1 Using the V2/V3 F1TDC Module (8/26/13)

1.1 Controlling the Module

Communication with the module is by standard VME bus protocols. All registers and memory locations are defined to be 4-byte entities. The VME slave module has three distinct address ranges.

A24 – The base address of this range is set by a 12-element DIP switch on the board. It occupies 4 Kbytes of VME address space, organized in 1 K 32-bit words. Relative to the base address, this space is utilized as follows:

000-FFF – Register space to control and monitor the module

A32 - The base address of this range is programmed into register ADR32. It occupies 8 Mbytes of VME address space, organized in 2 M 32-bit words. A read of any address in this range will yield the next TDC data word from the module. Even though the module is a FIFO, the expanded address range allows the VME master to increment the address during block transfers. This address range can participate in single cycle, 32-bit block, and 64-bit block reads. The only valid write to this address range is the data value 0x80000000 which re-enables the module to generate interrupts (after one has occurred). The address range must be enabled by setting ADR32[0] = 1.

A32 - The lower and upper limits of this address range are programmed into register ADR_MB. This common address range for a set of TDC modules in the crate is used to implement the Multiblock protocol. By means of token passing TDC data may be read out from multiple TDC modules using a single logical block read. The board possessing the token will respond to a read cycle in this address range with the next TDC data word from that module. The token is passed along a private daisy chain line to the next module when it has transferred all data from a programmed number of events (register BLOCK SIZE). The address range must be enabled by setting ADR_MB[0] = 1.

1.2 Module Operation

The following describes the setup and operation of the TDC in single and multiple module applications.

Single Module – After a reset of the module (CSR[31] = 1), the F1 chip configuration registers may be written directly through an A24 register (3C). The reference clock source is set to INTERNAL (CTRL[2..0] = 3). The sync_reset source is set to SOFT (CTRL[4..3] = 3, CTRL[9] = 1), and a sync_reset signal is generated (CSR[28] = 1). The BLOCK SIZE register is loaded with the number of events (i.e. triggers) that constitute a *block.* The INTERRUPT register may be loaded with the interrupt ID and level if the module is to initiate an interrupt when the defined *block* of data is available for readout. The address for data access is loaded (ADR32). The event level interrupt (CTRL[24] = 1) is enabled if interrupt generation is desired. The BERR response is enabled (CTRL[25] = 1) to allow the module to indicate when the complete *block* of data has been read out. Permit data flow (CTRL2(0) = 1). Then set the trigger source to FRONT PANEL (CTRL[6..5] = 1).

When the programmed number of triggers has been received, the Block Ready Flag (CSR[4]) will be set and an interrupt will be generated if enabled. The user should initiate a DMA block read (32 or 64-bit) from the address in stored in ADR32. The length of the block read should be programmed to be larger than the expected size of the data *block* (e.g. 4 Mbytes). The module will terminate the DMA transfer by issuing BERR when all data from the *block* has been transferred. Interrupt generation must be re-enabled by writing 0x80000000 to the address in ADR32.

<u>Multiple Modules</u> – All modules should be reset and loaded with the configuration register values as described above for a single module. A P0 (VXS) distribution of reference clock, sync_reset, and trigger signal is assumed here. Set CTRL[2..0] = 0, CTRL[4..3] = 2, CTRL[6..5] = 2. The BLOCK SIZE register is loaded in each module. A <u>unique</u> address for data access is loaded into ADR32 for each module. The <u>common</u> address range for the Multiblock protocol is loaded into ADR_MB for each module. All modules are programmed to participate in the Multiblock protocol (CTRL[26] = 1). The *left-most* TDC module in the system is programmed as the *first* module (CTRL[27] = 1), and the *right-most* TDC module is designated the *last* module (CTRL[28] = 1). For the *first* module, the INTERRUPT register may be loaded and the event level interrupt bit enabled, if desired. The BERR response is enabled for the *last* module only. For VXS P0 signal distribution systems, the *token* passing lines are included in the system. Front signal distribution systems need external connections on the backplane (P2) between slots for token passing lines. Permit data flow in each module (CTRL2(0) = 1).

