

This document details how the new zero-suppressed TDCs work...or at least how I (Kaz) believes it works...

The new TDC FPGA code written by Grass eliminates zeroes in the TDC data at the level of the TDC logic when data are written into the board during data-taking. During run1, this zero-suppression was performed in the VME CPU when CODA tried to read the TDCs at trigger. In essence, the CPU acted as a filter to removed the extra zeroes, at the cost of extra overhead in deadtime.

Some semantics:

SPILL – 5 seconds of beam ~once a minute

EVENT – number of triggers received by the DAQ. During run1, there were on the order of hundreds of EVENTS per SPILL.

HIT – number of particles seen by a particular scintillator in a hodoscope or wire in a wire chamber (each of which maps to a TDC channel). There may be multiple HITS in any element per EVENT.

TDC Configuration:

The new TDC has a circular ring buffer of 256 buffers for every 4 channels. Each buffer is 1 MB in size, and can record a HIT. The circular buffer can be split into 2, 4, or 8 segments, and the user can specify how many of these segments one wants to use for data-taking. The advantage of this segmentation is that if the number of HITS per EVENT is low, one can simply use say, segments 1 and 2 only, and reduce the copy-in-progress (CIP) time of the TDC. In this case, segments 1 and 2 will act as a circular buffer by themselves, while ignoring the rest of the segments.

Each buffer has a running clock which resets to zero once it reaches the Buffering Timing Window.

When a hit is received in a block, the current time on the clock is recorded. The TRIGGER is treated in a similar manner to a HIT, and occupies the last block of a particular EVENT. One tick of the clock is 0.44ns. Therefore, when one calculates $t_{TDC} = t_{TRIGGER} - t_{HIT}$, the resolution from both the TRIGGER and the HIT must be considered. The same goes for calculating the effects of differential non-linearity, since the timebins that the TRIGGER and HIT occupied may have different widths.

It is important to understand there are 2 timewindows in use. The Buffering Timewindow, and the Selection Timewindow. The Buffering Timewindow is set by the segmentation, and gives the full window within which the TDC values are recorded. The Selection Timewindow is a smaller window inside of the Buffering Timewindow within which the TDC values are read out. The Selection Timewindow is designed to reduce data volume in the event that we are able to narrow our signal to a range that is smaller than the Buffering Timewindow.

The specs of the TDC code for each segmentation setting can be found in Table 1.

Table 1

CSR [7:5]	Segmentation	Circular Buffer Size (MB)	Buffer Timing Window	CIP Time
000	8	32	512 ns	$32 \times 16 \times 16 \text{ns} = 8 \mu\text{s}$
001	4	64	1024 ns	$64 \times 16 \times 16 \text{ns} = 16 \mu\text{s}$
011	2	128	2048 ns	$128 \times 16 \times 16 \text{ns} = 32 \mu\text{s}$

The size and split of each 1MB buffer in the ring can be adjusted according to Table 2. This feature allows us to use the boards as multi-hit TDCs.

Table 2

CSR [2:0]	Number of Events	Max number of words/event
5	32	32
4	16	64
3	8	128
2	4	256
1	2	512
0	1	1024

Scalers:

Each hit seen by a TDC channel can be scaled in addition to having its time recorded. There 8 buffers available per channel. However, the 64 channels must be started/stopped/cleared in unison. Therefore, if one wants to stop and read channel 1, all 64 channels must be stopped. However, if the first 64-channel buffer is stopped, the second buffer can be invoked while the first buffer is being read and cleared.

Multiple Hit Elimination:

In many cases with our wire chambers and proportional tubes, one particle may create multiple ions along its track as it traverses the gas. Each of these ions may get attracted to a single wire, albeit with different times. The result is multiple HITS appearing in a single wire from one particle. The conventional means of eliminating this multiple HIT issues was to simply grab the first HIT from the train of pulses. However this elimination was performed at the analysis stage, after the data were written to disk. The multiple HIT elimination feature of the TDC allows one to inhibit for a user-defined time (in multiples of 4ns) after a hit has been seen in a particular channel. There are two inhibit modes. One is the "non-updating mode", where an input is inhibited for a set amount of time. The second is the "updating mode" where the inhibit-timer is refreshed every time a new HIT is seen during the inhibit timewindow. (i.e if HIT A creates a inhibit timewindow of 16ns, and if HIT B comes at 10ns, then not only is HIT B inhibited, but the inhibit timewindow will go on for another 16 seconds from the front edge of HIT B, for a total inhibit timewindow of 26ns, unless another HIT comes in and further extends the window).

Data Format:

The raw CODA data format has the following event types:

1. FEE information event
2. Physics event
3. Slowcontrol event
4. Beginning of Spill (BOS) event
5. End of Spill (EOS) event

Of these event types, the FEE information event and the Physics event are relevant to the TDCs.

FEE (Front-End Electronics) Information word.

The FEE information word is a flag that tells us what the TDC settings are in terms of timewindows used, multihit elimination on/off, etc. There is a FEE word available for each board.

