

VME Intensity Monitor - Task #15052

Sum device for muon campus toroid (R:TOR703)

01/09/2017 02:24 PM - John Diamond

Status:	New	Start date:	09/22/2017
Priority:	Normal	Due date:	
Assignee:	Elliott McCrory	% Done:	0%
Category:	Sums	Estimated time:	0.00 hour
Target version:		Spent time:	0.50 hour
Description			
Request from Kyle:			
with the Muon program coming online this year we'd really like to have a R:TOR703 sum device over the \$E9 because they will be taking multiple extractions on the \$F6.			
Subtasks:			
Task # 17764: More summation devices requested			New

History

#1 - 01/12/2017 02:45 PM - John Diamond

- Assignee changed from John Diamond to Elliott McCrory
- Estimated time changed from 4.00 h to 16.00 h

This is the request from Kyle that I brought up in our meeting yesterday. They are requesting a new sum for I:TOR852 as well (see [#15051](#)). I recommend addressing this one first because S60TOR is not operational yet. I'll add a couple of comments to this ticket to explain how the sum devices work and then feel free to stop by with any questions.

#2 - 01/12/2017 03:15 PM - John Diamond

Most of the work is handled by the SumMgr class. SumMgr holds a table of sum devices that are created with the create(..) member function. Each device has an arm, trigger and disarm TCLK event. After receiving the arm event the trigger event is activated. For each trigger event the toroid is sampled (via a filter chain) and added to the sum after waiting for the given delay (usually 5ms). The trigger is disarmed when the disarm event arrives and the sum is reset to 0.

The SumMgr relies on a EvtTrigTclkEventGenerator object to deliver event notifications to a VxWorks message queue when the TCLK events occur. There is a task spawned by the SumMgr constructor that pends on the message queue waiting for the event messages to arrive, it then dispatches to one of the _handleXxx methods to arm/trigger/disarm.

Configuring the sum devices is done in the FE's startup script using the vmeintSumCreate(..) function defined in vmeintCommands.cpp.

The best example of an injection sum device is R:SE3853, which sums R:TOR853 on the \$E3. There are related devices for the E0, E1, E2 and E9 cycles as well. Take a look at mi14tor's startup script to get an idea of how they're configured.

Finally, the sum's ACNET readout is handled via the SumAccessor class which just delegates to VMEInt::sumRead(..). The Sum ID (specified as the first argument to vmeintSumCreate(..)) is passed through the SSDN channel.

#3 - 01/26/2017 04:46 PM - Elliott McCrory

These are ACNET Device Index = 0x000e (DEVICE_ID_INJ_SUM)

Examples are in the mi14tor startup scripts. The configuration for R:SE3853 is

```
vmeintSumCreate 0, 0, 0, 0xe3, 0xe6, 400, 6
```

These are interpreted as:

```
1. id      = 0
2. chainID = 0
3. filterIdx = 0
4. arm_on  = $e3
5. disarm_on = $e6
6. trigger = 400
7. delay   = 6
```

#4 - 04/02/2017 11:56 AM - John Diamond

FYI - I created a filter chain for R:TOR703, id = 0x0.