

Algorithms for Vertex QT-DSM Tree RHIC 2015 p+Au Run

Eleanor Judd

June 5, 2015

Change Log:

| Date | Description |
|--------------|--|
| May 4, 2015 | First version for the 2015 p+Au data taking. The BBC and VPD boards are using their standard algorithms. The ZDC QT board (ZD001) and DSM board (ZD101) are using the heavy-ion versions of their algorithms. The VT201 algorithm has been changed to remove the Preceded and BBC-Large logic and replace it with the ZDC thresholds. |
| May 26, 2015 | A new version of the ZD101 algorithm has been made to match the input from the hybrid ZDC QT algorithm. In the hybrid logic the ZDC East signals are processed using the heavy ion logic and the ZDC West signals are processed using the proton logic. The ZD101 DSM algorithm has also been simplified. The ZDC deadtime logic has been deleted because it is not currently in use and the scaler output bits have been reduced to the set that are currently used by the scaler system. The output of the new ZD101 algorithm has been arranged to match the input of the existing VT201 algorithm. |
| June 5, 2015 | The VT201 algorithm has been re-implemented using new more conservative software, which fixes the pass-through of the ZDC-W-Back bit. Also the description of the ZDC logic has been updated to make it clear which threshold bits are based on digital sums and which are based on analog sums |

The Vertex branch of the DSM tree is used to locate the primary vertex of the RHIC beam collisions at STAR. All three relevant trigger detectors connect to this branch: Zero Degree Calorimeters (ZDC), Beam-Beam Counters (BBC) and the Vertex Position Detector (VPD). The raw detector signals are digitized and pre-processed in QT boards. The DSM tree is then used to calculate TAC differences and combine ADC information.

Layer 0 QT Boards: BBQ_BB001:002

There are two BBC small-tile QT boards: one processes data from the East side of the detector and the other from the West side. The algorithm has been changed in 2014 to add a slewing correction to the original logic. Please see the documentation provided by Chris Perkins for a detailed description of the new algorithm at

http://www.star.bnl.gov/public/trg/TSL/Software/qt_v6_d_doc.pdf

The algorithm still forms a 16-bit ADC Sum and 12-bit TAC Max. Only channels that satisfy a “good hit” requirement are included in the ADC Sum and TAC Max. A “good hit” is defined as one where the ADC value is greater than some threshold and the corresponding TAC value is greater than TAC_MIN and less than TAC_MAX. The channel mask register can be used to force the algorithm to ignore certain channels, but note that ADC and TAC channels must each be masked individually. The old algorithm is documented at

http://www.star.bnl.gov/public/trg/TSL/Software/qt_v5_6_doc.pdf

The slow correction logic is the same logic that was added to the MTD QT boards in 2013, and is documented at http://www.star.bnl.gov/public/trg/TSL/Software/qt_v6_c_doc.pdf

1. Layer 1 DSM Boards: BBC_BB101

The BB101 DSM board processes data from the BBC-small-tile detector. The algorithm receives ADC-sum and fastest-TAC data from the QT boards. The ADC sums are compared to thresholds. A set of bits specified by the user is chosen from each incoming TAC value to send to the scaler system. In parallel, the TAC difference is calculated. The difference is set to zero if either of the two incoming TACS is zero, because a TAC value of zero implies there were no good hits on that side of the BBC.

RBT File: bbc_bb101_2009_a.rbt

Users: BB101

Inputs: Ch0/1 = QT Board BB001 (East)
Ch2/3 = QT Board BB002 (West)
Ch4/7 = Unused

From each QT board:
bits 0:15 = ADC-Sum
bits 16:27 = Max TAC (Value of zero implies NO good hits)

LUT: 1:1

Registers:

Four registers, all thresholds can be set independently
R0: BBCsmall-EastADCsum_th (16 bits)
R1: BBCsmall-WestADCsum_th (16)
R2: BBCsmall-EastTAC-select (3)
0 => select bits 0:6
1 => select bits 1:7
...
5 => select bits 5:11

R3: BBCsmall-WestTAC-select (3)
Same value definitions as for R2

Action:

- 1st Latch input
- 2nd Compare each ADC-sum to its threshold
Calculate: TAC difference = $4096 + \text{TAC-E} - \text{TAC-W}$
Define: Good-TAC-E = $\text{TAC-E} > 0$, same for West side
Make all possible bit selections from TAC-E and TAC-W, including overflow logic. For example:
 TAC-E-overflow-0 = TAC-E(7), (8), (9), (10) or (11)
 If (TAC-E-overflow-0 = 1) then TAC-E-scaler-0 = 127
 Else TAC-E-scaler-0 = TAC-E(0:6)
Same logic for all possible bit selections from TAC-E (see description of register R2) and TAC-W
- 3rd Delay ADC-sum threshold bits
Zero out TAC difference if either Good-TAC-E or Good-TAC-W is false, otherwise just delay TAC difference
Use R2 to select the TAC-E scaler bits:
 If (R2 = 0) then chose TAC-E-scaler-0
 Else if (R2 = 1) then chose TAC-E-scaler 1
 Etc...
Do the same for West side, using R3 to control the selection.
- 4th Latch output

Output to VT201:

- (0-12) TAC difference
- (13) Unused
- (14) ADC-sum-E > th0
- (15) ADC-sum-W > th0

Scalers:

- (0-6) selected bits of TAC-E
- (7-13) selected bits of TAC-W
- (14) ADC-sum-E > th0
- (15) ADC-sum-W > th0

2. Layer 0 QT Boards: BBQ_BB003

There is just one BBC large-tile QT board and it receives data from both the East and West sides of the detector. The algorithm was written by Chris Perkins and is documented at http://www.star.bnl.gov/public/trg/TSL/Software/qt_v5_f_doc.pdf

3. Layer 1 DSM Board: BBC_BB102

The BB102 DSM board processes data from the BBC-large-tile detector. The algorithm receives a hit flag and fastest-TAC data for each of the East and West sides of the detector

from the QT board. The hit flags indicate there was at least one good hit on each side, and they are just passed through to the output. A set of bits specified by the user is chosen from each incoming TAC value to send to the scaler system. In parallel, the TAC difference is calculated. The difference is set to zero if either of the two incoming TACS is zero, because a TAC value of zero implies there were no good hits on that side of the BBC.

RBT File: bbc_bb102_2010_b.rbt

Users: BB102

Inputs: Ch0/1 = QT Board BB003 (East and West)
Ch2/7 = Unused

From the QT board:
bits 0:11 = MAX TAC East (value of zero implies no good hits)
bits 12:23 = MAX TAC West
bit 24 = East hit
bit 25 = West hit

LUT: 1:1

Registers:

R0: BBCLarge-EastTAC-select (3)
0 => select bits 0:6
1 => select bits 1:7
...
5 => select bits 5:11
R1: BBCLarge-WestTAC-select (3)
Same value definitions as for R0

Action:

- 1st Latch input
- 2nd Delay hit bits to 4th step
Calculate: TAC difference = 4096 + TAC-E – TAC-W
Define: Good-TAC-E = TAC-E > 0, same for West side
Make all possible bit selections from TAC-E and TAC-W, including overflow logic. For example:
TAC-E-overflow-0 = TAC-E(7), (8), (9), (10) or (11)
If (TAC-E-overflow-0 = 1) then TAC-E-scaler-0 = 127
Else TAC-E-scaler-0 = TAC-E(0:6)
Same logic for all possible bit selections from TAC-E (see description of register R0) and TAC-W (see register R1)
- 3rd Zero out TAC difference if either Good-TAC-E or Good-TAC-W is false, otherwise just delay TAC difference to the 4th step
Use R0 to select the TAC-E scaler bits:
If (R0 = 0) then chose TAC-E-scaler-0
Else if (R0 = 1) then chose TAC-E-scaler-1
Etc...
Do the same for West side, using R1 to control the selection.

4th Latch output

Output to VT201:

(0-12) TAC difference
(13) Unused
(14) East hit
(15) West hit

Scalars:

(0-6) selected bits of TAC-E
(7-13) selected bits of TAC-W
(14) East hit
(15) West hit

4. Layer 0 QT Boards: BBQ_VP001:002

The two VPD QT boards use the same algorithm as is used by the two small-tile BBC QT boards. See documentation above for BBQ_BB001:002.

5. Layer 1 DSM Board: BBC_VP101

RBT File: bbc_vp101_2009_a.rbt

Users: VP101

Inputs: Ch0/3 = Unused
Ch4/5 = QT Board VP003 (East)
Ch6/7 = QT Board VP004 (West)

The VP101 DSM board receives VPD data from 2 QT boards. The logic needed to do this analysis is the same as that used by the BB101 algorithm. The VP101 algorithm is therefore identical to the BB101 algorithm in every way, except for the input map. Please see the BBC_BB101 documentation above for details of the logic.

