HITSUM Logic Functional Description

Hai Dong, Ed Jastrzembski (3/15)

<u>**Overview**</u>

The HITSUM logic implemented in the FPGA provides the user with a flexible mechanism to define a logic pulse based on the pattern of signals at the module's 16 inputs OR on the total digital sum of enabled channels (hence the term HITSUM). The logic pulse can directly trigger the module for readout, or it can be driven out and used to trigger a set of modules. With external logic, trigger signals from multiple FADC modules can be combined to form a composite trigger that initiates readout of modules (FADC + other).

We first discuss a trigger based on the pattern of signals at the module's 16 inputs.

A Hit Bit for a channel is a logic pulse generated when an input signal's digitized data crosses the threshold programmed for the channel. The width of the logic pulse is programmable in units of the clock period (4ns for 250 MHz). The behavior is that of a discriminator, but is based on the digitized data of the channel rather than the input analog signal. The threshold that defines the logic pulse is the same as that used in the data processing /readout path. The Hit Bit pulse can be delayed; the delay is programmed in units of the clock period. This allows the relative timing of the channels to be adjusted.

Each of the 16 channels can produce a Hit Bit. The pattern of Hit Bits, called the Hit Bit data, is 16 bits wide.

There are three modes available to process the Hit Bit data.

In <u>Boolean Overlap Mode</u>, specific input channels selected by the user are required to have their Hit Bits overlap for a logic pulse to be generated. Hit Bit data is updated every clock period. When the logic is enabled and when the Hit Bit data contains Hit Bits from the selected channels, a pulse (*Live Trig*) is generated. The width of *Live Trig* is programmed by the user. While *Live Trig* is asserted, further pulse generation is suppressed.

In <u>Table Mode</u>, the Hit Bit data is mapped directly to the address of a 65536 x 1 RAM. The RAM is programmed by the user. Addresses programmed with a '1' correspond to Hit Bit patterns that result in the generation of a logic pulse. Hit Bit data and the output of the RAM are updated every clock period. When the logic is enabled and a '1' appears at the RAM output, a pulse (*Live Trig*) is generated. The width of *Live Trig* is programmed by the user. While *Live Trig* is asserted, further pulse generation is suppressed.

In <u>Window Mode</u>, the 65536 x 1 RAM is used in a more sophisticated way. Input channels are selected by the user to have the ability to generate a *hit window* when a Hit Bit occurs on them. If multiple Hit Bits occur, the hit window is initiated by the earliest one. The hit window has a programmable width. Hit Bits that occur at any time during the hit window are used to form the RAM address. The Hit Bits do not have to overlap each other to be included in the collective hit pattern; they only need to overlap the hit window. The address is applied to the RAM at the end of the hit window. The RAM is programmed by the user. Addresses programmed with a ‘1’ correspond to collective Hit Bit patterns that result in the generation of a logic pulse (*Live Trig*) when the logic is enabled. The width of *Live Trig* is programmed by the user. While *Live Trig* is asserted, hit window generation and pulse generation is suppressed.

A trigger from the module can instead be based on the total digital sum of enabled channels. Sum Data is the sample-by-sample sum of the 16 input channels. If a channel is disabled it contributes exactly zero to the sum. Sum Data is updated every clock period. The 16-bit Sum Data is compared to a threshold programmed by the user. When the sum is above this threshold and the logic is enabled, the pulse *Live Trig* is asserted. While *Live Trig* is asserted, further pulse generation is suppressed.

Processing of the 16-bit Hit Bit Word

1. The bits are active Hi. The rising edge triggers a pulse generator (one shot) of user programmable width. A programmable delay (4 ns steps) for each input allows the user to adjust timing for coincidence formation.

2. Three modes of operation are available: Window Mode, Boolean Overlap Mode, and Table Mode.

3. Window Mode (Figure 1):
    a. Bits are selected by the user that can generate a Hit Window. When multiple selected bits occur, the earliest one starts the window generation.
    b. The width of the Hit Window is programmed by the user. The window has a minimum width of 4 ns and maximum width of 262.14 uS (16-bit word).
    c. Hit Bits that occur while the Hit Window is active are latched. At the end of the hit window this latched Hit Bit pattern is presented as an address to a 65536 x 1 RAM. If the latched pattern had been programmed as a ‘1’ in the RAM, a Window Mode pulse is generated.
    d. A fixed-width pulse *Live Trig* is generated from the leading edge of the Window Mode pulse. The width of the *Live Trig* pulse is programmable from 4 ns to 262.14 us.

4. Boolean Overlap Mode (Figure 2):
    a. Bits are selected (Overlaped Selected Bits) by the user for coincidence.
    b. An Overlapped Mode pulse is generated whenever there is an overlap of the Selected Bits. The pulse width is the same as the duration of the overlap.
    c. A fixed-width pulse *Live Trig* is generated from the leading edge of the Overlap Mode pulse. The width of the *Live Trig* pulse is programmable from 4 ns to 262.14 us.

