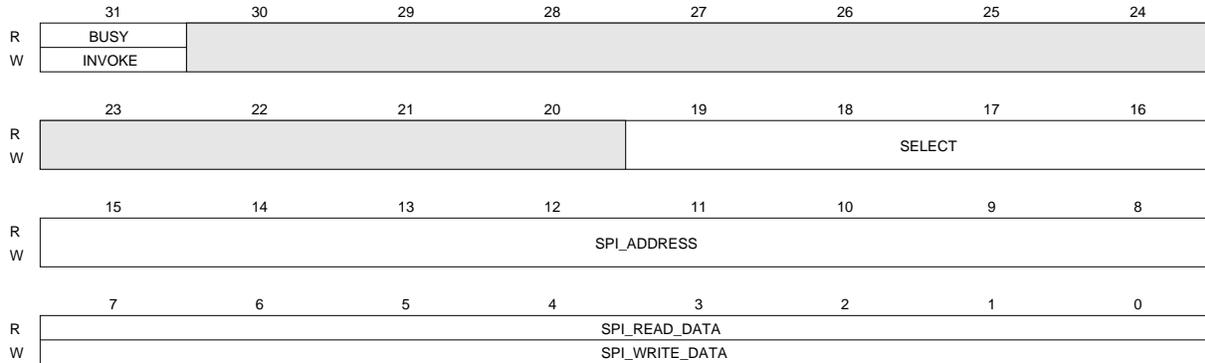


Figure 7: DAC SPI interface register.

STRIP_DAC_CSR
 Base address: 0xc400
 Offset: 0x0050

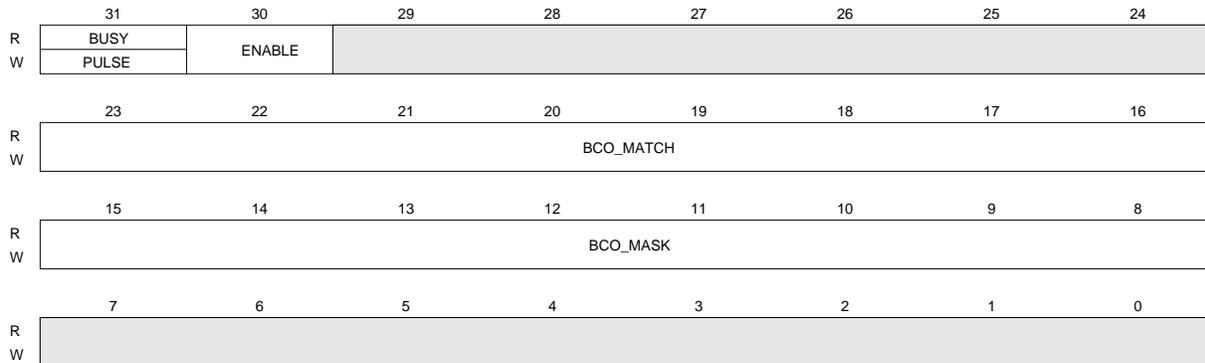


Figure 8: DAC command/status register.

5.5 DAC interface

There are four MAX5893 12-bit 500 MSPS DAC's on the STIB that are configured by means of an SPI interface through the STRIP_DAC_SPI register. The bit fields in this register are shown in Figure 7. Individual DAC's are addressed by the corresponding bits in the SELECT field. A write operation is requested when bit 16 is set to zero in which case the data stored in the low 8-bits of the STRIP_DAC_SPI register is stored in the addressed SPI register. When bit 16 is '1', an SPI register read is requested and the data from the addressed register is shifted in and made available for reading as the low 8-bits of the STRIP_DAC_SPI register. Note that there is a one read-cycle latency before the data appears.

The normal register configuration is to write 0 to register 0, 0x44 to register 1 and 0x40 to register 2. The DAC can be disabled and put into a low-power sleep state by writing 0x1c to register 0.

The STRIP_DAC_CSR register, shown in Figure 8, controls the trigger conditions for pulse generation. Bits 8..15 define an 8-bit mask and bits 16..23 define an 8-bit pattern that must match the bitwise AND of the low 8-bits of the BCO counter and the mask in order to trigger a pulse when

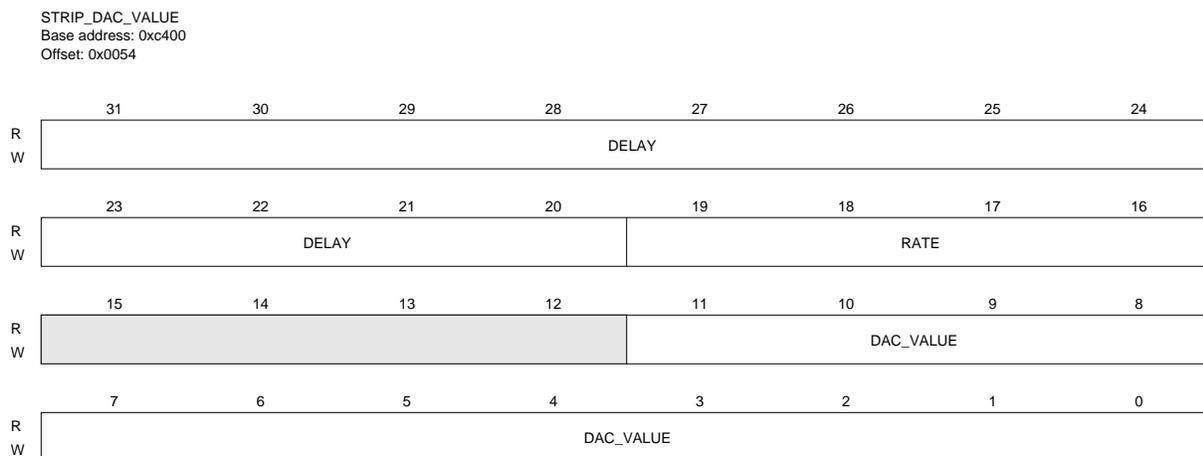


Figure 9: DAC pulse amplitude/delay control register.

sampled on the falling edge of BCOCLK. A single pulse is generated by writing a '1' to bit 31, while a continuous sequence of pulses is generated by writing a '1' to bit 30. Reading from bit 31 indicates whether the DAC trigger interface is busy.

The amplitude and precise timing of the pulse is controlled by the STRIP_DAC_VALUE register, shown in Figure 9. Bits 20..31 define the terminal count for a 200 MHz counter that adjusts the timing after the matching BCO counter before the pulse is generated. This allows the pulse to be placed at any point in the BCO sampling window. Bits 16..19 define the rate at which the pulse ramps back to baseline and bits 0..11 define the amplitude of the voltage step generated by the DAC. The FSSR2 chips respond to a negative pulse, so this field requires a negative, signed, 2's compliment number. To protect the state machine from getting stuck, a rate of 1 is substituted in the case where bits 16..19 are all zero.

Figure 10 illustrates the timing of signals generated when BCO_MATCH = 0, DELAY = 0, VALUE = -512. The pulse is generated data is read out with BCO = 4 and data becomes available for writing to the FIFO approximately 6 BCO clock cycles later.

6 Command Line Interface

The MicroBlaze core provides a command line interface that provides extensive monitoring and debugging facilities. While interaction with the monitor is not required for normal operation, it provides valuable feedback about many aspects of the internal state of the FPGA. Most commands do nothing more than display memory contents using context specific formats, or parse input arguments into bit fields and write the result to specific memory mapped registers. An interrupt handler processes incoming UDP packets, responding to NTP and DHCP packets, and UDP packets that implement the bridge between an external host on the network and the IOBUS address space.

