Implementation of QT Algorithm for ZDC
Run 10 – Au+Au
qt32b_l0_v5_e.mcs

Chris Perkins

12/22/2009

QT Code Version: 0x5e

Description:

This algorithm compares the analog East and West Sums to three different thresholds, compares the Attenuated East + West analog Sum to one threshold, and outputs the E1TAC and W1TAC signals if they are within some range.

This algorithm does **not** use the "Good Hit" requirements that were used last year.

The E1TAC signal (Daughter A, ch4) is passed on to L0 if:

TAC_MIN < E1TAC < TAC_MAX

Otherwise, 0x000 is passed on to L0 for E1TAC. An equivalent condition is required to pass on W1TAC (Daughter C, ch4).

The ESum, WSum and E+WASum threshold bits have no requirements on their TAC signals; a threshold bit is '1' if the corresponding channel is above the corresponding threshold with no other requirements.

Inputs:

QT8A: ESum (ch2), E1TAC (ch4)
QT8B: E+WASum (ch 11)
QT8C: WSum (ch18), W1TAC (ch20)
QT8D: None

Registers (1 Set Per Daughter Card)**:**

Alg. Reg. 0 (Reg 13): Not Used
Alg. Reg. 1 (Reg 14): TAC_MIN
Alg. Reg. 2 (Reg 15): TAC_MAX
Alg. Reg. 3 (Reg 16): Analog E/W Sum Threshold 0 (Or E+WASum Threshold)
Alg. Reg. 4 (Reg 17): Analog E/W Sum Threshold 1
Alg. Reg. 5 (Reg 18): Analog E/W Sum Threshold 2
Reg. 11: Channel Mask

LUT:

Timing adjustments/pedestal subtraction for each PMT

Algorithm Latch: ??

Action (21x RHIC Clock)**:**

```
1st:  Mask channels and Latch inputs
         If mask bit = 1, channel data = 0

2nd:  For first TAC channel of each daughter (ch4):
            TAC above threshold: TAC > TAC_MIN → Good_TAC_MIN
            TAC below threshold: TAC < TAC_MAX → Good_TAC_MAX
            (Note: This result is ignored on daughters B,D)
      Compare third ADC channel (ch2) to three thresholds (for E/W Sum Ths)
            → ADC_th0, ADC_th1, ADC_th2
             (Note: This result is ignored on daughters B,D)
      Compare fourth ADC channel (ch3) to first threshold (for E+W ASum Th)
            → ADC_EW_th
            (Note: This result is ignored on daughters A,C,D)

3rd:  Check TAC range:
            if (Good_TAC_MIN && Good_TAC_MAX)
                  First TAC channel (ch4) → TAC_OUT
            else
                  0x000 → TAC_OUT
      Delay Threshold Bits

4th:  Latch Output Bits to next daughter or L0 FPGA
         if(daughter A)
            (0-11)   :   Passed from previous daughter
            (12-23)  :   TAC_OUT   (E1TAC)
            (24-26)  :   Passed from previous daughter
            (27)     :   ADC_th0    (ESum > th0)
            (28)     :   ADC_th1    (ESum > th1)
            (29)     :   ADC_th2    (ESum > th2)
            (30)     :   Passed from previous daughter
            (31-33)  :   ‘0’
         else if(daughter B)
            (0-11)   :   Passed from previous daughter
            (12-23)  :   Passed from previous daughter
            (24-26)  :   Passed from previous daughter
            (27-29)  :   Passed from previous daughter
            (30)     :   ADC_EW_th    (E+W ASum > th0)
            (31-33)  :   ‘0’
         else if(daughter C)
            (0-11)   :   TAC_OUT    (W1TAC)
            (12-23)  :   Passed from previous daughter
            (24)     :   ADC_th0    (WSum > th0)
            (25)     :   ADC_th1    (WSum > th1)
            (26)     :   ADC_th2    (WSum > th2)
            (27-29)  :   Passed from previous daughter
```

```
        (30)      :   Passed from previous daughter
        (31-33)  :   ‘0’
    else if(daughter D)
        (0-11)    :   Passed from previous daughter
        (12-23)  :   Passed from previous daughter
        (24-26)  :   Passed from previous daughter
        (27-29)  :   Passed from previous daughter
        (30)      :   Passed from previous daughter
        (31-33)  :   ‘0’
```

L0 Output to DSM:

(0-11)	:	W1TAC (if within range)
(12-23)	:	E1TAC (if within range)
(24)	:	WSum > Th0
(25)	:	WSum > Th1
(26)	:	WSum > Th2
(27)	:	ESum > Th0
(28)	:	ESum > Th1
(29)	:	ESum > Th2
(30)	:	E+W ASum > Th0
(31)	:	‘0’