HDMC Evolution (1)

Murrough Landon – 20 March 2000

We have, for some time, had a wishlist of our desired improvements to HDMC. Some of these have now been addressed. For others, we have agreed potential solutions.

Handling of Bus Errors

This is essential to allow scanning of VME space, checking that an expected configuration is correct, and for robustness against errors interactively entering VME addresses.

This has now been implemented, using the same scheme as in the UK diagnostics. Bus errors are transported across the network VME connection and generate C++ exceptions.

Remove STL/Qt from VME BusServer

Previously the HDMC BusServer, which provides VME access via the network to remote systems, used the same STL and Qt libraries as the rest of the code. Unfortunately these are not supported on our old MVME167 systems running LynxOS.

This dependence has now been removed. (But has yet to be tested in the UK?). In the meantime, we made progress using Linux on these old processors instead of LynxOS.

HDMC Evolution (2)

Overview GUI for Modules

Users of the UK diagnostics are familiar with the large panels showing groups of a Modules Registers and Memories. HDMC displays each Register in a separate window, so you can end up with lots of little windows.

Cornelius has agreed to look into implementing a collective GUI to gather a Modules Parts (registers, memories, etc) into a single display. He will report on the feasibility of this at Mainz.

C++ Access to Register Bitfields

In the GUI, you have easy access to all the bits and bit fields of a Register. For the DAQ and for test programs, we would like similar easy access to bit fields from C++.

The layout of each register is defined in a configuration file. This could be parsed to create C++ code which implements the kinds of functions we require.

A detailed scheme needs to be worked out and agreed at Mainz.

C++ Access to Module Registers

Similarly, the GUI shows the list of Registers belonging to a Module. In HDMC this is a dynamic list: the Module class doesnt “know” its own Registers.

For the DAQ, it would be convenient if Module subclasses “knew” the list of their own Registers. This should not be difficult to implement.

HDMC Evolution (3)

Verifying Registers etc

Testing of Registers etc in HDMC is currently entirely visual.

It would be useful if all Parts had a recursive Verify() method which could be defined to run an acceptance check on the Part.

This should be fairly simple to add.

Scripting

For more extensive tests, it would be useful to have a scripting capability. A rudimentary facility has been implemented, but use of a standard scripting language (Perl, Python) would be much better.

This is non trivial.

Use of DAQ -1 Configuration Database

HDMC reads its configuration from a structured text file. The DAQ will use an OO database to store the hardware configuration. It would be useful if HDMC could also use the same configuration data. This would still need to be supplemented by other detailed configuration files.

However, our configuration database still needs to be defined... (though some work has started).