6.1 Low-level commands

The low-level commands do not pertain to specific hardware modules or address space.

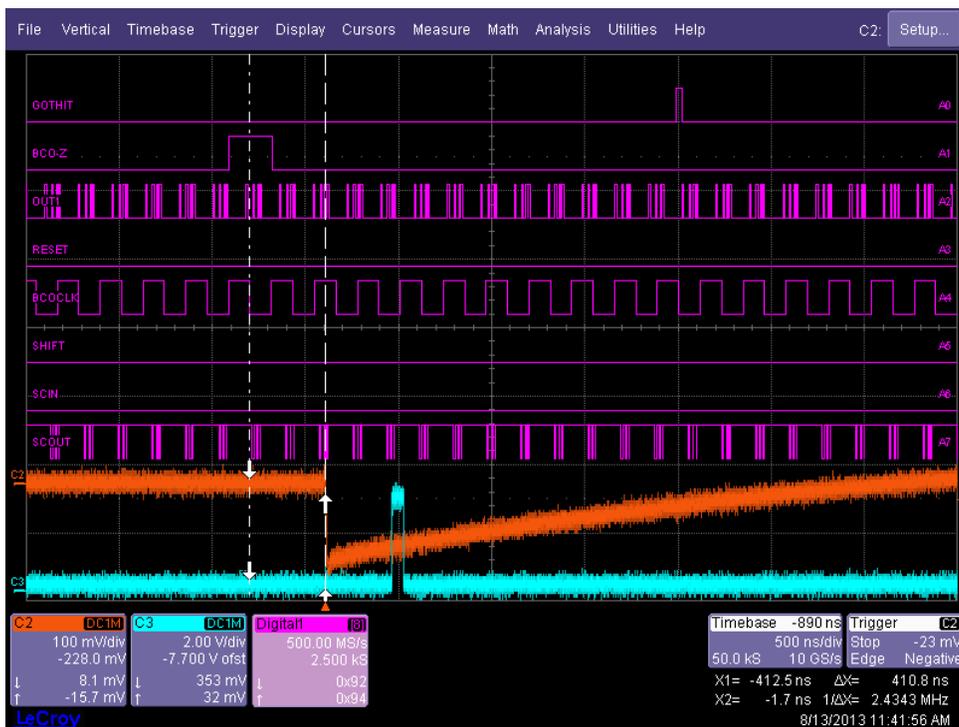


Figure 10: Example of DAC pulse timing. Channel 2 is the output of the DAC observed at the MMCX connector on the FSSR2 hybrid board. Channel 3 is the HITOR output of the FSSR2 chip in response to the input pulse. The observed delay between the trigger and the pulse output is approximately 410 ns, mostly due to the data latency in the MAX5893 DAC.

type	description
ethio	log individual GbE-IOBUS transactions (actually that's all there is right now)

Table 17: Log message filtering options.

6.1.1 peek {address}

Displays the contents of the specified 32-bit memory address.

6.1.2 poke {address} {data}

Writes the specified 32-bit data word to the 32-bit memory address.

6.1.3 log [ethio {on|off}]

Controls the level of output displayed on the console. Different message types are summarized in Table 17.

6.1.4 eint

Unmask interrupts (default).

6.1.5 dint

Mask interrupts, preventing the interrupt handler from being invoked by incoming UDP packets.

6.2 Clock interface

The clock interface was really set up to provide an example of a slave interface. However, it does provide a register that is incremented once per second and which is set to the current time when an NTP packet is received.

6.2.1 time

Displays the current time, in a particular time zone (EDT).

option	description
<code>ntp-servers</code>	address of NTP server
<code>netmask</code>	netmask local area network
<code>routers</code>	gateway router address
<code>fixed-address</code>	fixed IP address assigned to the device
<code>dhcp-client-identifier</code>	host ID associated with the device

Table 18: List of interpreted DHCP options.

6.3 Gigabit Ethernet interface

6.3.1 rx

Dumps the contents of the spy FIFO that buffers octets received from the GbE physical interface.

6.3.2 tx

Dumps the contents of the spy FIFO that records octets transmitted on the GbE physical interface.

6.3.3 udp

Displays the contents of registers associated with the UDP packet processor. These include the source and destination port, the packet length and the contents of the UDP payload.

6.3.4 arp [ip]

When an IP address is provided, generates an ARP request to retrieve its hardware address. When no argument is provided, the command displays the contents of the ARP table.

6.3.5 ntp {ip}

Formats and sends an NTP packet to the specified IP address. If the host is running an NTP server, it will respond with a NTP reply packet that will be received and acted upon by the interrupt handler to set the current time.

6.3.6 dhcp

Initiates a DHCP request broadcast on the local network. Responses are processed by the interrupt handler to retrieve various options that can be provided in the hosts `dhcpd.conf` file. The specific options that are interpreted are listed in Table 18.

6.3.7 stat

Displays Ethernet interface counters.

6.3.8 state

Displays debugging information related to the internal state of the Ethernet interface... read the code.

6.4 Strip interface

The slow controls registers are programmed by means of the STRIP_SC register, but several commands are provided to facilitate the correct formatting of the various bit fields. The set of commands that provide this interface is summarized in Table 19. Each command requires the options `read`, `set`, `reset`, `default` or `write` followed by an argument containing the data to be stored. Writing to the KILL and INJECT registers requires four 32-bit arguments to control all 128 bits. Commands are directed to individual chips selected using the `focus` command. Other commands provide ways to interact with the serdes links and data stream received from the individual readout chips.

6.4.1 focus {mask} {chip_id} {readback}

Selects the focus for slow-controls commands. The 8-bit `mask` argument selects which cables on which to drive the SHIFT and SCIN lines. The `chip_id` is the wire-bonded chip address (9-13) or the wild-card address (21). When reading from the slow-controls interface, only one cable can be selected at a time. The `readback` argument (0-7) specifies which cable will be sampled to fill the bits in the STRIP_SCO_n registers.

6.4.2 sc

The `sc` command takes no arguments and dumps the internal state of the slow controls register interface.

6.4.3 scraw

This command sets the RAW bit in the STRIP_SC register and shifts out the bit stream stored in the STRIP_SCI registers without any interpretation. This was primarily used for debugging.

6.4.4 link {status|fifo {n}}

Displays the status of the serdes associated with each chip in the system. If the link synchronization has been successfully acquired, the information provided includes the chip status sent with each sync

command	No.	register	bits	default	description
pdata	1	pulser data	?	0	?
pctrl	2	pulser control	?	0	?
vbn	3	Integrator Vbn	8	139	rarely changed
shpvbp2	4	Shaper Vbp2	8	121	
shpvbp1	5	Shaper Vbp1	8	116	
blrvbp1	6	Base-line restorer Vbp1	8	80	
dvtm	7	Discriminator Vtm	8	0	threshold low-side DAC
dvthr0	8	Discriminator Vtp<0>	8	255	DAC for discriminator threshold 0
dvthr1	9	Discriminator Vtp<1>	8	255	DAC for discriminator threshold 1
dvthr2	10	Discriminator Vtp<2>	8	255	DAC for discriminator threshold 2
dvthr3	11	Discriminator Vtp<3>	8	255	DAC for discriminator threshold 3
dvthr4	12	Discriminator Vtp<4>	8	255	DAC for discriminator threshold 4
dvthr5	13	Discriminator Vtp<5>	8	255	DAC for discriminator threshold 5
dvthr6	14	Discriminator Vtp<6>	8	255	DAC for discriminator threshold 6
dvthr7	15	Discriminator Vtp<7>	8	255	DAC for discriminator threshold 7
aline	16	Active lines	2	0	0=1 line, 1=2 lines, 2=4 lines, 3=6 lines
kill	17	Kill select	128	0	
injssel	18	Inject select	128	0	
sdata	19	Send data	1	0	0=send no data, 1=send data
reject	20	Reject hits	1	1	0=accept hits, 1=reject hits
spr	24	Smart programming reset	–	–	
dcr	27	Digital control register	8	0	
scr	28	Smart core reset	–	–	deasserts SHIFT at BCO=0
aqbco	30	Acquire BCO number	8	0	synced to BCO=0

