January 11, 2022

**AESOP-Lite DAQ Board**

**Event-PSOC Firmware Documentation**

## Channel Connections

Table 1. Calibration-pulse and PMT input connections, from left to right when looking down at the connector-end of the DAQ board. Note that the TOF chip SSN is 2. The SSN codes given here are bits 2,1,0. Bit 3 is always 1. The main PSOC, then, is addressed by 0000, and 1000 is “none”. Also note that Channel 5 (T2) has an external DAC and comparator, so the input to P2[0] is digital, whereas for the other channels it is analog. The four PSOC DACs are labeled by channel number. The guard signal, G, does not participate in the trigger logic but is discriminated and registered in the data.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Detector | T3 | G | T4 | T1 | T2 |
| PMT Connector | J10 | J2 | J17 | J15 | J25 |
| Calibration Connector | J12 | J11 | J18 | J16 | J26 |
| PSOC pin for preamp output | P4[6] | P3[2] | P3[3] | P4[5] | P2[0] |
| Schematic PSOC pin for preamp output | 9 | 16 | 15 | 14 | 19 |
| Channel | 2 | 1 | 4 | 3 | 5 |
| Time-of-Flight channel |  |  | 2 (B) | 1 (A) |  |
| Trigger bit | 1 | N/A | 0 | 3 | 2 |
| External SAR SPI SSN | 3 | 1 | 6 | 7 | 4 |

## Event Output Format

* 0xDC00FF
* 1-byte record length
* 0xDD (instead of the command echo), or 0xDB if TOF debug bytes are included
* 0x00 (zero length for the command-data echo)
* 0x5A45524F (“ZERO” in ASCII)
* 2-byte run number (MSBs first)
* 4-byte event number (count of accepted triggers)
* 4-byte time stamp (LSB increments each 5 milliseconds)
* 4-byte trigger count (includes triggers not accepted)
* 4-byte time and date
  + Bits 26-31: the year minus 2000
  + Bits 22-25: month (1-12)
  + Bits 17-21: day of the month (1-31)
  + Bits 12-16: hour (0-23)
  + Bits 6-11: minute (0-59)
  + Bits 0-5: second (0-59)
* 1-byte trigger status
  + Bit 0: primary trigger (electrons)
  + Bit 1: secondary trigger (typically without T2)
  + Bit 2: tracker trigger 0
  + Bit 3: tracker trigger 1
  + Bit 4: guard
* 2-byte T1 ADC counts
* 2-byte T2 ADC counts
* 2-byte T3 ADC counts
* 2-byte T4 ADC counts
* 2-byte G ADC counts
* 2-byte TOF time difference, in 10 ps units
* 2-byte tracker trigger count
* 1-byte tracker command count
* 1-byte status register and tracker trigger capture
  + Bits 6 and 7: tracker trigger capture
  + Bit 5: StoreADC
  + Bit 4: GO
  + Bit 3: Done
  + Bit 2: TOF interrupt
  + Bit 1: tcB
  + Bit 0: tcA
* Optional 10 bytes for debugging the TOF
  + 1-byte, number of TOF readouts since the last trigger in channel A
  + 1-byte, number of TOF readouts since the last trigger in channel B
  + 2-byte, channel A reference clock
  + 2-byte, channel B reference clock
  + 2-byte, channel A internal PSOC clock count
  + 2-byte, channel B internal PSOC clock count
* 1-byte, number of tracker boards in the readout
* Ordered sequence of tracker board data records of variable length
  + All installed boards are included unless the available space in the data record runs out
  + Individual boards come out with zero hits if
    - there really were no hits.
    - there was not enough space to write out the hits. These will have a bad CRC and an error bit set.
* 0x46494E49 (“FINI” in ASCII)

## Command Format

* Each command is formatted as "S1234<sp>xyW" repeated 3 times, followed by <cr><lf>
  + where 1234 are 4 ASCII characters, each representing a nibble
  + 12 gives us the data byte and 34 the address byte
* data byte: {7:0} gives the command code
* address byte: {7:6} and {1:0] give the number of data-byte "commands" to follow, 0 to 15
* {5:2} = 0x8 indicate the event PSOC
* All data arrive in up to 15 subsequent data-byte "commands" For those,
  + bits {7:0} of the command byte are the data for the command in progress
  + bits {7:6} and {1:0} give the data-byte number, 1 through 15
* Subsequent commands must wait until after the correct number of data bytes has arrived