When the programmed number of triggers has been received, the *event level flag* will be set in each module and an interrupt will be generated by the *first* module (if enabled). The user should initiate a DMA block read (32 or 64-bit) from the address stored in ADR_MB. The length of the block read should be programmed to be larger than the total size of data from all modules (e.g. 4 Mbytes x # modules). Since the *first* module initially has the *token*, it will respond with data to the VME bus cycles. When data from the *first* module's block has been depleted, it passes the *token* to the next module in the chain. This module will respond with data until its data block is exhausted and the *token* is passed to the next module. The *last* module will terminate the DMA transfer by issuing BERR when all data from its *block* has been transferred. The token must be returned to the *first* board by writing '0x1000000' to the CSR register of the *first* board. Interrupt generation must be re-enabled by writing 0x80000000 to the address in ADR32 of the *first* module.

1.3 Module Registers

VERSION – board/firmware revision (0x0)

[7…0] – (R) – firmware revision

[15…8] – (R) – board revision

[31…16] – (R) – board type (“00F1”)

CSR – Control/Status (0x4)

0 – (R) – Reference clock PLL locked status (1 = locked)

1 – (R) – Board clock PLL locked status (1 = locked)

2 – (R) – TDC chip configuration error

3 – (R) – Block of Events Accepted (Event Level Flag)

4 – (R) – Block of Events Ready for readout

5 – (R) – BERR status (1 = module asserted BERR)

6 – (R) – Token status (1 = module has token)

7 – (R) – Events on board empty flag status (1 = no events)

8 – (R) – TDC chip 0 error

9 – (R) – TDC chip 1 error

10 – (R) – TDC chip 2 error

11 – (R) – TDC chip 3 error

12 – (R) – TDC chip 4 error

13 – (R) – TDC chip 5 error

14 – (R) – TDC chip 6 error (V2 only)

15 – (R) – TDC chip 7 error (V2 only)

16 – (R) – Spare (read as ‘0’)

17 – (R) – Internal Buffer #0 empty flag

18 – (R) – Internal Buffer #1 empty flag

19 – 23 – Spare (read as zero)

24 – (W) – Token Return – return token to 1st module (after readout)

25 – 26 – Spare

27 – (W) – Pulse software generated START (if CTRL[9] = 1)

28 – (W) – Pulse software generated SYNC_RESET (if CTRL[9] = 1)

29 – (W) – Pulse software generated TRIGGER (if CTRL[9] = 1)

30 – (W) – Pulse soft reset

31 – (W) – Pulse hard reset

<u>CTRL – Control</u> (0x8)

0 – (R/W) – Reference clock select (0 = P0, 1 = internal/front panel)

1 – (R/W) – Internal reference clock enable (0 = OFF, 1 = ON)

2 – (R/W) – Internal/front panel reference clock select (only if CTRL[0] = 1) (0 = internal, 1 = front panel)

[4…3] – (R/W) – SYNC_RESET source (0 = none, 1 = FP, 2 = P0, 3 = soft)

[6…5] – (R/W) – TRIGGER source (0 = none, 1 = FP, 2 = P0, 3 = soft)

[8…7] – (R/W) – START source (0 = none, 1 = FP, 2 = P0, 3 = soft)

9 – (R/W) – Enable Soft control signals (SYNC_RESET, TRIGGER, START)

10 – 23 – Spare

24 – (R/W) – Enable Interrupt

25 – (R/W) – Enable BERR response

26 – (R/W) – Enable Multiblock protocol

27 – (R/W) – FIRST board in Multiblock system

28 – (R/W) – LAST board in Multiblock system

29 – (R/W) – Multiblock Token passed on P0

30 – (R/W) – Multiblock Token passed on P2

31 – Spare

<u>EVENT COUNT</u> (0xC) – Number of events currently stored on board

[23…0] - (R) – Event count[23…0]

[31…24] – (not used – read as ‘0’)

<u>BLOCK SIZE</u> (0x10)

[15…0] - (R/W) – Number of events defining a block

[31…16] – (not used = read – as ‘0’)

<u>INTERRUPT</u> (0x14)

[7…0] – (R/W) – Interrupt ID (vector)

[10…8] – (R/W) – Interrupt Level [2..0]. Valid values = 1,..,7.

