Trigger Software

January 2008

John Nelson

These notes are intended for anyone taking over responsibility for trigger software. The software is supported by three types of operating systems and the individual elements will be outlined under these headings.

All software is based in the STAR software repository and can be obtained from CVS by reference to online/RTS/trg

VME hardware and VxWorks OS

Trigger hardware consists of numerous VME crates, each controlled by one CPU (two for L1) and containing boards of different types. VxWorks is a real-time operating system (without virtual memory) and the address space of each CPU covers the address space of all the boards it controls.

The VME standard is fully documented and is a necessary background to understanding how the crate system works.

VxWorks provides support for inter-process communication and there is a trigger-daq inter-processor messaging system which is fully described in DAQ documentation as ICCP9,

There are 6 types of VME systems covered by trigger software: Scalers (VME wrappers only), DSM, QT, L1 and L0, and RCC Common code is used for all DSM crates, and separate common code for the QT crates. The L1 crate is unique and has its own suite of code. The TCU is controlled by the L0 CPU (situated in the L1 crate) and L0 code manages the flow of tokens to the TCU and generates the event-build command to the rest of trigger software.

A common set of utilities and libraries is also provided.

The scaler crate is controlled by software which sends data to l2ana02 in the L2 system.

RCC monitors the RHIC clock and the RCC board. This has specific control code.

Note that all data produced by VxWorks systems is big-endian.

Code is compiled on startrg.starp.bnl.gov which supports the VxWorks cross-compilers and libraries.

Each VME CPU is configured by a startup script.

L2 and Linux

Level 2 is controlled by code in the l2ana01 computer supported by RedHat 2.4 OS. Data from all the trigger detectors, including BTOW and ETOW, are collected by L2 and formatted into a trigger data block and sent onward to DAQ. Shell scripts run in the background to move other data to l2ana02 and to trgscratch for further processing and, in the case of l2ana02, to RCF and HPSS for storage. Most of the shell scripts are initiated at boot-time. See /etc/rc.local on different computers.

The ALICE RORC links to BTOW/ETOW are also managed by L2 software.

The primary function of L2 is to generate an ACCEPT (or ABORT) decision which is fed to the code in L0 controlling the TCU. This decision is the result of data analysis performed by a series of algorithms that L2 calls once all data from a given event have been received.

Note that all data produced by Linux are little-endian and byte-swapping is used to translate VME data. The trigger block is finally constructed in big-endian format.

Support from SUN Solaris

The cross-compilers for VxWorks are provided on startrg.starp while the CVS connection to AFS is provided by startrg2.

The whole of trigger is controlled by configuration files which describe the contents of each VME crate including the registers and lookup tables that must be loaded at the start of a run.

These files are parsed by a suite of routines and a single binary file (Tier1 file) is generated as a result. MakeConfig and ReadConfig can be executed on either startrg or startrg2.

Occasionally it is necessary to modify MakeConfig code in order to incorporate additions to the syntax of the configuration files.

Other utilities are provided to mask hot towers in the Barrel and Endcap calorimeters.

General notes

Code is generally fully documented inline. Documentation of hardware is contained on the TRG sub-system web-site and familiarity is essential.

VxWorks libraries and operating routines are documented and explained in considerable detail in two reference books, copies of which are widely available.

Interaction with Run Control is described in ICCP9 documentation on the DAQ sub-system web-site. Headers in /RTS/include provide other essential information.

Starting point

You need to get an overview of how the trigger system works, including the role of the TCU, the DSM tree as well as the function of tokens, and the RHIC clock and finally the initial processing to form an ACCEPT (or ABORT) decision by L2.

Become familiar with the directory layout of the 'trg' account on startrg.starp. On your own account, use CVS to check-out trigger software from online/RTS/trg. You should also checkout online/RTS/include which contains all common headers required by trigger and DAQ. Pay particular note to trg/include/trgStructures.h which defines trigger data.

Understand how DSM code works, including mechanisms for reading DSM boards, how the configuration file is obtained from Run Control and then decoded. Understand the sequence of messages that are sent from Run Control to start the run, configure the boards, and finally end the run. Understand how the messaging system works. Importantly, understand how error/information logging operates.

The QT system operates in a similar way, although the readout and configuration of the QT boards is different.

The configuration engines DSM_Config and QT_Config are essential reading.

Study code in L1, beginning with the token manager. Leave the hardware interface code to the last. This manages the rate of flow of tokens to the TCU, executing ACCEPT and ABORT commands from L2, and dealing with low priority messages from the token manager.

Once you are familiar with the various aspects of trigger operation in VME, then move on to L2.

Have fun.