## Command Codes

|  |  |  |  |
| --- | --- | --- | --- |
| **Code** | **Description** | **# data bytes out** | **Command Data bytes** |
| 0x01 | Load a PHA threshold DAC  CH 1-4: 8 bit DAC internal to the PSOC  CH 5: 12 bit external DAC | 0 | 0: PHA channel  1: High-order byte  2: Low-order byte (CH 5 only) |
| 0x02 | Read a PHA threshold DAC | 2 | 0: PHA channel |
| 0x03 | Read accumulated error codes | Variable | N/A |
| 0x04 | Load a 12-bit TOF threshold DAC | 0 | 0: TOF channel number  1: High-order byte  2: Low-order byte |
| 0x05 | Read a 12-bit TOF DAC | 2 | 0: TOF channel number |
| 0x06 | Turn red LED-2 on or off | 0 | 0: 1 for on, 0 for off |
| 0x07 | Return the code version number | 1 | N/A |
| 0x10 | Send arbitrary command to the tracker | Variable | 0: FPGA (0-7)  1: Tracker command code  2: Number of data bytes to follow  3 & up: Data bytes |
| 0x41 | Load a Tracker ASIC register | TKR echo | 0: FPGA (0-7)  1: Chip address (0-11)  2: Register type:  1 = calibration mask  2 = data mask  3 = trigger mask  3: Option  0 = mask selected channels off  1 = set selected channels on  4: Number of 2-byte items to follow:  5 and up: for each item  First byte = starting channel  Second byte = # of channels |
| 0x42 | Start a Tracker calibration sequence (pulses the channels selected by the calibration mask, using a pulse height set for the calibration DAC) |  | 0: FPGA (0-7)  1: Trigger delay  2: Trigger tag |
| 0x43 | Read a calibration event | Variable | 0: Trigger tag (must match 0x42 tag) |
| 0x0C | Reset the TOF chip | 0 | N/A |
| 0x0D | Modify the TOF chip configuration | 0 | 0: which byte to modify  1: new value for the selected byte |

## DAQ Timing

The timing sequences programmed into the PSOC firmware goes as follows:

1. **Trigger window:** in the Discriminator page of the schematic there is a component named TrigWindow\_V1. It contains a schematic with two components: a custom Verilog component named SignalCrop\_v1 and a 7-bit hardware counter Count7. The Verilog contains a 4-bit counter that determines how long the output will remain high when a rising edge is detected from the comparator that looks at the PHA channel preamp output. It is currently set at 14 clock counts with a 12 MHz clock, i.e. 1.17 microseconds. That determines the trigger window, because the trigger logic acts on the outputs, taking the AND of the 1.17 microsecond long pulses. Since it is a 4-bit counter, this window can be increased a bit, up to 1.25 μs by changing the parameter in the SignalCrop\_v1 component. Any more than that would require increasing the size of the counter (or lowering the clock frequency). This can only be changed by modifying the schematic and rebuilding.
2. **Preamp settling time:** after the end of the trigger window the Verilog waits for the comparator output to go low. It then starts the 7-bit hardware counter. The counter is started from a value set in software in the C code by the function *setSettlingWindow*, which is called by the command 0x3A. The default is set to 32 in the schematic but is overridden in the C code by an initial value of 24. When it has counted down to zero (a delay of 2 microseconds), then the Verilog code goes back to the state of looking for another rising edge. Thus the channel deadtime will last for the duration of the time-over-threshold of the preamplifier output plus another 2 microseconds (for the default). Only after that will the channel be ready to sense a new rising edge. The 2 microseconds is there to avoid retriggering on a wiggle back above threshold that might follow the preamplifier signal (especially with hysteresis set to zero for the comparators).
3. **PHA timing:** in hardware this is independent of the trigger timing but needs to accommodate it. In the ADC page of the schematic, the SAR\_ADC\_CTRL Verilog component starts acting when any of the channels goes above threshold. There is a 7-bit fixed-function hardware down-counter in the ADC schematic that is set in software to start from 32 but could be modified easily via command 0x4B (function *setPeakDetResetWait*). Once started, the Verilog waits for a GO signal until the Count7 counts down (1.3 microseconds for the 32 setting and 24 MHz clock). In principle it should wait for at least as long as the trigger window (plus a small logic latency), to be sure never to miss a coincidence for the GO. If it doesn't get a GO, then it waits for the OR of the channels to go back down plus another count-down of the 7-bit counter, while holding high the reset signal for the peak detectors, before it starts looking for the OR of channels to go high again. If it does get a GO, then it sends a signal to the ADCs to digitize. It then waits for the 7-bit counter to count down to zero again, and then it sends a signal to the CPU that the digitization is done. After that it waits for the OR of channels to go low and then holds the reset high on the peak detectors for a full down-count of the 7-bit counter. The readout of the ADCs is initiated by the CPU after it has received the “done” signal. Note that this Verilog component uses the 7-bit hardware counter for multiple purposes, in order to minimize the number of flip-flops needed (since the PSOC does not have large number of them, unlike an FPGA). It needs to count long enough to cover the trigger window, to give the ADCs enough time to digitize, and to give the peak detectors enough time to drain their charge through the MOSFET switch.
4. **Deadtime:** there are several contributors to the trigger deadtime. First, the individual channels do not recover immediately if they go above threshold, because the preamp must first decay back below threshold, then then the counter has to tick off the settling time, as explained above, before the channel can accept another rising edge. Second, every time any channel goes above threshold, if there is no trigger “GO”, then the system waits for twice the *setPeakDetResetWait* time described above, i.e. first looking for the GO and second to reset the peak detectors. That is a total of 2.6 microseconds, which may not be acceptable if the channel singles rate is too high, in which case the delay setting can be reduced (or the thresholds raised). Third, if there is a trigger “GO”, then the trigger gets disabled for the duration of the event readout. First, the ADCs are read out, the Tracker is readout out, the TOF is read out, and then all the data are formatted and shipped off to the main PSOC. This takes much longer than the above mentioned hardware deadtime components, but it happens only at the trigger rate, not the OR of the singles rates. It is easy to monitor the readout contribution to the deadtime, because we count the triggers that arrive and not accepted pending the readout. One can estimate the other contributions from the singles rates that we monitor for each PMT channel.
5. **Trigger Status Capture:** Status\_Reg\_Trg captures the status of the various triggers each time there is a “GO”. The status is held by set-reset latches, one for each PMT trigger, one for each Tracker view, and one for the guard. The latches are reset every time there is a rising edge of the OR from the five comparators. After that, any signal above threshold will set them. The status register then needs to be readout before another channel-OR comes along and clears the latches. It is read first thing by the interrupt service routine of the “GO” signal, which has the highest priority of all interrupts. Note that the reset time of the peak detectors prevents a new channel-OR from happening for that period of time.

## Time of Flight

The time difference is calculated in units of 10 ps as (channel T4 minus channel T1). Each of the two “stop times” is measured independently from the other and also independently from the instrument trigger. The times correspond to threshold crossings of fast discriminators that view the raw PMT signals prior to amplification by the preamplifiers of the analog PHA chain. A 51-ohm resistor separates the point where the TOF voltage is measured from the preamplifier input, so the negative-going signal is given by the PMT current times that resistance (plus the frequency-dependent input impedance of the non-ideal charge amplifier). The threshold voltage is set independently for each channel by 12-bit DACs (and inverted by op-amps to be negative).

The specialized 2-channel timing chip has to be set up prior to the run by loading its volatile configuration register from the Event PSOC via SPI. The SPI is not used to read out the resulting data. Instead, the data flow out of the chip on LVDS lines and are captured in PSOC 32-bit shift registers. There are two options for reading the shift registers. The first, which was the first developed, interrupts the CPU for every stop event of either channel, after which the interrupt service routine reads the register contents into a large FIFO buffer. The second sends each stop event by DMA to an intermediate buffer than can hold up to about 24 events (limited by the number of DMA “transaction descriptors” available). When the intermediate buffer is full, then the CPU is interrupted to transfer the buffer to the large FIFO. In both cases, the PSOC clock is captured, to 5 ms resolution, along with the TOF chip time data. The DMA method should be preferable, as it results in fewer CPU interrupts, which in excess could bog down the CPU, depending on the noise rate.