6. Layer 0 QT Board: BBQ_ZD001

The single ZDC QT board receives signals from both the East and West sides of the ZDC. The algorithm can be configured to compare the ZDC-Front and ZDC-Back ADC values, and their digital sum, to a threshold (the “proton” logic) or to compare the analog sum to multiple thresholds (the “heavy ion” logic). It should be noted that the proton logic uses the “good hit” requirement for all hits (ADC > threshold and associated TAC in window) but the heavy ion logic does not. In the current setup the East side is configured to use the heavy-ion logic and the West side is configured to use the proton logic. The algorithm was written by Chris Perkins and is documented at http://www.star.bnl.gov/public/trg/TSL/Software/qt_v6_f_doc.pdf

7. Layer 1 DSM Board: BBC_ZD101

The ZD101 DSM board processes data from the ZDC detector. The algorithm receives TAC data from the ZD001 QT board. The TAC difference is calculated. The difference is set to zero

if either of the two incoming TACS is zero, because a TAC value of zero implies there were no good hits on that side of the ZDC. A user-specified set of bits is then chosen to be passed on to VT201. The algorithm also receives various threshold bits from the East and West sides of the detector. The East threshold bits were produced using the heavy-ion QT logic which does NOT include a “good hit” requirement. They are therefore zeroed out if the East TAC value is zero. The West threshold bits were produced using the proton QT logic, which DOES include a “good hit” requirement. Those bits are therefore just passed through to the output unmodified

RBT File: bbc_zd101_2015_a.rbt

Users: ZD101

Inputs: Ch0/1 = QT Board ZD001
Ch2:7 = Unused

From the QT board:
bits 0:9 = West-1 TAC
bits 10:19 = East-1 TAC
bits 20:22 = Truncated Total West sum (Unused in this algorithm)
bit 23 = Front West ADC > threshold
bit 24 = Back West ADC > threshold
bit 25 = West ADC digital sum > threshold
bit 26 = East analog sum > threshold-0
bit 27 = East analog sum > threshold-1
bit 28 = East analog sum > threshold-2
bit 29 = East analog sum > threshold-3
bit 30 = East attenuated analog sum > threshold-4 (Unused in this algorithm)
bit 31 = East attenuated analog sum > threshold-5 (Unused in this algorithm)

LUT: 1:1

Register:

R0: ZDC-TACdiff-select (2 bits)
0 => select bits 0:8
1 => select bits 1:9
2 => select bits 2:10

Action:

- 1st Latch input
- 2nd Delay the 4 East threshold bits to the 3rd step.
Delay the 3 West threshold bits directly to the 4th step.
Calculate: TAC difference = 1024 + TAC-E – TAC-W
Define: Good-TAC-E = TAC-E > 0, same for West side
- 3rd Combine the 4 East threshold bits with the Good-TAC-E bit, e.g.:
Sum-E-th0 = East-th0 and Good-TAC-E, etc...
Use R0 to select the TAC difference bits, including overflow logic and the “good” TAC cut, i.e.:
Diff-overflow-0 = TAC-diff(9) or (10)
Diff-overflow-1 = TAC-diff(10)

```

If (Good-TAC-E = 0 or Good-TAC-W = 0) then output = 0
Else if (R0 = 0)
    If (Diff-overflow-0 = 1) then output = 511
    Else output = TAC-diff(0:8)
Else if (R0 = 1)
    If (Diff-overflow-1 = 1) then output = 511
    Else output = TAC-diff(1:9)
Else
    Output = TAC-diff(2:10)

```

4th Latch output

Output to VT201:

```

(0-8) TAC difference
(9) Digital-Sum-W > threshold
(10) Front-W > threshold
(11) Back-W > threshold
(12) Analog-Sum-E > th0
(13) Analog-Sum-E > th1
(14) Analog-Sum-E > th2
(15) Analog-Sum-E > th3

```

Scalers:

```

(0-9) Unused
(10) Digital-Sum-W > th
(11) Digital-Sum-W > th
(12) Analog-Sum-E > th0
(13) Analog-Sum-E > th0
(14) Front-W > th
(15) Back-W > th

```

8. Layer 2 Vertex DSM Board: L1-VT201

All threshold bits of the Vertex tree from the large and small-tile BBC, the ZDC and the VPD are brought into the Vertex DSM. The large-tile BBC data is no longer needed for triggering so it is ignored by this algorithm. The threshold bits from the other 3 detectors are passed on to the TCU. In parallel the TAC difference values are brought into the Vertex DSM. Windows are placed around each TAC difference, and the “inside window” bits get passed through to the TCU and the scaler system. A minimum bias bit, based on an OR of information from the 3 detectors is also created for use by the scaler system