Table 19: Command interface to slow control registers.

word on the serial link. The `fifo` argument dumps the contents of the FIFO that spys on data received on the serial link from a particular chip. An example of the output is as follows:

```
$ link fifo 2
d31ab04f : 2,3;1101,01011,04,111
d41ab04f : 2,4;1101,01011,04,111
d51ab04f : 2,5;1101,01011,04,111
d11ab081 : 2,1;1101,01011,08,000
d21ab08f : 2,2;1101,01011,08,111
d51ab08f : 2,5;1101,01011,08,111
d11ab0e1 : 2,1;1101,01011,0e,000
d21ab0ef : 2,2;1101,01011,0e,111
d31ab0ef : 2,3;1101,01011,0e,111
```

This shows hits from strip number 7, set number 1 (strip 15) on channel 2 that are above threshold. Fields in order from left to right are the raw 32-bit data word, the channel, chip, strip and set numbers, followed by the 8-bit BCO counter and the 3-bit ADC.

6.5 Example procedures

The following examples illustrate some typical operations performed using the command line interface. All arguments after “--” are treated as comments.

6.5.1 Network configuration

Although the IP address, netmask and default gateway are defined as constants in the monitor source code, they can be changed. The following example shows how they could be retrieved using DHCP and configured using the command line interface. This is a bit contrived since the advantage of having DHCP in the first place would be to make this automatic, but this level of automation would be straight forward to implement at some point in the future.

```
CAPTAN test debug rev 0.07
Built Jul 7 2013 17:19:36
ip addr 192.168.1.5 netmask 255.255.255.0 gw 192.168.1.102
Interrupts enabled.
Ready...
$ -- Issue DHCP request
$ dhcp
DHCP from 192.168.1.102 --> 192.168.1.5 OFFER
DHCP from 192.168.1.102 --> 192.168.1.5 ACK
host id = 'stib01'
server ip = 192.168.1.102
my ip = 192.168.1.5
ntp server ip = 128.46.154.76
netmask = 255.255.255.0
router = 192.168.1.102
$ -- Configure Ethernet
$ ip 192.168.1.5 255.255.255.0 192.168.1.102
$ -- Set the clock
$ ntp 128.46.154.76
NTP from 128.46.154.76 - 17:40:41 EDT
$ time
17:40:56 EDT
$
```

6.5.2 Strip configuration

The following example shows how to issue a chip reset and configure registers for triggered readout.

```
$ bcoclk int 3/2
$ bcoclk
Internal T=15 ns, f=66.666 MHz
BCOCLK = f*3/2/8 = 12.500 MHz - locked
$ poke c4000000 00000000
c4000000 = 60000002 <-- 00000000
$ -- Issue chip reset and synchronize serial links on cable 0
$ poke c4000004 d0000003
c4000004 = 00000000 <-- d0000003
$ link status
      --0--  --1--
  Aligned  11111  00001
 Sync error 00000  00000
  SendData 00000  00000
 RejectHits 11111  00001
   Alines0  00000  00000
 AcquireBC0 11111  00001
   Pulsing  00000  00000
 Valid data 00000  00000
$ poke c10000a4 c0a80166  -- Set receiver address
c10000a4 = 00000000 <-- c0a80166
$ poke c10000a8 beefb798  -- Set sender/receiver ports
c10000a8 = 00000000 <-- beefb798
$ send 192.168.1.102 47000  -- Send test packet
$ focus 3 21 0
$ dcr write 16  -- mod 256
$ scr set  -- sync to bco zero
$ -- Set discriminator thresholds
$ dthr0 write 32
$ dthr1 write 40
$ dthr2 write 48
$ dthr3 write 56
$ dthr4 write 64
$ dthr5 write 72
$ dthr6 write 80
$ dthr7 write 128
$ focus 1 9 0
$ kill write 0 0 0 1  -- Kill channel 31
$ dthr0 write 35
$ focus 1 13 0
$ dthr0 write 34
$ poke c4000000 0f800040  -- Enable triggered readout
$ poke c4000060 00000004  -- BCO offset for trigger
```

```
$ poke c4000068 0002805c -- Periodic unbiased trigger
$ poke c4000070 3f874040 -- HITOR trigger inputs 0,1
$ poke c400007c 1f070040 -- Periodic unbiased trigger input
$ dac write f 0 00 -- Default DAC configuration
$ dac write f 1 44 -- 2x interpolation, modulation off, real
$ dac write f 2 40 -- 2's compliment, dual port, rising edge
$ poke c4000050 4000ff00 -- DAC pulse trigger
$ poke c4000054 00010e00 -- DAC pulse amplitude
$ focus 3 21 0
$ sdata set
$ reject reset
```

6.5.3 Strip data analysis

The next example shows how to enable data transmission, mask all but a single channel and set the discriminator threshold low enough to fire on noise above the pedestal threshold.

```
$ -- Reset chips
$ poke c4000004 9000003f
$ link status
      --0--  --1--  --2--  --3--  --4--  --5--
Aligned 00100 00000 11111 00000 00000 10001
Sync error 00000 00000 00000 00000 00000 00000
SendData 00000 00000 00000 00000 00000 00000
RejectHits 00000 00000 11111 00000 00000 00000
Alines0 00000 00000 00000 00000 00000 00000
AcquireBCD 00000 00000 11111 00000 00000 00000
Pulsing 00000 00000 00000 00000 00000 00000
Valid data 00000 00000 00000 00000 00000 00000
$ -- Select channel 2 chip 5
$ focus 4 13 2
$ reject reset
$ sdata set
$ dthr0 write 16
$ dvtm write 8
$ link status
      --0--  --1--  --2--  --3--  --4--  --5--
Aligned 00000 00000 11111 00000 00000 10001
Sync error 00000 00000 00000 00000 00000 00000
SendData 00000 00000 00001 00000 00000 00000
RejectHits 00000 00000 11110 00000 00000 00000
Alines0 00000 00000 00000 00000 00000 00000
AcquireBCD 00000 00000 11111 00000 00000 00000
Pulsing 00000 00000 00000 00000 00000 00000
Valid data 00000 00000 00001 00000 00000 00000
$ -- Clear FIFO's
$ poke c4000004 4000003f
c4000004 = 0000003f <-- 4000003f
$ link fifo 2
d50ad941 : 2,5;0101,01101,94,000
d50ad341 : 2,5;0101,01101,34,000
d50ad1c1 : 2,5;0101,01101,1c,000
d50ad4c1 : 2,5;0101,01101,4c,000
d51d2681 : 2,5;1110,10010,68,000
d50ad141 : 2,5;0101,01101,14,000
d50ad911 : 2,5;0101,01101,91,000
d515d081 : 2,5;1010,11101,08,000
d50ad8c1 : 2,5;0101,01101,8c,000
d50ad331 : 2,5;0101,01101,33,000
```