11 - 15 – (not used)

[20…16] – (R) – Geographic Address (slot number) in VME64x chassis.

21 – 22 – (not used – read as ‘0’)

23 – (R) – Parity Error in Geographic Address.

24 – 31 – (not used – read as ‘0’)

<u>ADR32 – Address for data access</u> (0x18)

0 – (R/W) – Enable 32-bit address decoding

1 – 5 – (not used – read as 0)

[15…6] – (R/W) – Base Address for 32-bit addressing mode (4 Mbyte total)

ADR_MB – Multiblock Address for data access (0x1C)

0 – (R/W) – Enable Multiblock address decoding

1 – 5 – (not used – read as 0)

[15…6] – (R/W) – Lower Limit address (ADR_MIN) for Multiblock access

16 – 21 – (not used – read as 0)

[31…22] – (R/W) – Upper Limit address (ADR_MAX) for Multiblock access

The board that has the TOKEN will respond with data when the VME address satisfies the following condition:

$$ADR\_MIN \leq Address < ADR\_MAX.$$

TDC STATUS (4 registers)

(0xXX – TDC chips 'A' & 'B'): (0x20 – TDC chips 0 & 1)
(0x24 – TDC chips 2 & 3)
(0x28 – TDC chips 4 & 5)
(0x2C – TDC chips 6 & 7)

(bits 0 – 15 for chip 'B', bits 16 – 31 for chip 'A')

0 – Resolution LOCKED (B)

1 – TDC Hit FIFO Overflow (B)

2 – TDC Trigger FIFO Overflow (B)

3 – TDC Output FIFO Overflow (B)

4 – External FIFO Full (B)

5 – External FIFO Almost Full – BUSY asserted if chip enabled (B)

6 – External FIFO Empty (B)

7 – TDC Initialized (B)

8 – Loss of Resolution Lock Occurred (B)

9 – TDC Hit FIFO Overflow Occurred (B)

10 – TDC Trigger FIFO Overflow Occurred (B)

11 – TDC Output FIFO Overflow Occurred (B)

12 – External FIFO Full Occurred (B)

13 – 15 – (not used – read as 0)

16 – Resolution LOCKED (A)

17 – TDC Hit FIFO Overflow (A)

18 – TDC Trigger FIFO Overflow (A)

19 – TDC Output FIFO Overflow (A)

20 – External FIFO Full (A)

21 – External FIFO Almost Full – BUSY asserted if chip enabled (A)

22 – External FIFO Empty (A)

23 – TDC Initialized (A)

24 – Loss of Resolution Lock Occurred (A)

25 – TDC Hit FIFO Overflow Occurred (A)

26 – TDC Trigger FIFO Overflow Occurred (A)

27 – TDC Output FIFO Overflow Occurred (A)

28 – External FIFO Full Occurred (A)

29 – 31 – (not used – read as 0)

<u>SCALER 1 – Event Count</u> (0x30)

[31…0] – (R) – total event count

<u>SCALER 2 – Event Counters</u> (0x34)

[3…0] – (R) – low 4 bits of event counter for TDC chip #1

[7…4] – (R) – low 4 bits of event counter for TDC chip #2

[11…8] – (R) – low 4 bits of event counter for TDC chip #3

[15…12] – (R) – low 4 bits of event counter for TDC chip #4

[19…16] – (R) – low 4 bits of event counter for TDC chip #5

[23…20] – (R) – low 4 bits of event counter for TDC chip #6

[27…24] – (R) – low 4 bits of event counter for TDC chip #7

[31…28] – (R) – low 4 bits of event counter for TDC chip #8

<u>SPARE</u> (0x38) (R/W)

