Work on HDMC in the UK

Murrough Landon – 6 July 2000

Overview

- Generation of C++ Register subclasses for access to bit fields from DAQ programs
- Configuration and Parts files for ROD and TTCvi
- Verify for Registers
- Incomplete development of ModuleView
- Miscellaneous

Generation of C++ Register subclasses

Requirement

HDMC displays bit fields of Registers to interactive users. The DAQ requires such access from C++.

Desired Calling Syntax in C++

```
MyReg *myreg;
int value = myreg->SingleBit();
int oldvalue = myreg->FourBitField(6);
```

Implementation

Small program parses `default.conf` file and generates subclasses of `ModuleRegister32` or `ModuleRegister16` for specified list of Registers.

```
$ cat >register32.list
DSS.MotherCR
DSS.MotherSR
^D
$cat >register16.list
TTCvi.CSR1
TTCvi.CSR2
^D
$ regbuild -d /tmp -l register32.list
$ regbuild -d /tmp -l register16.list -c ModuleRegister16
```

Status

Working: but only for the default bit fields. At the moment alternate representations are ignored.

Configuration and Parts files

Makefile for default.conf

New scheme to create default.conf from separate files using the C preprocessor `#include` directive. This makes it easier to update configurations for different modules independently.

Various Additions

New definitions of ROD register formats added to default.conf and new parts file created. Some updates for TTCvi and DSS.

Verify for Registers

Overview

HDMC version 0.2 incorporates a general scheme for verifying Parts.

By default the `verify()` method of a Part verifies itself by calling `verify_me()` and may then recursively verify all its component Parts.

The default `verify_me()` method just returns `true`.

Registers

ModuleRegister subclasses now have a real `verify_me()` method which performs a number of write, read and compare cycles:

- a few fixed patterns: `0`, `FFFF`, `AAAA`, `5555`, etc
- each single bit, ie: `1`, `2`, `4`, `8`, etc
- pseudorandom patterns up to the number of cycles requested

ModuleView (1)

Requirements

The old diagnostics by default shows all of a modules registers collected together in (approximately) one window in a fairly compact, customised display.

By contrast, HDMC has always presented each register in a separate window whose layout is generated automatically from the Register format descrption in the `default.conf` file. Originally there was no GUI at all for the Module class.

For modules with many registers this can result in many windows and a cluttered screen.

Intentions

The ModuleView is intended to

- provide a GUI for Modules
- display all the known Registers of a Module in a fairly compact view
- generate the layout automatically (by default)
- allow for some customisation of layouts (by specification in the default.conf file)
- still allow easy access to the old Register GUI from the ModuleView

ModuleView (2)

Implementation

- The GuiRegister class has been split up to allow the register display to use two different styles. WidCompactReg and WidStandardReg are subclasses of a new base class WidRegister.
- Some extra methods have been added to the BitPack class which is now in a separate file from BitMapper (changes to Makefile required).
- A new layout classes (SimpleFlowLayout) has been added, using one of the Qt examples. This is used in the compact register display.
- A tabbed style display is used for Modules with too many registers to be comfortably displayed in one window.

Outstanding

- Basic implementation has been done.
- Some layout bugs (in new and old styles) have become apparent with the new version of Qt. Still to be fixed
- Signals and slots not yet connected for new style
- Customised layout not yet started
- Cleanup of the old style register display code would be very desirable.

Miscellaneous Others

DAQ Use

- Use object library...
- Read partial Part hierarchies
- ...some problems to be resolved

Busserver and Bit3

- Bit3 VME link used at Birmingham
- Needs special treatment after bus error
- Busserver modifications tested

AOB

Future Developments?

- Cleanup code/layout of Register widgets
- Register characteristics: read/write masks, readout to DAQ, other?
- Extend verify methods to memories
- Add non-Register parts to ModuleView?
- Status/message area to be added to main window?

HDMC vs DAQ -1 Configuration?

- Consider future configuration database
- Want single primary source
- **Either**: OO database in DAQ -1
- **Or**: Parts and configuration files in HDMC
- Convert HDMC to use DAQ -1 database?
- ...or convert database to/from text files?
- Consult with DAQ -1 people

Further Suggestions?