TCU Counters

H.J. Crawford, J.M Engelage, E.G. Judd, J.M. Nelson, M. Ng, C. Perkins, J.M. Landgraf

March 2, 2011

All of the trigger-related logic on the TCU is implemented in two FPGAs on a daughter card. One FPGA contains all the logic for actually issuing triggers. The other contains the counter logic for monitoring how often each of the various trigger conditions is met. A communications scheme has been devised to allow the first FPGA to pass all the necessary information to the second. This document describes that scheme, as well as which counters are implemented, and how they are controlled and read out.

Change Log:

Date	Description
August 17, 2009	Original Version. Contained questions about: - list of counters and their enable/disable signals - size of counters - when to save data (external "latch" command, or internal timeout)
August 19, 2009	Questions answered: - list of counters and their control signals is OK - counters will be 40-bits wide - data saving will be initiated by external "latch" command If there is an "error" or "overflow", the VME client will just log those messages, rather than actively notify run control. A paragraph was added to the Counter Operations section to make it clear that latching the data into static registers does not interfere with incrementing the counters, and that the registers can then be read out over VME while the TCU is still running. The fast clock that will be used for multiplexing data on the U1:U2 link was increased from 4xRHIC to 8xRHIC. This will make it possible to increase the number of triggers in the future from 32 to 64, and still have enough bits and time available on the link to transfer all the data.
March 2, 2011	The gating on the PHYS counters was changed so that they count whenever the TCU is in Run mode. The PHYS+LIVE and PHYS+LIVE+PS counters still require Run mode and tokens in the token FIFO in order to count.

Counter Operation

The following set of counters has been implemented:

Name	Description	Number of Counters	Control Signals
PHYS	For each trigger (currently there are 32), the number of bunch crossings where the	32	Run/Stop

	physics conditions are satisfied.		
PHYS+LIVE	For each trigger, the number of bunch crossings where the physics conditions and the detector-live conditions are satisfied.	32	Run/Stop Token FIFO Empty Halt
PHYS+LIVE+PS	For each trigger that is enabled, the number of bunch crossings where the physics conditions, the detector-live conditions and the prescale condition are all satisfied and there are tokens available so a trigger is issued.	32	Run/Stop Token FIFO Empty Halt
DET-LIVE	For each detector, the number of bunch crossings where the detector is Live	16	Run/Stop
TCU-DEAD	The number of bunch crossings where the TCU cannot issue a trigger because the token FIFO is empty, or it has been "Halted"	1	Run/Stop
CROSSINGS	The number of bunch crossings	1	Run/Stop
TRIGGERS	The number of bunch crossings where a trigger was issued	1	Run/Stop
RESPONSES	The number of bunch crossings where a response was issued	1	Run/Stop
	Total	116	

Each counter is 40 bits wide. All the counters are cleared in parallel during the configuration process (see VME Control and Readout section below). After that each counter is incremented by 1 during every RHIC clock tick where its input bit is set and the counter is enabled. The control signals listed in the table for each counter are used to enable/disable that counter.

In order to use the counters for real-time monitoring it is necessary for them to be read-out frequently (e.g. every few seconds) when a run is in progress. To this end, periodically, the current value of each counter is latched into a static register. This happens in parallel with the normal process of incrementing the counters, and therefore does not interfere with the incrementing procedure. Once it has been saved, the data in the static registers is read out over VME while the TCU is still running. The handshake between the TCU and the VME client is described in detail in the VME Control and Readout section below.

Even the bunch crossing counter, counting at RHIC clock rate, will take over a day to count up to 2^{40}. No STAR runs are that long so we should never have to deal with overflows. As a result, the counters do not need to be cleared after the current value has been read-out. The values read out during a run are therefore running totals. The values read at End-of-Run are the final run totals. This means that, in a change from the previous version of the TCU, no offline addition of intermediate counter values is needed to get the run totals.