<u>F1TDC CHIP CONFIGURATION</u> (0x3C) (R/W)

[23…21] – F1 chip address

20 – Common bit (1 = broadcast)

[19…16] – register address on F1 chip

[15…0] – data for F1 register selected

<u>CTRL2 (R/W)</u> (0x40)

0 – GO DATA – allow data to be accepted from F1 chips

15 – Force BUSY output

16 – System Test Mode (overrides Single Board Test Mode)

17 – Single Board Test Mode

18 – Trigger LED on (Single Board Test Mode)

19 – STATUS LED on (Single Board Test Mode)

<u>CONFIGURATION_CSR</u> (0x44) – (Firmware Update)

31 – (R/W) – Write Enable (1 = write mode, 0 = read mode)

30 – (R/W) – Bulk Erase (bit 31 = 1 also required)

29 – (R/W) – Sector Erase (bit 31 = 1 also required)

[28…16] – (R/W) – Reserved

[15…9] – (R) – Reserved

8 – (R) – Busy (operation in progress)

[7…0] – (R) – Last Valid Data Read

<u>CONFIGURATION_ADR/DATA (R/W)</u> (0x48) – (Firmware Update)

[31…8] – EPROM address

[7…0] – EPROM data to write

<u>SERIAL EPROM (R/W)</u> (0x4C) – (Module Serial Number)

The serial number is 7 ASCII characters long, and extends from EPROM byte address 0 thru 6. Read the serial number character-by-character. To read one character:
WRITE data word **0x3000 | eprom_byte_address** to this register,
READ data word from this register. Character = data_word >> 24

<u>TRIGGER 2 SCALER</u> – (0x50)

[31…0] – (R) – total TRIGGER 2 count

<u>SYNC_RESET SCALER</u> – (0x54)

[31…0] – (R) – total SYNC_RESET count

TEST SCALER – (0x58)

TOKEN TEST (0x5C)

TOKEN TEST SCALER (0x60)

AUXILIARY ‘B’ (3 registers) (0x64 – 0x6C) – **V2 ONLY**

PULSE DELAY (0x64) – **V3 ONLY**

PULSER DAC (0x68) – **V3 ONLY**

PULSER CONTROL (0x6C) – **V3 ONLY**

BLOCK COUNT – (0x70)

[31…20] – not used

[19…0] – (R) - number of event BLOCKS on board (non-zero → CSR[4] = 1).

BLOCK FIFO COUNT – (0x74)

[31…6] – not used

[5…0] – (R) - number of entries in BLOCK WORD COUNT FIFO

BLOCK WORD COUNT FIFO – (64 deep FIFO) (0x78)

[31…25] – not used (read as ‘0’)

24 – (R) – count not valid (word count FIFO empty)

[23…20] – not used (read as ‘0’)

[19…0] – (R) - number of words in next event BLOCK

STATUS SPARE – (0x7C)

System Test Registers (0x80 – 0x9C) (set CTRL(17) = 1 (System Test Mode))

TEST BIT REGISTER (0x80)

0 – (R/W) – trigger_out_p0 (1 = asserted, 0 = not asserted)

1 – (R/W) – busy_out_p0 (1 = asserted, 0 = not asserted)

2 – (R/W) – sdlink_out_p0 (1 = asserted, 0 = not asserted)

3 – (R/W) – token_out_p0 (1 = asserted, 0 = not asserted)

[4 – 7] – (R/W) – spare out test bits

8 – (R) – status_b_in_p0 state (1 = asserted, 0 = not asserted)

9 – (R) – token_in_p0 state (1 = asserted, 0 = not asserted)

[10 - 14] – (R) – reserved (read as '0')

15 – (R) – 'clk_31.25' counter status (1 = counting, 0 = not counting)

[16 - 31] – (R) – reserved (read as '0')

CLOCK_31.25 COUNT REGISTER (0x84)

0 – (W) – Write '0' resets the counter. Write '1' initiates 20 us counting interval.

