ESTIMATION OF THE RESPONSE TIME AND DATA FLOWS IN THE TOTEM DETECTOR CONTROL SYSTEM

F. Lucas Rodríguez, CERN, Geneva, Switzerland

Abstract

The TOTEM experiment at the LHC is composed by 3 different detectors, (Roman Pots silicon detectors, CSC T1 and GEM T2 telescopes). The Detector Control System (DCS) is generated in a highly automated process from external representations for connectivity and behavior. From these representations (one of them, the Finite State Machine tree) it is also possible to estimate the response of the system. It is possible to assign weights to each one of the nodes and estimate data transfers among subsystems, memory consumptions, reaction times, storage needs,... The main purpose of those estimations is not to do a full predictability analysis of the system, but just to provide a help in case of performance problems.

INTRODUCTION

The TOTEM (Total crOss secTion, Elastic scattering and diffraction dissociation Measurements) experiment at CERN [1, 2] will measure the size of the proton and also monitor accurately the LHCs luminosity [3]. To do this TOTEM must be able to detect particles produced very close to the LHC beams.

TOTEM consists of "Roman Pot Stations" (RP), "Cathode Strip Chambers" (CSC) Telescope 1 (T1) and "Gas Electron Multipliers" (GEM) Telescope 2 (T2). The T1 and T2 detectors are located on each side of the CMS interaction point in the very forward region, but still within the CMS cavern. Two Roman Pot stations are located on each side of the interaction point at 220 m and 147 m inside the LHC tunnel. Each Roman Pot station consists of two groups of three Roman Pots separated by a few meters.

Such kind is in the learning phase that will produce elaborated requirements for the Control System [4] [5].

TOOL FOR THE CALCULATIONS

An specific tool has been developed for the calculation of the rate of information exchanged among all the hardware component using the method proposed in next Section. It uses the pinout information of the detector and some heuristics to build a Finite State Machine (FSM) of the detector [6]. Also assigns to each level different values according to the Product Breakdown Structure (PBS) tag for the calculation factors as seen in Table 1.

Table 1: PBS configuration entry for Temp. Sensors

Property	Value	Explanation
Id	E.03.05.03	The identifier in the PBS.
Description	E. M. - Temp.	A text description of the PBS identifier.
Information Chunk	4.00 B	The information size of the value transmitted in the readout.
Variation Prob.	1	The probability of changing the value between two readout intervals.
Archiving Freq.	300 s	This value is multiplied by the probability of change and the information size.
Archiving Node	4.70 GB	Is the information that this node has to archive only by itself.
Archiving Overhead	16.00 B	Represents the overhead in the structure to store the InformationChunk in a database.
Readout Rate Freq.	500 ms	This value is multiplied by the probability of change and the information size.
Readout Rate Node	37.50 Kb/s	Is the information that this node is generating only by itself.
Readout R. Overhead	8.00 B	Represents the overhead of the InformationChunk in the communication protocol.
Time Execute	100 ms	Time needed to execute a request.

This tool is highly modular, and the process of calculation has three steps:

1. Parse the tables and build the FSM tree.
2. Assign and match the PBS entries in the tree.
3. Execute correspondent algorithm for the calculations.

Each algorithm is contained in a independent class that is dynamically loaded. It explores the FSM tree and the PBS items and calculates certain tags. Also, an algorithm can generate new tags in the PBS item during its execution.

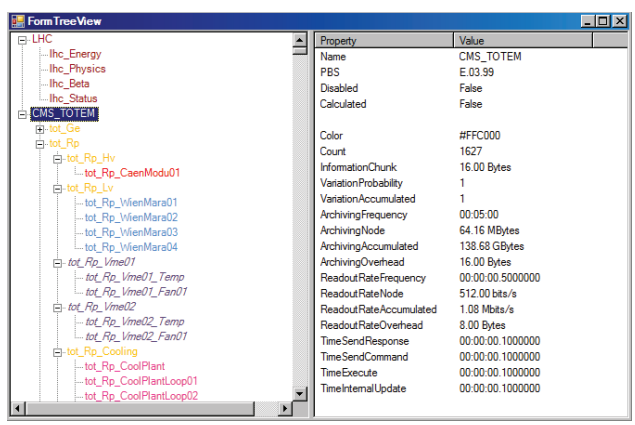

Figure 1: User interface of the tool

ANALYSIS OF THE RESPONSE TIME

Model of the CAN bus: SyncInter Validation

The first step is to calculate the time need to do a full readout of the bus. This is addressed with the *SyncInter* parameter. It indicates with what frequency the ELMB in the CAN bus must be pooled to retrieve the information.