RBT File: 11_vt201_2015_e.rbt

Users: VT201

Inputs: Ch 0 = BB101
 Ch 1 = BB102 (Unused in this algorithm)
 Ch 2 = ZD101
 Ch 3 = Unused
 Ch 4 = VP101
 Ch 5:7 = Unused

From Small tile BBC-DSM BB101
(0-12) Small tile TAC-Difference
(13) Unused
(14/15) Small tile ADC East/West sum > th0

From ZDC DSM ZD101
(0-8) TAC difference
(9) Digital-Sum-W > th
(10) Front-W > th
(11) Back-W > th
(12) Analog-Sum-E > th0
(13) Analog-Sum-E > th1
(14) Analog-Sum-E > th2
(15) Analog-Sum-E > th3 (Unused in this algorithm)

From VPD-DSM VP101
(0-12) VPD TAC-Difference
(13) Unused
(14/15) VPD ADC East/West > th0

LUT: Either 1-to-1 or TAC-difference range conversion

Registers:

R0: BBCsmall-TACdiff-Min (13 bits)
R1: BBCsmall-TACdiff-Max (13)
R2: ZDC-TACdiff-Min (9)
R3: ZDC-TACdiff-Max (9)
R4: VPD-TACdiff-Min (13)
R5: VPD-TACdiff-Max (13)
R6: VPD-TACdiff2-Min (13)
R7: VPD-TACdiff2-Max (13)
R8: Minimum-Bias-Select (3)

Action

1st Latch inputs

2nd Delay the BBC and ZDC threshold bits to the 4th step.
Delay the VPD bits and a 2nd copy of the BBC threshold bits to the 3rd step.
Compare each of the 3 TAC differences to its minimum and maximum value, as specified in the relevant registers. The logic looks for the TAC difference to be greater than the minimum and less than the maximum. The VPD TAC difference is compared to two separate sets of min/max values defining two separate windows.

3rd Make the VPD coincidence bit:
$$\text{VPD-COINC} = \text{VPD-E} > \text{th0} \text{ and } \text{VPD-W} > \text{th0}$$

Combine the results of the TAC difference comparisons to determine if each TAC difference is inside its specified window, e.g.:

$$\text{VPD-Tdiff} = R4 < \text{VPD TAC difference} < R5$$

Combine the results of the TAC difference comparisons and the ADC threshold bits to make the minimum bias bit, using R8 to turn each component on/off, i.e.:

$$\text{MB} = (\text{R8}(0) \text{ and BBC-S-Tdiff and BBC-S-E} > \text{th0 and BBC-S-W} > \text{th0}) \text{ or} \\ (\text{R8}(1) \text{ and ZDC-Tdiff}) \text{ or} \\ (\text{R8}(2) \text{ and VPD-Tdiff})$$

4th Latch Outputs

Output to TCU:

| Bit | Name | Description |
|--------|-------------|---|
| Bit 0 | BBC-TAC | BBC small-tile TAC difference in window |
| Bit 1 | BBC-E | BBC small-tile East ADC sum > threshold |
| Bit 2 | BBC-W | BBC small-tile West ADC sum > threshold |
| Bit 3 | ZDC-W | ZDC Digital-Sum-West > th |
| Bit 4 | ZDC-Front-W | ZDC Front-West > th |
| Bit 5 | ZDC-Back-W | ZDC Back-West > th |
| Bit 6 | ZDC-TAC | ZDC TAC difference in window |
| Bit 7 | ZDC-E | ZDC Analog-Sum-East > th0 |
| Bit 8 | ZDC-th1-E | ZDC Analog-Sum-East > th1 |
| Bit 9 | ZDC-th2-E | ZDC Analog-Sum-East > th2 |
| Bit 10 | Unused | |
| Bit 11 | Unused | |
| Bit 12 | MinB | At least one selected TAC difference in window |
| Bit 13 | VPD-TAC | VPD TAC difference in window |
| Bit 14 | VPD-TAC2 | VPD TAC difference in 2 nd window |
| Bit 15 | VPD-COINC | VPD East ADC sum > threshold and VPD West ADC sum > threshold |

Output to Scalers

| Bit | Description |
|------------|---|
| Bit 0 | BBC small-tile TAC difference in window |
| Bits 1:5 | Unused |
| Bit 6 | ZDC TAC difference in window |
| Bits 7:10 | Unused |
| Bit 11 | VPD TAC difference in window |
| Bit 12 | Unused |
| Bit 13 | MinB |
| Bits 14:15 | Unused |