[31 - 0] – (R) – CLK_31.25 counter value. (Should be 625 after count interval.)

SYNC_IN_P0 COUNT REGISTER (0x88)

0 – (W) – Write '0' resets the counter.

[31 - 0] – (R) – SYNC_IN_P0 counter value.

<u>TRIG1_IN_P0 COUNT REGISTER</u> (0x8C)

0 – (W) – Write '0' resets the counter.

[31 - 0] – (R) – TRIG1_IN_P0 counter value.

<u>TRIG2_IN_P0 COUNT REGISTER</u> (0x90)

0 – (W) – Write '0' resets the counter.

[31 - 0] – (R) – TRIG2_IN_P0 counter value.

<u>TEST SPARE 1</u> – (0x94)

<u>TEST SPARE 2</u> – (0x98)

<u>TEST SPARE 3</u> – (0x9C)

1.4 Data Format

The F1TDC chip outputs 24-bit words. The words are of 2 types: header/trailer and data. The header/trailer words are used as event or channel separators, and provide information such as the event number and trigger time. A data word contains the time measurement for a hit. The bit assignments are shown below.

Header / Trailer word	0	1 Bit Trigger FIFO overflow	6 Bit Event number	9 Bit Trigger time	1 Bit Xor setup register	3 Bit Chip address	3 Bit Channel address

data word time measurement	1	0	3 Bit Chip address	3 Bit Channel address	16 Bit Time

The header/trailer words can be enabled for each channel of the chip. The headers/trailers take up space in the F1TDC chip's output buffer while providing redundant information. So we enable only the header for channel 0, and the trailer for channel 7. This minimal header/trailer information is used to assemble event fragments from different chips into a single event fragment that is associated with a trigger for the board. The <u>chip data stream</u> then appears as follows:

header – channel 0
    data – channel 0, earliest hit
        …
    data – channel 0, latest hit
    data – channel 1, earliest hit
        …
    data – channel 1, latest hit
        …
        …
    data – channel 7, earliest hit
        …
    data – channel 7, latest hit
trailer – channel 7

During event building on the module, all trailers are deleted. In the following discussion the module is assumed to have 8 F1 chips (V2). (The V3 module has 6 F1 chips.) The data stream associated with a trigger will appear as follows:

header – chip 0, channel 0

data – chip 0, channel 0, earliest hit
…
data – chip 0, channel 0, latest hit
…
…
data – chip 0, channel 7, earliest hit
…
data – chip 0, channel 7, latest hit
header – chip 1, channel 0
data – chip 1, channel 0, earliest hit
…
data – chip 1, channel 0, latest hit
…
…
data – chip 1, channel 7, earliest hit
…
data – chip 1, channel 7, latest hit
…
…
…
header – chip 7, channel 0
data – chip 7, channel 0, earliest hit
…
data – chip 7, channel 0, latest hit
…
…
data – chip 7, channel 7, earliest hit
…
data – chip 7, channel 7, latest hit

Chip headers contain independent measurements of the Event Number and Trigger Time. To assure that all chips of the module stay correctly synchronized, the Event Number and Trigger Time should be monitored by the user.

In a ***future*** version of the firmware, chip headers after chip 0 will be suppressed from the data stream if their Event Number and Trigger Time are identical to the values of the preceding chip. Since the first chip header is always inserted into the data stream, the Event Number and Trigger Time information for each chip can be completely reconstructed.