U1: U2 Link

In order to implement these counters in FPGA U2, all the bits need to be transferred from U1 to U2 on the U1:U2 link. Once there, the counters need to be read-out over VME. The VME communications bus does not connect to U2 directly, so that bus must be extended to U2 from U1 on the same U1:U2 link. That link is also used to transmit the output data from U1:U2. The full list of bits that need to be transferred between U1 and U2 is:

Name		Number of Bits
Counter Inputs and Control Signals	PHYS	32
	PHYS+LIVE	32
	PHYS+LIVE+PS	32
	Detector-LIVE	16
	Token FIFO Empty	1
	Halt	1
	Run/Stop	1
	Trigger	1
	Response	1
	Sub-total	117
VME Communications Bus	Address/Data Bus	32
	Bus Control	7
Output	Detector Bitmask	16
	TRG command	4
	DAQ command	4
	Token	12
	Total	192

There are a total of 192 bits that need to be transferred from U1:U2. However, the link is only 124 bits wide, so some multiplexing is necessary. The counter input and control signals will be multiplexed onto a 32-bit wide section of the link, using an 8xRHIC clock. 8 clock ticks on a 32-bit wide bus is enough time to transfer 256 (i.e. 8x32) independent bits. This is more than enough for the current set of 117 counter-related bits. In the future we may increase the number of triggers from 32 to 64. In this case there would be 213 counter-related bits (i.e. 3x64 + 16 + 5). 213 bits can still fit within the 256-bit limit of the multiplexed bus. The multiplexing scheme will bring the total number of bits used in the link down to 107, leaving 17 bits spare for future upgrades.

Clocks

The counter data needs to be transferred across the U1:U2 link every tick of the RHIC clock. It is multiplexed in U1 using an 8xRHIC clock, and then de-multiplexed in U2. In order for this data transfer to be reliable it is necessary for U1 and U2 to use the same 8xRHIC clock. The 8xRHIC clock is generated inside the U1 FPGA by multiplying the frequency of the RHIC clock using the Digital Clock Manager (DCM) built into the FPGA. The DCM also produces an inverted 8xRHIC clock. Both of those clocks are then driven, with dedicated global clock buffers, to a

dual data rate flip-flop whose output is connected directly to an output buffer of the U1 FPGA. The result is a properly constructed 50% duty-cycle 8xRHIC clock that can be used by U2.

VME Control and Readout

The scheme for the counter readout is one that is initiated by the VME client. At some point the VME client initiates the readout process, and all the counters simultaneously save their current value into static registers. Since the TCU uses the VME32 protocol, and the counters are all 40 bits wide, there are two static registers for each counter. With the current list of 116 counters, this results in 232 registers. These registers are then read out over VME. A handshake is necessary between the VME client and the TCU counter logic.

The readout procedure is as follows:

- The VME client initiates the procedure by sending a "latch counter values" command to the TCU.
- When it receives that command, the TCU latches the current values of all the counters into static registers and sets a "data ready" bit in a status CSR
- The VME client polls on that bit until it is set, and then reads out all the registers
- The VME client then sets a "reading done" bit in a control CSR.
- When the TCU detects that the "reading done" bit is set, it clears the "data ready" bit.
- If the TCU is ever instructed to latch the counter values, but the "data ready" bit has not been cleared, this indicates that old counter values have not been read out and are about to be over-written. In this case the TCU sets an "error" flag in the status CSR
- If the TCU ever detects that any one counter had overflowed it sets an "overflow" bit in the status CSR
- If the VME client ever finds the "error" or "overflow" bits have been set it will log those error messages appropriately.

The registers that are needed for counter control and status are:

Name	Definition
Counter Control	Bit 0: Clear all counters Bit 1: Latch counter values Bit 2: Reading done
Counter Status	Bit 0: Data ready Bit 1: Error Bit 2: Overflow

Even though the TCU currently has 32 triggers implemented, we do not necessarily use them all every run. If a trigger is not enabled during a particular run, its associated PHYS+LIVE+PS counter will never see a non-zero input bit, so the counter value will always be zero. There is no simple way to zero-suppress the register list in VHDL. So, there will be a fixed one-to-two counter-to-register mapping. The VME client can choose whether to read all registers every time, or make a run-specific list and read just those registers associated with enabled triggers.