$$\begin{aligned} BaudsInElmbReadout &= NumChannels * FrameLenght* \\ &\quad * BitToBauds\,(\text{baud}) \\ BaudsInCanReadout &= BaudsInElmbReadout \\ &\quad * NumberElmb\,(\text{baud}) \\ AdcInter &= (1/AdcRate) * 1000\,(\text{ms}) \\ ElmbInter &= NumChannels * AdcInter\,(\text{ms}) \end{aligned}$$

So the final value can be expressed such as

$$\begin{aligned} \text{BusExtractInter} &= (BaudsInCanReadout/CanSpeed)* \\ &\quad * 1000\,(\text{baud / (baud/ms)} = \text{ms}) \end{aligned}$$

3 requisites must be satisfied to be a valid *SyncInter*:

- *BusExtractInter must be less than ElmbInter*
  The bus has to be faster than the ELMB generation of data.
- *SyncInter must be bigger than BusExtractInter*
  Is is reasonable to wait the time needed for a full readout an ELMB before triggering a new one. If not the bus utilization would be over 100%.
- *SyncInter must be less than ElmbInter*
  For not loosing measured values it is necessary to trigger and transmit the values faster than they are being generated in the ELMB.

All those 3 requisites can be resumed in the following condition

$$SyncInter \in (BusExtractInter, ElmbInter) \qquad (1)$$

So the bus occupancy can be defined as:

$$Occupancy = (BusExtractInter/SyncInter) * 100$$

Values for the Roman Pots ELMBs chain for Temperature and Vacuum

This is the worst case scenario in TOTEM layout. It is a bus configured at 500000 bauds/s with 4 ELMB with 64 channels configured at a ADC speed of 1.88Hz. The frame length is 29 bits, the increase of the conversion from bit to bauds is factor 1.1.

The interval is (17ms, 34042ms). So an interval of 400 ms fulfills all the requirements and leads to a bus occupancy of 4.1%.

It is important to remark that changing the ADC rate of the ELMB does not affect the bus occupancy. This parameter is only affected by the *SyncInter*. What happens that the change in the ADC rate can lead to a no longer valid *SyncInter* as stated in Equation 1.

Model of the OPC

In the actual TOTEM DCS design, every machine has its own OPC server and client. The HV and LV OPC servers use the concept of OPC groups for refreshing the values.

The two main parameters in a OPC group are:

- *UpdateRate*
  It determines how frequently the client is notified of new values.
- *RefreshTimer*
  If no update is received in this interval, the client forces a poll to the server. Consequently *UpdateRate* must be greater than *RefreshTimer*. If this parameter is 0 this functionality is disabled.

The specification of OPC 3.0 allows to handle OPC items individually without any group. The ELMB OPC server implements this functionality.

Model of the FSM

Each FSM node has a processing time after one of the children changes or after receiving a command for the parent. This time is specific for each node, because it is directly related to the number of children, and the number of states of the node itself. Furthermore it can even include network requests,…

On average a reasonable estimation can be 500 ms for this internal processing time for each node. This value is obtained empirically.

Global chain

The DCS can be considered as a “soft real-time” system. If the sensors are read with some delay, or the commands sent a few milliseconds later, no damage or wrong calculations should take place.

An example of estimation of the response time of a DCS action is the vacuum failure. The DCS needs to react on time by disconnecting the cooling plant loop and the Hv and Lv power. The aim is to avoid condensations with the consecutive formation of ice around the electronics leading to short circuits. This scenario is calculated in *Situation A* of Table 2. The FSM states will propagate upwards 3 levels in the FSM hierarchy up to the proper station. Here a command is sent to the cooling child node to close the loop.

The approach is straightforward:

1. Determine the *RefreshRateSensor* of the vacuum sensor and the actuator. It will be the *SyncInter* of the bus CAN, or the *UpdateRate* for Wiener and CAEN OPC servers.
2. Calculate the *CommonNode* in the FSM hierarchy between the *SensorNode* and the *ActuatorNode*.
3. Assign to *Measure* the number of levels of difference between the *CommonNode* and the sensor.
4. Assign to *Command* the number of levels of difference between the *CommonNode* and the actuator.
5. The total *FsmProcessing* time is (*Measure* + 1 + *Command*) $*$ 500 ms.
6. The final *ReactionTime* for a hypothetical soft interlock is *RefreshRateSensor* + *FsmProcessing* + *RefreshRateActuator*.