Any difference in the Event Number among the chips indicates a serious error that requires a reset of the board. **Trigger Time differences of up to 1 count among the chips is acceptable.** (Note that for a 9-bit Trigger Time, 0 and 511 differ by 1.) For the Trigger Time, this assumes that an external SYNC_RESET signal has been successfully applied at the start of the run. <u>The user is strongly encouraged to monitor the Event</u>

Number and Trigger Time provided in these headers to assure synchronization across the entire system.

The data word transferred across the VME bus includes the 24-bit TDC word AND additional word type and F1TDC chip error information, as follows:

DATA - module data word (4 Mbyte address range, base programmed in register ADR32)

[31…27] – Word type

26 – F1 chip Resolution Locked Status

25 – F1 chip Output FIFO Overflow Status

24 – F1 chip Hit FIFO Overflow Status

[23…0] – F1 chip word, as described above

1.5 Data Word Categories

Data words from the module are divided into two categories: Data Type Defining (bit 31 = 1) and Data Type Continuation (bit 31 = 0). Data Type Defining words contain a 4-bit data type tag (bits 30 - 27) along with a type dependent data payload (bits 26 - 0). Data Type Continuation words provide additional data payload (bits 30 – 0) for the *last defined data type*. Continuation words permit data payloads to span multiple words and allow for efficient packing of some types of data. Any number of Data Type Continuation words may follow a Data Type Defining word.

Current Data Type List for F1TDC

0 – block header
1 – block trailer
2 – event header
3 – trigger time
4 – 6 – (undefined)
7 – time measurement data
8 – chip header
9 – 12 – (undefined)
13 – reserved
14 – data not valid (empty module)
15 – filler (non-data) word

Data Types

Block Header (0) – indicates the beginning of a block of events. (High-speed readout of a board or set of boards is done in blocks of events.)

(31) = 1
(30 – 27) = 0
(26 – 22) = slot ID (set by VME64x backplane)
(21 – 11) = number of events in block
(10 – 0) = event block number (used to align blocks when building events)

Block Trailer (1) – indicates the end of a block of events. The data words in a block are bracketed by the block header and trailer.

(31) = 1
(30 – 27) = 1
(26 – 22) = slot ID (set by VME64x backplane)
(21 – 0) = total number of words in block of events

Event Header (2) – indicates the start an event. The included trigger number is useful to ensure proper alignment of event fragments when building events. The 27-bit trigger number (134 M count) is not a limitation, as it will be used to distinguish events within event blocks, or among events that are concurrently being built or transported.

(31) = 1
(30 – 27) = 2
(26 – 0) = trigger number

Trigger Time (3) – time of trigger occurrence relative to the most recent global reset. Time is measured by a 40-bit counter that is clocked by the 31.25 MHz system clock. The global reset signal is distributed to every module in the system. The assertion of the global reset clears the counters and inhibits counting. The de-assertion of global reset enables counting and thus sets t = 0 for the component. The trigger time is necessary to ensure system synchronization and is useful in aligning event fragments when building events. With careful clock, trigger, and global reset distribution it may be possible to achieve identical trigger times from all components of the system. However, even if t = 0 is not the same for all components, *changes* in trigger times can be monitored to ensure system synchronization is maintained. The five bytes of the trigger time

$$\text{Time} = T_A\ T_B\ T_C\ T_D\ T_E$$

are reported in two words (Type Defining + Type Continuation):

<u>Word 1</u>:

(31) = 1
(30 – 27) = 3
(26 – 24) = reserved (read as 0)
(23 – 16) = T_C
(15 – 8) = T_D

(7 – 0) = T_E

Word 2:

(31) = 0
(30 – 16) = reserved (read as 0)
(15 – 8) = T_A
(7 – 0) = T_B

Time Measurement Data (7) – indicates an input channel has a hit within the defined time window relative to the trigger signal