6.5.4 Masking strip channels

The channels to mask are shifted in reverse order, so the lowest order bit in STRIP_SCI0 corresponds to channel 127.

```
$ -- mask all but channel 127
$ kill write ffffffff ffffffff ffffffff ffffffff
$ -- clear the FIFO
$ poke c4000004 4000003f
c4000004 = 0000003f <-- 4000003f
$ link status
      --0--  --1--  --2--  --3--  --4--  --5--
Aligned  00000  00000  11111  00000  00000  10001
Sync error 00000  00000  00000  00000  00000  00000
SendData  00000  00000  00001  00000  00000  00000
RejectHits 00000  00000  11110  00000  00000  00000
Alines0   00000  00000  00000  00000  00000  00000
AcquireBCD 00000  00000  11111  00000  00000  00000
Pulsing   00000  00000  00000  00000  00000  00000
Valid data 00000  00000  00000  00000  00000  00000
$ -- FIFO is empty so lower the threshold
$ dvtm write 8
$ link fifo 2
$ link fifo 2
d51ba561 :    2,5;1101,11010,56,000
d51ba0b1 :    2,5;1101,11010,0b,000
$ -- Now we only receive hits from set number 15, strip number 7 (channel 127)
```

6.6 Ethernet-IOBUS bridge

While the command line interface provides a way of interacting directly with the firmware and is useful for development and debugging, it is envisioned that the configuration of the state of the STIB system will be manipulated by means of the Gigabit Ethernet link.

6.7 ETHIO protocol

A simple protocol has been developed that provides access to the MicroBlaze memory map. The protocol uses a UDP packet to send a set of instructions that are interpreted by the MicroBlaze core. These are executed in sequence and acknowledged by sending a UDP packet to the originating ethernet address and port. The instruction format is shown in Figure 11.

The first octet in the instruction is an index that numbers the instructions in the packet consecutively starting from 1. The end of the instruction list is indicated by a zero octet.

The second octet contains an instruction code. The high bit is clear for a request and is set in the acknowledgment. The next two octets indicate the number of octets of parameters that follow.

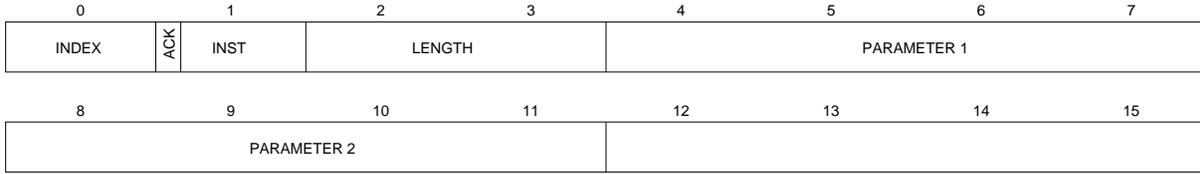


Figure 11: Ethernet-IOPBUS instruction format.

instruction	code	length	parameters
write	0x01	8	32-bit address, 32-bit data
write ack	0x81	0	–
set	0x02	8	32-bit address, 32-bit mask of bits to set
set ack	0x82	0	–
read	0x04	4	32-bit address
read ack	0x84	8	32-bit address, 32-bit data
reset	0x05	8	32-bit address, 32-bit mask of bits to clear
reset ack	0x85	0	–
waitclr	0x07	12	32-bit address, 32-bit mask of bits to test, 32-bit timeout loop count
waitclr ack	0x87	4	actual loop count until bits were cleared
waitset	0x08	12	32-bit address, 32-bit mask of bits to test, 32-bit timeout loop count
waitset ack	0x88	4	actual loop count until bits were set

Table 20: Instruction codes for the Ethernet-IOPBUS protocol.

All data words are transmitted with most-significant bits sent first. Table 20 lists the instruction codes and the data sent as arguments and returned in acknowledgments.

6.8 Discussion

The UDP protocol is described as “unreliable” in the sense that it provides no guarantee that data will be delivered, that it will be received or that it will be processed correctly. Any reliability requirements must be built into a higher level protocol that uses UDP as the mechanism for data transfer. The other end of the spectrum is TCP, which provides extensive mechanisms for determining whether data has been received and successfully delivered to a user application, and for retransmitting packets that may not have been received. Furthermore, TCP achieves high bandwidth utilization by sending packets before receiving acknowledgments of previously transmitted packets. Any packets that are retransmitted could arrive out of order and would need to be reassembled into the correct sequence by the receiver. This functionality requires a significant amount of overhead for buffering data and maintaining lists of received and acknowledged packets.

In contrast, the TFTP protocol uses UDP to send data, with the receiver acknowledging each packet that is correctly received and processed while the transmitter waits for the acknowledgments before transmitting the next packet. This achieves low latency except when packets are lost, which incurs a delay due to the timeout when waiting for the acknowledgement.

The main reliability requirements for the ETHIO protocol are as follows:

- Packets should be acknowledged, but the transmitter may choose to ignore the acknowledgment if guaranteed reliability is not necessary
- Duplicate packets should be acknowledged but not processed

Currently, the protocol addresses the first, but not the second requirement. The addition of a packet sequence number in the header would be sufficient to keep track of which packets have been processed, as is done in the TFTP protocol.

6.8.1 Collisions

The GbE interface can provide multiple clients for receiving UDP packets but the simplest instantiation includes one client that receives all UDP traffic. This creates the potential for collisions, whereby one packet's payload is overwritten by an incoming packet either before or as it is being processed. There are several ways that this possibility can be mitigated:

- Only buffer packets with a matching destination port, discarding all others
- Provide multiple UDP clients that buffer packets based on different destination ports
- Double buffer the incoming and delivered data