- a. 0x84 = Event type
- b. 0xe906f011 = FEE ID
- c. 0xffff0BCD = TDC Registry word
 - o A: 0
 - o B: 0 = rising edge only, 1 = both rising and falling edges
 - o C: 0 = 8 segmentation, 2 = 4 segmentation, 6 = 2 segmentation
 - o D: Event buffer on Table 2. 0 = 1 event, 1 = 2 event, etc...
- d. 0x0BCD0000= TDC Registry word (pertains to the multihit elimination and scaler selection setting). CD controls the multihit elimination setting, while B controls the scaler setting.
 - o Multihit Elimination: 'CD' in hex. In Binary, that's 8 bits: abcdefgh
 - cdefgh goes from 000000 (0 in decimal) to 111111 (63 in decimal). The calculation of the length of the inhibit pulse is: $4ns \times (4 + \text{clock_cycle})$, where clock_cycle is the control value above in decimal notation (ranging from 0-63)
 - b: 0 is non-updating mode, 1 is updating mode
 - a: 0 = disable multihit elimination, 1 = enable multihit elimination
 - Example:
 - 'CD' = B3 (hex) is 10110011 in binary. This means **Enable multihit elimination, use non-updating mode**, and the **length of the inhibit pulse is 51**. The actual inhibit pulse in ns is then calculated as: $4ns \times (4+51) = 220ns$. The full length of the inhibit pulse can range from 16 to 268ns.
 - o Scaler selection status: 'B' in hex. In binary, that's 4 bits: abcd.
 - There are 8 scaler buffers in this version of the TDC.
 - Each bit above is set by the user to determine which scaler buffer is active and in use. Only one scaler buffer can be active at a time.

- Example:

If we use scaler buffer 5, then this will be 0101, and 'AB' = 05 (hex).

- e. This control word is the TDC selection timewindow. 0xABCD0EFG
 - EFG is the low limit of the selection window. That's 10 bits in binary. Conversion from binary to time is $4\text{ns} \times (\text{clock_cycle in decimal})$
 - BCD is the high limit of the selection window. That's 10 bits in binary. Conversion from binary to time is $4\text{ns} \times (\text{clock_cycle in decimal})$
 - A is selection window on or off. 0 or 8.
 - Example:
834800CC is selection window ON, 348 (hex) = 840 clock cycles = 3360ns, CC (hex) = 204 clockcycles = 816ns
 - If the lower limit is higher than the higher limit, there will be no data (there are no internal checks).

TDC Physics Event:

The data format for the new TDCs is considerably different compared to the old TDC code. In the new TDC, both the trigger time and the individual HIT times are recorded for each event. The TDC time can be calculated from the difference of these two values. In each event (CODA event), the header contains the trigger time.

In each TDC physics event, we have:

Word1: Number of words in event.

Word2: ROC ID

Word3: VXTicks

Word4: FEE Event

Word5: Board ID and number of words in TDC buffer. 0xabcd efgh. abcd refers to the board ID. efgh is the number of words in the TDC buffer until the next FEE event (new Board).

Word6: dummy word...exists solely to enable block transfer through DMA.

Word7: Header word gives trigger time. 0xabcd efgh, which is 32 bits (0-31).

- Bits 0-3 ('h'): Fine time. 0000 to 1001 (bin) , which is 0-9 (decimal).
Rising edge: $1-9 (\text{clock_cycle})$. $\text{FineTime} = 4\text{ns} - [444\text{ps} \times (\text{clock_cycle})]$
Falling edge: $1-8 (\text{clock_cycle})$. $\text{FineTime} = 500\text{ps} \times (\text{clock_cycle})$
- Bits 4-15 ('efg'): Rough time. 0 to 111111111111 (binary) , which is 0-4096 (decimal).
The RoughTime = $4\text{ns} \times \text{clock_cycle} (0-4096)$.
- The **trigger time = RoughTime + FineTime**
- Bit 16: always 1, since we always use the rising edge for the trigger. ('d')
- Bits 17-19: always 0. ('d')
- Bits 20 – 30: Total number of hits in event. ('abc')

- Bit 31: Always 1. Means this word has the trigger information. ('a')
- Example:
 $0x8a3130e7$
 $8a3(\text{hex}) = 1000\ 1010\ 0011\ (\text{bin}) = \text{Trigger data}$ with 163 hits to follow in this event.
 $1 = \text{Rising edge}$
 $30e = 782\ (\text{clock_cycles}) = 3128\ \text{ns}$
 $7 = 4\text{ns} - (7 \times 0.444) = 4\text{ns} - 3.108\ \text{ns} = 0.892$
 Trigger time = 3128.892 ns

Word8: Data Word. Header word gives data time. 0xabcd efgh, which is 32 bits (0-31).

- Bits 0-3 ('h'): Fine time. 0000 to 1001 (bin) , which is 0-9 (decimal).
 Rising edge: $1-9\ (\text{clock_cycle})$. $\text{FineTime} = 4\text{ns} - [444\text{ps} \times (\text{clock_cycle})]$
 Falling edge: $1-8\ (\text{clock_cycle})$. $\text{FineTime} = 500\text{ps} \times (\text{clock_cycle})$
- Bits 4-15 ('efg'): Rough time. 0 to 111111111111 (binary) , which is 0-4096 (decimal).
 The RoughTime = $4\text{ns} \times \text{clock_cycle}\ (0-4096)$.
- The **trigger time = RoughTime + FineTime**
- Bit 16: always 1, since we always use the rising edge for the trigger. ('d')
- Bits 17-19: always 0. ('d')
- Bits 20 – 23: Hit number of this event. ('c')
- Bits 24-27: Channel number. ('b')
- Bits 28-29: Cable number: ('a')
- Bit 30: Always 1. No reason. ('a')
- Bit 31: Always 0. Means this word has the data information. ('a')
- Example:
 $0x6a3120e7$
 $6a3(\text{hex}) = 0110\ 1010\ 0011\ (\text{bin}) = \text{Cable3 Channel10 Hit3}$
 $1 = \text{Rising edge}$
 $20e = 526\ (\text{clock_cycles}) = 2104\ \text{ns}$
 $7 = 4\text{ns} - (7 \times 0.444) = 4\text{ns} - 3.108\ \text{ns} = 0.892$
 Data time = 2104.892 ns

The clock is a ring, so the trigger time can appear to come before the data time. There are no flags indicating that the ring has reset. So the decoding algorithm must calculate both values and adjust for the ring buffer reset.