5. Table Mode
    a. In this mode, the Hit Bit data directly addresses the 65536 x 1 RAM. Hit Bit data is updated on each clock period. If the Hit Bit pattern has been programmed as a ‘1’ in the RAM, a Table Mode pulse is generated.
    b. A fixed-width pulse *Live Trig* is generated from the leading edge of the Window Mode pulse. The width of the *Live Trig* pulse is programmable from 4 ns to 262.14 us.

Figure 1: Window Mode

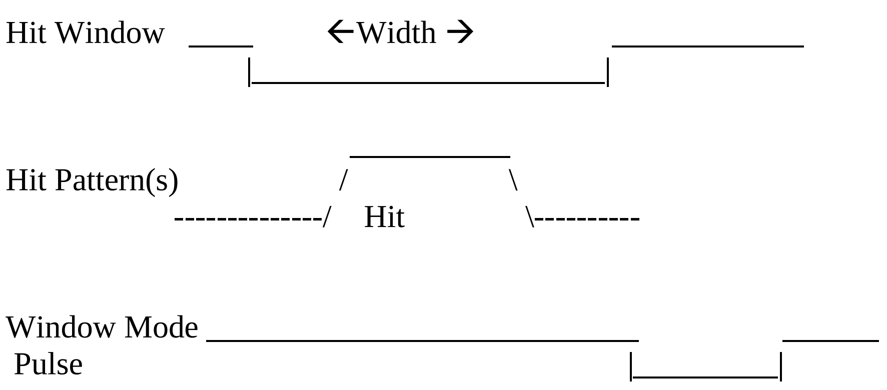
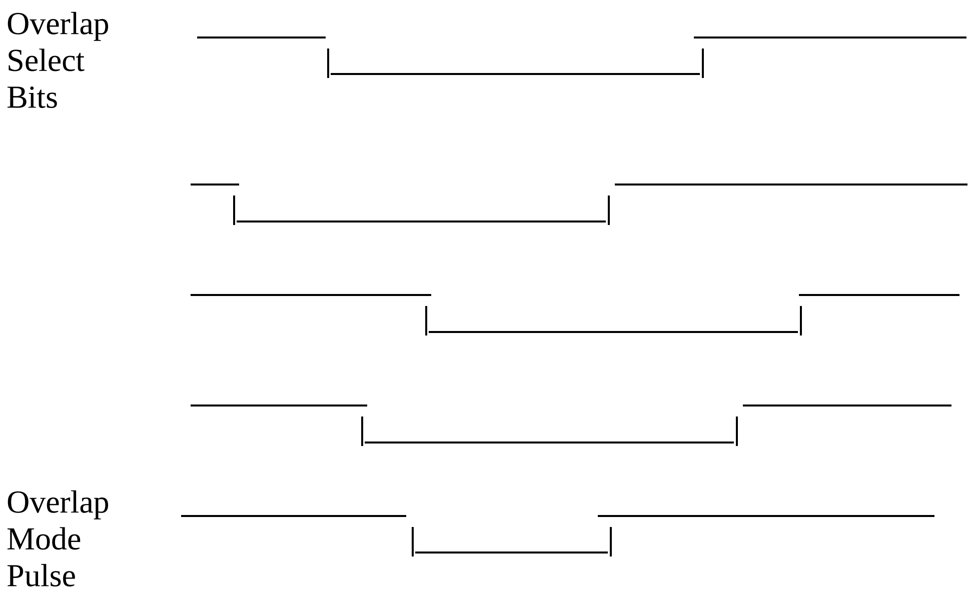


Figure 2: Boolean Overlap Mode



<u>**SUM Processing:**</u>

Since the sixteen ADC on the FADC board each produce 12-bit data, Sum Data is chosen to be a 16-bit quantity. Whenever Sum Data is above a programmed threshold, signal *Sum Trig* is asserted high. When the SUM mode is selected for the module, the fixed-width pulse *Live Trig* is generated from the leading edge of *Sum Trig*.

<u>**Processing of Live Trigger**</u>

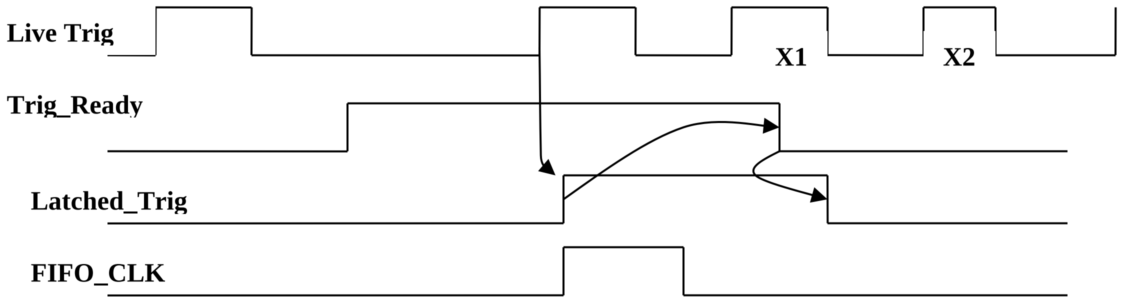
If the user does not want *Live Trig* to internally trigger the module, *Live Trig* can be driven out of the module. It may then be combined with other signals to form a trigger for the modules of the system.