A Connector Nets

Channel 0 - Connector J4			
pin	CAPTAN net	STIB net	pad
1		PULSE_IN_D2	
2		A_M4.GND	
3		A_M4.GND	
4		A_M4.GND	
5	BUSDHS_01DP_02S	STRIP_GOTHITOR_PAD_P[0]	B13
6	BUSDHS_01DN_03S	STRIP_GOTHITOR_PAD_N[0]	B12
7		A_M4.GND	
8		A_M4.GND	
9	BUSDHS_02DP_04S	STRIP_OUT2.4.PAD_P[0]	A16
10	BUSDHS_02DN_05S	STRIP_OUT2.4.PAD_N[0]	A15
11		A_M4.GND	
12		A_M4.GND	
13	BUSB_10DP_20S	STRIP_MCLKA_PAD_P	AA7
14	BUSB_10DN_21S	STRIP_MCLKA_PAD_N	Y7
15	BUSDHS_03DP_06S	STRIP_OUT2.3.PAD_P[0]	A10
16	BUSDHS_03DN_07S	STRIP_OUT2.3.PAD_N[0]	B10
17	BUSDHS_04DP_08S	STRIP_OUT2.2.PAD_P[0]	K26
18	BUSDHS_04DN_09S	STRIP_OUT2.2.PAD_N[0]	K25
19	BUSB_07DP_14S	STRIP_MCLKB_PAD_P	V4
20	BUSB_07DN_15S	STRIP_MCLKB_PAD_N	U4
21		A_M4.GND	
22		A_M4.GND	
23	GENERAL_05DP_10S	STRIP_OUT2.1.PAD_P[0]	C13
24	GENERAL_05DN_11S	STRIP_OUT2.1.PAD_N[0]	C12
25	BUSD_21DP_42S	STRIP_BCOCLK_PAD_P[0]	H22
26	BUSD_21DN_43S	STRIP_BCOCLK_PAD_N[0]	H21
27	BUSD_20DP_40S	STRIP_OUT1.4.PAD_P[0]	D24
28	BUSD_20DN_41S	STRIP_OUT1.4.PAD_N[0]	C24
29	BUSD_19DP_38S	STRIP_OUT2.0.PAD_P[0]	A22
30	BUSD_19DN_39S	STRIP_OUT2.0.PAD_N[0]	A21
31		A_M4.GND	
32		A_M4.GND	
33	BUSD_18DP_36S	STRIP_OUT1.3.PAD_P[0]	D22
34	BUSD_18DN_37S	STRIP_OUT1.3.PAD_N[0]	C22
35	BUSD_17DP_34S	STRIP_SCIN_PAD_P[0]	A20
36	BUSD_17DN_35S	STRIP_SCIN_PAD_N[0]	A19
37	BUSD_16DP_32S	STRIP_SHIFT_PAD_P[0]	E21
38	BUSD_16DN_33S	STRIP_SHIFT_PAD_N[0]	D21
39	BUSD_31DP_62S	STRIP_OUT1.2.PAD_P[0]	K24
40	BUSD_31DN_63S	STRIP_OUT1.2.PAD_N[0]	K23
41		A_M4.GND	
42		A_M4.GND	
43	BUSD_30DP_60S	STRIP_RESET_PAD_P[0]	J21
44	BUSD_30DN_32S	STRIP_RESET_PAD_N[0]	J20
45	BUSD_29DP_58S	STRIP_OUT1.1.PAD_P[0]	J23
46	BUSD_29DN_58S	STRIP_OUT1.1.PAD_N[0]	J22
47	BUSD_28DP_56S	STRIP_SCOUT_PAD_P[0]	J26
48	BUSD_28DN_57S	STRIP_SCOUT_PAD_N[0]	J25
49	BUSD_27DP_54S	STRIP_OUT1.0.PAD_P[0]	H26
50	BUSD_27DN_55S	STRIP_OUT1.0.PAD_N[0]	H25

Channel 1 - Connector J3				
pin	CAPTAN net	STIB net	pad	notes
1		PULSE_IN_D1		
2		A_M3_GND		
3		A_M3_GND		
4		A_M3_GND		
5	BUSD_12DP_24S	STRIP_GOTHITOR_PAD_P[1]	H20	
6	BUSD_12DN_25S	STRIP_GOTHITOR_PAD_N[1]	G20	
7		A_M3_GND		
8		A_M3_GND		
9	BUSD_11DP_22S	STRIP_OUT2_4_PAD_P[1]	D23	
10	BUSD_11DN_23S	STRIP_OUT2_4_PAD_N[1]	C23	
11	A_M3_GND			
12	A_M3_GND			
13	BUSB_10DP_20S	STRIP_MCLKA_PAD_P	AA7	
14	BUSB_10DN_21S	STRIP_MCLKA_PAD_N	Y7	
15	BUSD_10DP_20S	STRIP_OUT2_3_PAD_P[1]	C26	
16	BUSD_10DN_21S	STRIP_OUT2_3_PAD_N[1]	C25	
17	BUSD_09DP_18S	STRIP_OUT2_2_PAD_P[1]	F20	
18	BUSD_09DN_19S	STRIP_OUT2_2_PAD_N[1]	E20	
19	BUSB_07DP_14S	STRIP_MCLKB_PAD_P	V4	
20	BUSB_07DN_15S	STRIP_MCLKB_PAD_N	U4	
21		A_M3_GND		
22		A_M3_GND		
23	BUSD_08DP_16S	STRIP_OUT2_1_PAD_P[1]	E23	
24	BUSD_08DN_17S	STRIP_OUT2_1_PAD_N[1]	E22	
25	GENERAL_02DP_04S	STRIP_BCOCLK_PAD_P[1]	G48	IO.L9P_CC.LC_5
26	GENERAL_02DN_05S	STRIP_BCOCLK_PAD_N[1]	G47	IO.L9N_CC.LC_5
27	BUSD_07DP_14S	STRIP_OUT1_4_PAD_P[1]	M23	
28	BUSD_07DN_15S	STRIP_OUT1_4_PAD_N[1]	M22	
29	BUSD_06DP_12S	STRIP_OUT2_0_PAD_P[1]	M21	
30	BUSD_06DN_13S	STRIP_OUT2_0_PAD_N[1]	M20	
31		A_M3_GND		
32		A_M3_GND		
33	BUSD_05DP_10S	STRIP_OUT1_3_PAD_P[1]	L26	
34	BUSD_05DN_11S	STRIP_OUT1_3_PAD_N[1]	M26	
35	GENERAL_03DP_06S	STRIP_SCIN_PAD_P[1]	D26	IO.L25P_CC.LC_5
36	GENERAL_03DN_07S	STRIP_SCIN_PAD_N[1]	D25	IO.L25N_CC.LC_5
37	GENERAL_04DP_08S	STRIP_SHIFT_PAD_P[1]	B47	IO.L7P_GC.LC_3
38	GENERAL_04DN_09S	STRIP_SHIFT_PAD_N[1]	A47	IO.L7N_GC.LC_3
39	BUSD_04DP_08S	STRIP_OUT1_2_PAD_P[1]	M25	
40	BUSD_04DN_09S	STRIP_OUT1_2_PAD_N[1]	M24	
41		A_M3_GND		
42		A_M3_GND		
43	BUSD_03DP_06S	STRIP_RESET_PAD_P[1]	L24	
44	BUSD_03DN_07S	STRIP_RESET_PAD_N[1]	L23	
45	BUSD_02DP_04S	STRIP_OUT1_1_PAD_P[1]	L21	
46	BUSD_02DN_05S	STRIP_OUT1_1_PAD_N[1]	L20	
47	BUSD_01DP_02S	STRIP_SCOUT_PAD_P[1]	L19	
48	BUSD_01DN_03S	STRIP_SCOUT_PAD_N[1]	K20	
49	BUSD_00DP_00S	STRIP_OUT1_0_PAD_P[1]	K22	
50	BUSD_00DN_01S	STRIP_OUT1_0_PAD_N[1]	K21	

