CMX Software Development: Status and Plans

Seth Caughron
Michigan State University

Level-1 Calorimeter Trigger Joint Meeting, 10/10/12

Overview

- The CMX module will replace the current CMM module in the same position in the L1Calo Readout crates

- From the project specification document:
  - Will inherit all main logical components of the CMM
  - Will receive higher data volumes from upgraded processor modules via the backplane
  - Higher data processing capabilities
    - To format and transmit data to the Topological Processor
    - To transmit additional information to the RODs
    - To perform some topo trigger tasks in “standalone mode” in the absence of a dedicated TP

- As with any new L1Calo module, we need two new software packages: **cmxServices** and **cmxSim**

Integration into L1Calo Framework

- cmxServices and cmxSim have been cloned from the existing CMM counterparts and are in SVN

- Also updated: infra-, dal-, is-, stats-, defs-, dbL1Calo, dbSim, etc….
  - → checking out and building a new L1Calo area with the standard scripts should include the new packages
  - → OKS module classes for CMX are there
  - → HDMC parts files are there
  - → test vector generators are there

- **However:** The contents of the CMX packages have not been updated for CMX functionality.
  - Changed objects named "CMM" to "L1CaloCMX"
  - Will be approximately adequate for backward-compatibility mode
  - But still have two other modes of operation

CMX Modes of Operation

- **First order of business: how to encode the switching between the three modes of operation?**
    - Backward-compatibility
    - Data source for TP
    - Standalone

- The services and simulation software will need to do very different things in these three modes.
    - → Is there an example of an L1Calo module that switches between modes in a similar way? Could be used as a rough template going forward.

- Backward-compatibility mode clearly the simplest
    - Maybe some slight changes to register map to accommodate new hardware
    - Otherwise, cloning the CMM software does the trick by definition and has already been done.

Data Source for TP Mode

- **Will be the main CMX mode in the fully-implemented upgrade program.**

- Input from the processor modules will be in a new, non-backward-compatible format.
  - Will require changes to the register map.
  - → How clear is this input data format right now?

- The CMX may (will?) continue to supply data to the CTP along with the TP… Changes to CTP output will be in the details.

- **Processing and output of data to the TP likely represents the area with the largest need for development.**
  - Sending even unprocessed or unreduced data is a new function.
  - → What is the current state of thinking around reducing the data in the CMX (i.e. sending only non-zero ROIs?). This is also a new functionality which needs new code.

Data Source for TP Mode (cont.)

- I assume a useable model for ROI output exists in the current code for output to CTP.

- However, the ROI information to TP is expected to look different from the current format, another area for development.

- A note on an envisioned test sub-mode:
  - For initial topological upgrade testing, we may want to operate the CMX in backward-compatible mode while sending unprocessed/reduced data to the TP in parallel for test purposes.
  - **Might be advantageous to add this to the list of modes and develop the code accordingly.**
  - → Is the firmware development currently assuming that this mode will be implemented?

Standalone Mode

• One CMX module may be used to perform topological processing tasks, e.g. if the TP is delayed.

• How to handle a CMX/TP in the code is a significant question.

• **Would obviously require a lot of coordination between CMX and TP software.**

• As the TP software is being developed in any case (see next talk), the guts of the code will already be there.

• This mode is described in the documentation as "optional," and I'm currently assigning it a low priority. → Is it foreseen to use this option?

Summary

- Packages for CMX services and simulation have been cloned from the CMM code, and CMX objects have been added to the rest of the L1Calo framework.
    - Including OKS and HDMC objects
    - This should be largely sufficient for backward-compatibility mode

- However, the software needs to "know" when to use which mode, and how we design the code around different modes is something which needs to be thought about carefully.

- The main mode – data source for TP – will require the most development of new code.

- The need for further work on e.g. test sub-modes or standalone mode will likely be determined by choices made on the hardware/firmware side.