When the self-trigger option *is* desired, *Live Trig* must be processed before being used by the module. The instantaneous rate of *Live Trig* depends on the rates of the input signals, the levels of the input thresholds, and how the user programs the HITSUM logic. The ADC processing algorithms require that there be a minimum separation of trigger signals (~100 ns), and limits the number of triggers that can occur within a given time period. The latter limits depend on the specific ADC processing mode and the detailed processing parameters used. Failure to respect these restrictions can result in data corruption and/or algorithm termination. When the FADC module is part of a system, the restrictions can be enforced at the top of the trigger distribution chain (Trigger Supervisor or Trigger Interface). For the self-trigger option we must have an on-board mechanism to enforce these restrictions.

The *Trig Ready* signal is the means by which we control the acceptance of *Live Trig* by the module (see <u>Figure 3</u>). While *Trig Ready* is low, all *Live Trig* pulses produced by the HITSUM logic are ignored. The user can force *Trig Ready* to be low; this is done while the user configures the module. Once triggers are enabled by the user, the state of *Trig Ready* is controlled by conditions within the module.

The interplay of *Live Trig*, *Trig Ready*, and *Latched Trig* signals are shown in <u>Figure 3</u>. When *Trig Ready* is high, the leading edge of a *Live Trig* pulse causes the assertion of *Latched Trig*. *Latched Trig* indicates that a trigger has been accepted by the module. The Hit Bit data pattern or Data Sum that caused the *Live Trig* to be issued is stored in a FIFO that can be read by the user. *Trig Ready* responds to the assertion of *Latched Trig* by going low; *Latched Trig* responds by going low also. Note that while *Latched Trig* is high, no *Live Trig* can be accepted. Requiring that *Trig Ready* be asserted when the <u>leading edge</u> of *Live Trig* occurs preserves the inherent timing of the trigger signal.

Figure 3: Internal Trigger Timing diagram



Trigger X1 is ignored because present trigger process is not done.
Trigger X2 is ignored because Trig Ready is low

Latched Trig is used to generate *Module Trigger*, the pulse that activates the ADC processing algorithms. *Module Trigger* has two parameters that are specified by the user: width and hold off period. (See INTERNAL TRIGGER CONTROL register.) *Trig Ready* cannot be asserted again until the width and hold off periods expire. Programming values such that **width + hold off > 100 ns** assures that the ADC processing algorithm's minimum separation of triggers requirement is satisfied.

Trig Ready may also be held low because too many accepted triggers are currently being handled by the ADC processing algorithm. Each accepted trigger is counted; an accepted trigger is considered *acknowledged* when all data associated with it has been transferred to the Control FPGA. To ensure that data buffers used by the ADC processing algorithm are not overrun, the number of *unacknowledged* triggers is continuously compared to a level (MAX2) programmed by the user. (See TRIGGER CONTROL register.) *Trig Ready* is held low when the number of unacknowledged triggers ≥ MAX2. MAX2 is based on the parameters (processing mode, # samples in window) loaded into the ADC processing algorithm. (See Appendix 1 for determining values for MAX2.)

Trig Ready may also be held low because on-board storage is nearly exhausted. *Trig Ready* is forced low whenever the number of 8-byte words in the external RAM is within 12K (1.2%) of memory capacity (i.e. 1,036,288).

<u>Appendix 1</u>- Maximum number of Unacknowledged Triggers (MAX2)

<u>Definitions</u>

Mode 1: Raw
Mode 2: Pulse Raw
Mode 3: Integral
Mode 4: TDC
Mode 7: Integral + TDC
Mode 8: Raw + TDC

PTW = number of samples in trigger window
NSB = number of samples before threshold crossing included in pulse (modes 2,3,7)
NSA = number of samples after threshold crossing included in pulse (modes 2,3,7)
NumberOfPulses = maximum number of pulses to find in window (modes 2,3,7)

Mode 1: X = 2040 / (PTW + 7)

Mode 2: X = 2040 / ((2 + NSA + NSB) * NumberOfPulses)

Mode 3: X = 2040 / (5 * NumberOfPulses)

Mode 4: X = 2040 / (4 * NumberOfPulses)

Mode 7: X = 2040 / ((5 * NumberOfPulses) + (4 * NumberOfPulses))

Mode 8: X = 2040 / ((PTW + 7) + (4 * NumberOfPulses))

MAX2 = minimum (9, X)