Channel 2 - Connector J8			
pin	CAPTAN net	STIB net	pad
1		PULSE_IN_C2	
2		A_M2_GND	
3		A_M2_GND	
4		A_M2_GND	
5	BUSCHS_04DP_08S	STRIP_GOTHITOR_PAD_P[2]	R20
6	BUSCHS_04DN_09S	STRIP_GOTHITOR_PAD_N[2]	R19
7		A_M2_GND	
8		A_M2_GND	
9	GENERAL_23DP_46S	STRIP_OUT2_4.PAD_P[2]	AC13
10	GENERAL_23DN_47S	STRIP_OUT2_4.PAD_N[2]	AD13
11		A_M2_GND	
12		A_M2_GND	
13	BUSB_10DP_20S	STRIP_MCLKA_PAD_P	AA7
14	BUSB_10DN_21S	STRIP_MCLKA_PAD_N	Y7
15	BUSC_11DP_22S	STRIP_OUT2_3.PAD_P[2]	AA18
16	BUSC_11DN_23S	STRIP_OUT2_3.PAD_N[2]	Y18
17	BUSC_10DP_20S	STRIP_OUT2_2.PAD_P[2]	Y20
18	BUSC_10DN_21S	STRIP_OUT2_2.PAD_N[2]	Y21
19	BUSB_07DP_14S	STRIP_MCLKB_PAD_P	V4
20	BUSB_07DN_15S	STRIP_MCLKB_PAD_N	U4
21		A_M2_GND	
22		A_M2_GND	
23	BUSC_08DP_16S	STRIP_OUT2_1.PAD_P[2]	Y19
24	BUSC_08DN_17S	STRIP_OUT2_1.PAD_N[2]	W19
25	BUSC_07DP_14S	STRIP_BCOCLK_PAD_P[2]	AF19
26	BUSC_07DN_15S	STRIP_BCOCLK_PAD_N[2]	AF20
27	BUSC_06DP_12S	STRIP_OUT1_4.PAD_P[2]	AE21
28	BUSC_06DN_13S	STRIP_OUT1_4.PAD_N[2]	AD21
29	BUSC_05DP_10S	STRIP_OUT2_0.PAD_P[2]	AF18
30	BUSC_05DN_11S	STRIP_OUT2_0.PAD_N[2]	AE18
31		A_M2_GND	
32		A_M2_GND	
33	GENERAL_18DP_36S	STRIP_OUT1_3.PAD_P[2]	AA16
34	GENERAL_18DN_37S	STRIP_OUT1_3.PAD_N[2]	AA15
35	BUSC_04DP_08S	STRIP_SCIN_PAD_P[2]	AF21
36	BUSC_04DN_09S	STRIP_SCIN_PAD_N[2]	AF22
37	BUSC_03DP_06S	STRIP_SHIFT_PAD_P[2]	AC18
38	BUSB_03DN_07S	STRIP_SHIFT_PAD_N[2]	AB18
39	GENERAL_19DP_38S	STRIP_OUT1_2.PAD_P[2]	AB13
40	GENERAL_19DN_39S	STRIP_OUT1_2.PAD_N[2]	AA13
41		A_M2_GND	
42		A_M2_GND	
43	BUSC_02DP_04S	STRIP_RESET_PAD_P[2]	AB20
44	BUSC_02DN_05S	STRIP_RESET_PAD_N[2]	AC20
45	GENERAL_20DP_40S	STRIP_OUT1_1.PAD_P[2]	AC14
46	GENERAL_20DN_41S	STRIP_OUT1_1.PAD_N[2]	AD14
47	GENERAL_21DP_42S	STRIP_SCOUT_PAD_P[2]	AA12
48	GENERAL_21DN_43S	STRIP_SCOUT_PAD_N[2]	AA11
49	GENERAL_22DP_44S	STRIP_OUT1_0.PAD_P[2]	AC16
50	GENERAL_22DN_45S	STRIP_OUT1_0.PAD_N[2]	AC15

Channel 3 - Connector J7				
pin	CAPTAN net	STIB net	pad	notes
1		PULSE_IN_C1		
2		A_M1_GND		
3		A_M1_GND		
4		A_M1_GND		
5	BUSCC_00DP_00S	STRIP_GOTHITOR_PAD_P[3]	P20	
6	BUSCC_00DN_00S	STRIP_GOTHITOR_PAD_N[3]	P19	
7		A_M1_GND		
8		A_M1_GND		
9	BUSCC_01DP_02S	STRIP_OUT2.4_PAD_P[3]	R24	
10	BUSCC_01DN_03S	STRIP_OUT2.4_PAD_N[3]	R23	
11		A_M1_GND		
12		A_M1_GND		
13	BUSB_10DP_20S	STRIP_MCLKA_PAD_P	AA7	
14	BUSB_10DN_21S	STRIP_MCLKA_PAD_N	Y7	
15	BUSCC_02DP_04S	STRIP_OUT2.3_PAD_P[3]	R22	
16	BUSCC_02DN_05S	STRIP_OUT2.3_PAD_N[3]	R21	
17	BUSCC_03DP_06S	STRIP_OUT2.2_PAD_P[3]	T26	
18	BUSCC_03DN_07S	STRIP_OUT2.2_PAD_N[3]	U26	
19	BUSB_07DP_14S	STRIP_MCLKB_PAD_P	V4	
20	BUSB_07DN_15S	STRIP_MCLKB_PAD_N	U4	
21		A_M1_GND		
22		A_M1_GND		
23	BUSCC_04DP_08S	STRIP_OUT2.1_PAD_P[3]	U23	
24	BUSCC_04DN_09S	STRIP_OUT2.1_PAD_N[3]	V23	
25	BUSC_31DP_62S	STRIP_BCOCLK_PAD_P[3]	R26	
26	BUSC_31DN_63S	STRIP_BCOCLK_PAD_N[3]	R25	
27	BUSCC_05DP_10S	STRIP_OUT1.4_PAD_P[3]	U25	
28	BUSCC_05DN_11S	STRIP_OUT1.4_PAD_N[3]	U24	
29	BUSCC_06DP_12S	STRIP_OUT2.0_PAD_P[3]	U22	
30	BUSCC_06DN_13S	STRIP_OUT2.0_PAD_N[3]	U21	
31		A_M1_GND		
32		A_M1_GND		
33	BUSCC_07DP_14S	STRIP_OUT1.3_PAD_P[3]	T21	
34	BUSCC_07DN_15S	STRIP_OUT1.3_PAD_N[3]	T20	
35	BUSC_30DP_60S	STRIP_SCIN_PAD_P[3]	P23	
36	BUSC_30DN_61S	STRIP_SCIN_PAD_N[3]	P22	
37	BUSC_29DP_58S	STRIP_SHIFT_PAD_P[3]	P25	
38	BUSB_29DN_59S	STRIP_SHIFT_PAD_N[3]	P24	
39	BUSCHS_00DP_00S	STRIP_OUT1.2_PAD_P[3]	AD19	IO.L25P_CC.SM7.LC.7
40	BUSCHS_00DN_01S	STRIP_OUT1.2_PAD_N[3]	AC19	IO.L25N_CC.SM7.LC.7
41		A_M1_GND		
42		A_M1_GND		
43	BUSC_28DP_56S	STRIP_RESET_PAD_P[3]	N21	
44	BUSC_28DN_57S	STRIP_RESET_PAD_N[3]	N20	
45	BUSCHS_01DP_02S	STRIP_OUT1.1_PAD_P[3]	AC21	IO.L24P_CC.LC.7
46	BUSCHS_01DN_03S	STRIP_OUT1.1_PAD_N[3]	AB21	IO.L24N_CC.LC.7
47	BUSCHS_02DP_04S	STRIP_SCOUT_PAD_P[3]	AA24	IO.L8P_CC.LC.7
48	BUSCHS_02DN_05S	STRIP_SCOUT_PAD_N[3]	Y24	IO.L8N_CC.LC.7
49	BUSCHS_03DP_06S	STRIP_OUT1.0_PAD_P[3]	AC25	IO.L9P_CC.LC.7
50	BUSCHS_03DN_07S	STRIP_OUT1.0_PAD_N[3]	AC26	IO.L9N_CC.LC.7