Table 2: Soft interlocks estimation

	Situation A	Situation B
Sensor	Vacuum sensor inside a Pot	Temperature sensor inside an Hybrid
Actuator	Cooling circuit for a full station	Wiener Maraton Low Voltage Group
Sensor Node	tot_Rp_45_220_fr_tp_Vacc01	tot_Rp_45_220_fr_tp_Temp01
Actuator Node	tot_Rp_45_220_CoolPlantLoop	tot_Rp_45_220_fr_LvG
Common Node	tot_Rp_45_220	tot_Rp_45_220_fr
Refresh Sensor	400 ms	400 ms
Refresh Actuator	500 ms	3000 ms
Measure	3	2
Command	1	1
FSM Processing	2500 ms	2000 ms
TOTAL *Reaction*	3400 ms	5400 ms

The complementary action of switching off the Hv and Lv is taken inside the Pot level, so the propagation is only 1 level up.

ORDERS OF MAGNITUDE OF THE INFORMATION EXCHANGED

The lifetime of the DCS is estimated to be 20 years of continuous operation. Table 3 gives a total overview of the DCS expected workload considering the RP and T2 detectors.

The main result of this table is that "only" 150 GB are needed for the TOTEM DCS archiving. And the total data-flow of the whole system is slightly above 1 Mbit/s.

The data-flow results are not a CPU load estimation, but are directly related. Even if the data-flow increases up to 2 Mbits/s taking in consideration T1 detector, a usual computer of nowadays could handle the requirements of the whole TOTEM experiment.

Table 3: Extract of distribution from DCS PBS elements

PBS	Description	Count	Archiving	Readout
E.03.01	High Voltage	34	5.33 GB	42.50 Kb/s
E.03.02	Low Voltage	86	8.08 GB	64.50 Kb/s
E.03.03	Front E. Electr.	18	1.13 GB	9.00 Kb/s
E.03.04	DCU	720	67.67 GB	540.00 Kb/s
E.03.05.01	E. M. - Canbus	4	160.40 MB	1.25 Kb/s
E.03.05.02	E. M. - ELMB	11	441.10 MB	3.44 Kb/s
E.03.05.03	E. M. - Temp.	120	4.70 GB	37.50 Kb/s
E.03.05.04	E. M. - Vacuum	144	5.64 GB	45.00 Kb/s
E.03.05.05	E. M. - Humid.	24	1.50 GB	12.00 Kb/s
E.03.06	Cooling	9	1.41 GB	11.25 Kb/s
E.03.08	Motor. values	101	15.82 GB	126.25 Kb/s
E.03.09	D. Safety Sys.	40	6.27 GB	50.00 Kb/s
E.03.99	FSM nodes	308	19.30 GB	154.00 Kb/s
E.03	SUM	1632	**139.47 GB**	**1.09 Mb/s**

CONCLUSIONS

This tool and philosophy provides only an estimation. It cannot be considered as a explicit scheduling mechanism or an study on synchronization.

However it has proved to be extremely useful for the design of the whole system. Usually the DCS is built and then takes place several test to see if it behaves properly. Those experimental test can never be comprehensive enough, and could lead to situations where the timing requirements does not fulfill.

ACKNOWLEDGMENT

I would like to give thanks to the TOTEM collaboration colleagues. They have provided the information needed of how to decompose the system in the different levels and the behaviour implemented in the FSM.

REFERENCES

[1] V. Berardi, M. G. Catanesi *et al.*, "TOTEM: Technical Design Report," CERN-LHCC-2004-002, January 2004.

[2] G. Anelli, G. Antchev *et al.*, "The TOTEM experiment at the CERN Large Hadron Collider," JINST, 2008.

[3] M. Albrow, G. Antchev *et al.*, "Prospects for diffractive and forward physics at the LHC," CERN/LHCC 2006-039/G-124, 2007.

[4] F. Lucas Rodríguez, I. Atanassov *et al.*, "The totem detector control system," in *Proceedings of ICALEPCS 2009*, Kobe, Japan, October 2009.

[5] F. Ravotti, I. Atanassov *et al.*, "The totem on-line radiation monitoring system," in *Proceedings of ICALEPCS 2009*, Kobe, Japan, October 2009.

[6] F. Lucas Rodríguez, I. Atanassov *et al.*, "Automation tools in the software development of the totem detector control system," in *IEEE NSS 2010*, Knoxville, Tennessee, November 2010.