How Trigger Software Works

Note 1: Run configuration

John Nelson

Before proceeding, use CVS to checkout online/RTS/include as well as online/RTS/trg/include. I will make constant reference to these as (RTS) and (trg). You should also have a copy of the messaging protocols to hand. (See DAQ sub-system web page and go to ICCP9.) Note that $DEV refers to /home/startrg/trg/trg_soft_dev on startrg.starp.bnl.gov.

The Run Control panel is used to select which principal entities are in the run, such as Trigger and DAQ , as well as the main detectors, TPC etc and, on the configuration page, which trigger detectors, CTB, FMS etc.

When the Startrun button is pressed, and the number of events is selected, the following sequence of actions takes place.

1. Run Control sends the RTS_SEND_CONFIG message (see ICCP9) to all systems that are selected to be in the run. The message includes the name of a the global setup file (see (RTS)RC_Config.h) which contains all information that every sub-system needs to have in order to configure itself. In the case of trigger detectors, this will include all the register and algorithm settings, the various parameters set in the TRG_RUN page, and, in particular, the name of the binary Tier1 file.

2. The Tier1 file. (In the early design of the configuration scheme, I thought that three files, in a chain - hence tiers - would be needed. In the end I found that the others could be hidden so only one file remained. Tier1) This file contains configuration details for all trigger hardware, that is, the TCU, RCC, all DSM and QT crates, L1 and L2. The files that drive the construction of the Tier1 files are to be found on startrg. (See $DEV/../cfg/Tier1) Look at one of these. It contains a short header explaining what is needed and then names of configuration files that are specific for each item of trigger hardware. Select one of the files that refers to a DSM crate, for example, and examine it. There is a long header which explains the syntax which describes the configuration requirements and default register settings for each DSM board in that crate.

These files are normally edited by Jack or Eleanor. The software that is used to parse these files and construct the final, single binary Tier1 file is MakeConfig and the source code is in $DEV/util/SUN/Tier. A binary Tier1 file can be decoded using ReadConfig. (If you are looking at these files, you should be logged on as ‘trg’ so try ReadConfig <Tier1 filename>

3. Each trigger entity on receipt of the RTS_SEND_CONFIG command, reads the global setup file and extracts those items relevant to itself. The Tier1 file is read and information is extracted and downloaded into the hardware. The main engines for doing this are DSM_Config.C and QT_Config.C the source codes of these is in $DEV/trglib. $DEV/include/trgConfig.h is an important header that is used in this process.

These engines are called from dsmConfig.C and qtConfig.C (See $DEV/DSM and $DEV/QT) Common code is used for all DSM crates and QT crates respectively. Each crate is known by a unique CONF_NUM (see $DEV/include//trgConfig.h) to allow individual changes, where necessary.

At this time, all pedestal files are loaded into lookup-tables (LUT).

The L1 and RCC crates are different and have their own configuration codes. (See $DEV/L1/config/L1_Software_Config.C and L1_Hardware_Config.C. Also

$DEV/RCC/server/RCC_Config.C) L1 configuration is more complicated because it must construct the LUTs for the TCU which is controlled by the L0 CPU (in the same crate as L1).

4. When each crate completes its configuration phase, it sends an acknowledge message to Run Control together with a success/fail indication. Run Control will wait (sometimes you see a W on the Run Control configuration page) until every sub-system marked in the run has responded. Only then will Run Control send RTS_RUN_START to every sub-system except RCC. As part of this message, Run Control indicates how many events are required. This is always 0 in the first phase of starting a run. This ensures that all sub-systems are primed and ready to go. When all the sub-systems have replied, RCC will be the last to receive RTS_RUN_START and RCC then puts all the DSM crates into run mode.

The DSMs are large memory banks with 64k addresses. The address pointer is set to zero by the RCC and then the pointer advances one address for every RHIC clock. It is essential that the address pointers advance in lock-step for every DSM board in every DSM crate.

At this point, all sub-systems are in a RUN_PAUSE state.

In the next phase, Run Control sends RTS_RUN_RESUME to all sub-subsystems although this is only used by L1. The message contains the number of events for the run. Now L1 sends a message to the L0 CPU, which controls the TCU, and tokens are loaded into one of the TCU registers and triggers can then be issued when an appropriate trigger condition occurs.

The run has started.