Channel 4 - Connector J19			
pin	CAPTAN net	STIB net	pad
1		PULSE_IN_B2	
2		B_M1_GND	
3		B_M1_GND	
4		B_M1_GND	
5	BUSBHS_03DP_06S	STRIP_GOTHITOR_PAD_P[4]	AE10
6	BUSBHS_03DN_07S	STRIP_GOTHITOR_PAD_N[4]	AD10
7		B_M1_GND	
8		B_M1_GND	
9	BUSBHS_04DP_08S	STRIP_OUT2_4.PAD_P[4]	AD17
10	BUSBHS_04DN_08S	STRIP_OUT2_4.PAD_N[4]	AD16
11		B_M1_GND	
12		B_M1_GND	
13	BUSB_10DP_20S	STRIP_MCLKA_PAD_P	AA7
14	BUSB_10DN_21S	STRIP_MCLKA_PAD_N	Y7
15	GENERAL_17DP_34S	STRIP_OUT2_3.PAD_P[4]	—
16	GENERAL_17DN_35S	STRIP_OUT2_3.PAD_N[4]	AD11
17	BUSB_22DP_44S	STRIP_OUT2_2.PAD_P[4]	AC4
18	BUSB_22DN_45S	STRIP_OUT2_2.PAD_N[4]	AB4
19	BUSB_07DP_14S	STRIP_MCLKB_PAD_P	V4
20	BUSB_07DN_15S	STRIP_MCLKB_PAD_N	U4
21		B_M1_GND	
22		B_M1_GND	
23	BUSB_20DP_40S	STRIP_OUT2_1.PAD_P[4]	AA4
24	BUSB_20DN_41S	STRIP_OUT2_1.PAD_N[4]	AA3
25	BUSB_19DP_38S	STRIP_BCOCLK_PAD_P[4]	Y2
26	BUSB_19DN_39S	STRIP_BCOCLK_PAD_N[4]	Y1
27	BUSB_18DP_36S	STRIP_OUT1_4.PAD_P[4]	W6
28	BUSB_18DN_37S	STRIP_OUT1_4.PAD_N[4]	W5
29	BUSB_17DP_34S	STRIP_OUT2_0.PAD_P[4]	W4
30	BUSB_17DN_35S	STRIP_OUT2_0.PAD_N[4]	W3
31		B_M1_GND	
32		B_M1_GND	
33	BUSB_16DP_32S	STRIP_OUT1_3.PAD_P[4]	W7
34	BUSB_16DN_33S	STRIP_OUT1_3.PAD_N[4]	V7
35	BUSB_31DP_62S	STRIP_SCIN_PAD_P[4]	R4
36	BUSB_31DN_63S	STRIP_SCIN_PAD_N[4]	R3
37	BUSB_30DP_60S	STRIP_SHIFT_PAD_P[4]	P8
38	BUSB_30DN_61S	STRIP_SHIFT_PAD_N[4]	N8
39	BUSB_29DP_58S	STRIP_OUT1_2.PAD_P[4]	P5
40	BUSB_29DN_59S	STRIP_OUT1_2.PAD_N[4]	P4
41		B_M1_GND	
42		B_M1_GND	
43	BUSB_28DP_56S	STRIP_RESET_PAD_P[4]	U6
44	BUSB_28DN_57S	STRIP_RESET_PAD_N[4]	U5
45	BUSB_27DP_54S	STRIP_OUT1_1.PAD_P[4]	AD2
46	BUSB_27DN_55S	STRIP_OUT1_1.PAD_N[4]	AD1
47	BUSB_26DP_52S	STRIP_SCOUT_PAD_P[4]	AF3
48	BUSB_26DN_51S	STRIP_SCOUT_PAD_N[4]	AE3
49	BUSB_25DP_50S	STRIP_OUT1_0.PAD_P[4]	AC2
50	BUSB_25DN_49S	STRIP_OUT1_0.PAD_N[4]	AC1

Channel 5 - Connector J20			
pin	CAPTAN net	STIB net	pad
1		PULSE_IN_B1	
2		B_M2.GND	
3		B_M2.GND	
4		B_M2.GND	
5	BUSB_12DP_24S	STRIP_GOTHITOR_PAD_P[6]	AD5
6	BUSB_12DN_25S	STRIP_GOTHITOR_PAD_N[6]	AD4
7		B_M2.GND	
8		B_M2.GND	
9	BUSB_11DP_22S	STRIP_OUT2.4.PAD_P[6]	AA9
10	BUSB_11DN_23S	STRIP_OUT2.4.PAD_N[6]	Y9
11		B_M2.GND	
12		B_M2.GND	
13	BUSB_10DP_20S	STRIP_MCLKA_PAD_P	AA7
14	BUSB_10DN_21S	STRIP_MCLKA_PAD_N	Y7
15	BUSB_09DP_18S	STRIP_OUT2.3.PAD_P[6]	AF6
16	BUSB_09DN_19S	STRIP_OUT2.3.PAD_N[6]	AF5
17	BUSB_08DP_16S	STRIP_OUT2.2.PAD_P[6]	AD3
18	BUSB_08DN_17S	STRIP_OUT2.2.PAD_N[6]	AC3
19	BUSB_07DP_14S	STRIP_MCLKB_PAD_P	V4
20	BUSB_07DN_15S	STRIP_MCLKB_PAD_N	U4
21		B_M2.GND	
22		B_M2.GND	
23	BUSB_06DP_12S	STRIP_OUT2.1.PAD_P[6]	U3
24	BUSB_06DN_13S	STRIP_OUT2.1.PAD_N[6]	U2
25	BUSB_05DP_10S	STRIP_BCOCLK_PAD_P[6]	T7
26	BUSB_06DP_11S	STRIP_BCOCLK_PAD_N[6]	T6
27	BUSB_04DP_08S	STRIP_OUT1.4.PAD_P[6]	T4
28	BUSB_04DN_09S	STRIP_OUT1.4.PAD_N[6]	T3
29	BUSB_03DP_06S	STRIP_OUT2.0.PAD_P[6]	P3
30	BUSB_03DN_07S	STRIP_OUT2.0.PAD_N[6]	P2
31		B_M2.GND	
32		B_M2.GND	
33	GENERAL_12DP_24S	STRIP_OUT1.3.PAD_P[6]	T24
34	GENERAL_12DN_25S	STRIP_OUT1.3.PAD_N[6]	T23
35	BUSB_02DP_04S	STRIP_SCIN_PAD_P[6]	R6
36	BUSB_02DN_05S	STRIP_SCIN_PAD_N[6]	R5
37	BUSB_01DP_02S	STRIP_SHIFT_PAD_P[6]	R2
38	BUSB_01DN_03S	STRIP_SHIFT_PAD_N[6]	R1
39	GENERAL_13DP_26S	STRIP_OUT1.2.PAD_P[6]	AC6
40	GENERAL_13DN_27S	STRIP_OUT1.2.PAD_N[6]	AB6
41		B_M2.GND	
42		B_M2.GND	
43	BUSB_00DP_00S	STRIP_RESET_PAD_P[6]	P7
44	BUSB_00DN_01S	STRIP_RESET_PAD_N[6]	P6
45	GENERAL_14DP_28S	STRIP_OUT1.1.PAD_P[6]	U20
46	GENERAL_14DN_29S	STRIP_OUT1.1.PAD_N[6]	T19
47	GENERAL_15DP_30S	STRIP_SCOUT_PAD_P[6]	V26
48	GENERAL_15DN_31S	STRIP_SCOUT_PAD_N[6]	V25
49	GENERAL_16DP_32S	STRIP_OUT1.0.PAD_P[6]	R8
50	GENERAL_16DN_32S	STRIP_OUT1.0.PAD_N[6]	R7