(31) = 1
(30 – 27) = 7
(26) – F1 chip Resolution Status (1 = locked, 0 = unlocked)
(25) – F1 chip Output FIFO Overflow Status (1 = overflow, 0 = normal)
(24) – F1 chip Hit FIFO Overflow Status (1 = overflow, 0 = normal)
(23) = 1
(22) = 0
(21 – 19) – F1 chip number (0-7)
(18 – 16) – F1 channel number on chip (0-7)
(15 – 0) – time measurement

Chip Header (8) – contains information about the trigger received by the F1 chip

(31) = 1
(30 – 27) = 8
(26) – F1 chip Resolution Status (1 = locked, 0 = unlocked)
(25) – F1 chip Output FIFO Overflow Status (1 = overflow, 0 = normal)
(24) – F1 chip Hit FIFO Overflow Status (1 = overflow, 0 = normal)
(23) = 0
(22) = F1 chip Trigger FIFO Overflow Status (1 = overflow, 0 = normal)
(21 – 16) – F1 chip Trigger Number
(15 – 7) – F1 chip Trigger Time
(6) – Setup Register Tag (should NOT change during run)
(5 – 3) – F1 chip number (0-7)
(2 – 0) – F1 channel number on chip (0-7)

Data Not Valid (14) – module has no valid data available for read out.

(31) = 1
(30 – 27) = 14
(26 – 0) = undefined

Filler Word (15) – non-data word appended to the block of events. Forces the total number of 32-bit words read out of a module to be a multiple of 2 or 4 when 64-bit VME transfers are used. These words should be ignored.

(31) = 1
(30 – 27) = 15
(26 – 0) = undefined

1.6 Event Blocking

To increase readout performance the module packages events into blocks. The number of events in a block is defined by the BLOCK_SIZE register. Data from the block of events is bracketed by the Block Header and Block Trailer. Event Headers separate the data for events within the block.

For example, suppose BLOCK_SIZE = 3. Readout of the block will yield the following data stream:

```
Block Header
        Event Header (event 1 of block)
                Trigger Time – word 1
                Trigger Time – word 2
                Chip 0 header
                        Time measurement data – chip 0
                   . . .
                Chip 7 header
                        Time measurement data – chip 7
        Event Header (event 2 of block)
                Trigger Time – word 1
                Trigger Time – word 2
                Chip 0 header
                        Time measurement data – chip 0
                   . . .
                Chip 7 header
                        Time measurement data – chip 7
        Event Header (event 3 of block)
                Trigger Time – word 1
                Trigger Time – word 2
                Chip 0 header
                        Time measurement data – chip 0
                   . . .
                Chip 7 header
                        Time measurement data – chip 7
Block Trailer
```

Note: Filler Words may appear before or after the Block Trailer in the readout stream. These force the total number of 32-bit words read out to be a multiple of 2 or 4, depending on the VME readout mode used. Filler Words should be ignored.

1.7 Input Channel Mapping

In the above description, channel number means the chip channel number reported by the F1 chip itself. How this relates to the input channel number (front panel) depends on the module type (V2 or V3). The input channel numbers are numbered consecutively from bottom to top of the front panel: (0…31) for V2, (0… 47) for V3.

Recall (from the F1 chip documentation) that in normal resolution mode (V3), the chip provides 8 measurement channels (0…7), while in high resolution mode (V2), pairs of channels are combined to provide 4 measurement channels. In high resolution mode, reported channel numbers 0 & 1 refer to the first measurement channel; similarly, the pairs 2 & 3, 4 & 5, 6 & 7 refer to the remaining high resolution measurement channels.

The module itself performs the following re-mapping of the F1 chip channel numbers:
7 ->0, 6->1, 5->2, 4->3, 3->4, 2->5, 1->6, 0->7.

The USER should apply the following relationships (C-code) to obtain the front panel input channel number:

```
V3: input_channel = (chip_number <<3) | chip_channel;

V2: int channel_map[8] = {0, 0, 1, 1, 2, 2, 3, 3};
    input_channel = (4 * chip_number) + channel_map[ chip_channel ];
```