Channel 6 - Connector J21			
pin	CAPTAN net	STIB net	pad
1		PULSE_IN_A2	
2		B_M3_GND	
3		B_M3_GND	
4		B_M3_GND	
5	BUSA_10DP_20S	STRIP_GOTHITOR_PAD_P[6]	C2
6	BUSA_10DN_21S	STRIP_GOTHITOR_PAD_N[6]	C1
7		B_M3_GND	
8		B_M3_GND	
9	BUSA_11DP_22S	STRIP_OUT2.4_PAD_P[6]	H8
10	BUSA_11DN_23S	STRIP_OUT2.4_PAD_N[6]	H7
11		B_M3_GND	
12		B_M3_GND	
13	BUSB_10DP_20S	STRIP_MCLKA_PAD_P	AA7
14	BUSB_10DN_21S	STRIP_MCLKA_PAD_N	Y7
15	BUSA_13DP_26S	STRIP_OUT2.3_PAD_P[6]	G6
16	BUSA_13DN_27S	STRIP_OUT2.3_PAD_N[6]	G5
17	BUSA_14DP_28S	STRIP_OUT2.2_PAD_P[6]	F8
18	BUSA_14DN_29S	STRIP_OUT2.2_PAD_N[6]	G8
19	BUSB_07DP_14S	STRIP_MCLKB_PAD_P	V4
20	BUSB_07DN_15S	STRIP_MCLKB_PAD_N	U4
21		B_M3_GND	
22		B_M3_GND	
23	BUSA_24DP_48S	STRIP_OUT2.1_PAD_P[6]	H6
24	BUSA_24DN_49S	STRIP_OUT2.1_PAD_N[6]	H5
25	BUSA_25DP_50S	STRIP_BCOCLK_PAD_P[6]	G2
26	BUSA_25DN_51S	STRIP_BCOCLK_PAD_N[6]	G1
27	BUSA_26DP_52S	STRIP_OUT1.4_PAD_P[6]	H4
28	BUSA_26DN_53S	STRIP_OUT1.4_PAD_N[6]	H3
29	BUSA_27DP_54S	STRIP_OUT2.0_PAD_P[6]	H2
30	BUSA_27DN_55S	STRIP_OUT2.0_PAD_N[6]	H1
31		B_M3_GND	
32		B_M3_GND	
33	BUSA_28DP_56S	STRIP_OUT1.3_PAD_P[6]	J2
34	BUSA_28DN_57S	STRIP_OUT1.3_PAD_N[6]	J1
35	BUSA_29DP_58S	STRIP_SCIN_PAD_P[6]	J5
36	BUSA_29DN_59S	STRIP_SCIN_PAD_N[6]	J4
37	BUSA_30DP_60S	STRIP_SHIFT_PAD_P[6]	J7
38	BUSA_30DN_61S	STRIP_SHIFT_PAD_N[6]	J6
39	BUSA_31DP_62S	STRIP_OUT1.2_PAD_P[6]	L1
40	BUSA_31DN_63S	STRIP_OUT1.2_PAD_N[6]	K1
41		B_M3_GND	
42		B_M3_GND	
43	BUSA_16DP_32S	STRIP_RESET_PAD_P[6]	C5
44	BUSA_16DN_32S	STRIP_RESET_PAD_N[6]	D5
45	BUSA_17DP_34S	STRIP_OUT1.1_PAD_P[6]	A9
46	BUSA_17DN_35S	STRIP_OUT1.1_PAD_N[6]	B9
47	GENERAL_11DP_22S	STRIP_SCOUT_PAD_P[6]	E13
48	GENERAL_11DN_23S	STRIP_SCOUT_PAD_N[6]	D12
49	BUSAHS_04DP_08S	STRIP_OUT1.0_PAD_P[6]	L4
50	BUSAHS_04DN_09S	STRIP_OUT1.0_PAD_N[6]	L3

Channel 7 - Connector J24			
pin	CAPTAN net	STIB net	pad
1		PULSE_IN_A1	
2		B_M4_GND	
3		B_M4_GND	
4		B_M4_GND	
5	BUSAHS_03DP_06S	STRIP_GOTHITOR_PAD_P[7]	D2
6	BUSAHS_03DN_07S	STRIP_GOTHITOR_PAD_N[7]	D1
7		B_M4_GND	
8		B_M4_GND	
9	BUSAHS_02DP_04S	STRIP_OUT2_4_PAD_P[7]	G10
10	BUSAHS_02DN_05S	STRIP_OUT2_4_PAD_N[7]	G9
11		B_M4_GND	
12		B_M4_GND	
13	BUSB_10DP_20S	STRIP_MCLKA_PAD_P	AA7
14	BUSB_10DN_21S	STRIP_MCLKA_PAD_N	Y7
15	BUSAHS_01DP_02S	STRIP_OUT2_3_PAD_P[7]	E3
16	BUSAHS_01DN_03S	STRIP_OUT2_3_PAD_N[7]	E2
17	BUSAHS_00DP_00S	STRIP_OUT2_2_PAD_P[7]	B6
18	BUSAHS_00DN_01S	STRIP_OUT2_2_PAD_N[7]	C6
19	BUSB_07DP_14S	STRIP_MCLKB_PAD_P	V4
20	BUSB_07DN_15S	STRIP_MCLKB_PAD_N	U4
21		B_M4_GND	
22		B_M4_GND	
23	BUSAA_07DP_14S	STRIP_OUT2_1_PAD_P[7]	N7
24	BUSAA_07DN_15S	STRIP_OUT2_1_PAD_N[7]	M7
25	BUSA_20DP_40S	STRIP_BCOCLK_PAD_P[7]	C4
26	BUSA_20DN_41S	STRIP_BCOCLK_PAD_N[7]	D4
27	BUSAA_06DP_12S	STRIP_OUT1_4_PAD_P[7]	M6
28	BUSAA_06DN_13S	STRIP_OUT1_4_PAD_N[7]	M5
29	BUSAA_05DP_10S	STRIP_OUT2_0_PAD_P[7]	M4
30	BUSAA_05DN_11S	STRIP_OUT2_0_PAD_N[7]	M3
31		B_M4_GND	
32		B_M4_GND	
33	BUSAA_04DP_08S	STRIP_OUT1_3_PAD_P[7]	M2
34	BUSAA_04DN_09S	STRIP_OUT1_3_PAD_N[7]	M1
35	BUSA_21DP_42S	STRIP_SCIN_PAD_P[7]	E1
36	BUSA_21DN_43S	STRIP_SCIN_PAD_N[7]	F1
37	BUSA_22DP_44S	STRIP_SHIFT_PAD_P[7]	F4
38	BUSA_22DN_45S	STRIP_SHIFT_PAD_N[7]	F3
39	BUSAA_03DP_06S	STRIP_OUT1_2_PAD_P[7]	L7
40	BUSAA_03DN_07S	STRIP_OUT1_2_PAD_N[7]	L6
41		B_M4_GND	
42		B_M4_GND	
43	BUSA_23DP_46S	STRIP_RESET_PAD_P[7]	G4
44	BUSA_23DN_47S	STRIP_RESET_PAD_N[7]	G3
45	BUSAA_02DP_04S	STRIP_OUT1_1_PAD_P[7]	K7
46	BUSAA_02DN_05S	STRIP_OUT1_1_PAD_N[7]	K6
47	BUSAA_01DP_02S	STRIP_SCOUT_PAD_P[7]	K5
48	BUSAA_01DN_03S	STRIP_SCOUT_PAD_N[7]	K4
49	BUSAA_00DP_00S	STRIP_OUT1_0_PAD_P[7]	K3
50	BUSAA_00DN_01S	STRIP_OUT1_0_PAD_N[7]	K2