http://cmsdoc.cern.ch/~cschwick/software/documentation/HAL/index.html

HAL (Hardware Access Library)

A library developed for the CMS data aquisition system at CERN.

Last update: 19th October 2006 (new Version)

A high level interface which allows for user friendly hardware access to VME or PCI modules.

The up-to-date HAL version with the supported drivers and BusAdapters is contained in the worksuite of the CMS online software. The download links below are only for those which want to use the HAL stand-alone (i.e. not in the CMS online context).

The library contains a software layer between the user program and the driver software to access hardware. A class called "BusAdapter" implements the interface to the various hardware drivers. Several BusAdapters are contained in the distribution. It is easy for the user to extend this collection by writing his own BusAdapter.

Sample HAL Address Table

```
*********************************************************************
*     item              AM width address  mask      read write description
*********************************************************************
PCI_ID                  09  4  00000000   ffffffff    1    0   Enable PCI memory
BaseAddressRegister     09  4  00000010   ffffffff    1    0   The Base Address Register
pem_en                  09  4  00000004   00000002    1    1   Enable PCI memory address
perr_en                 09  4  00000004   00000040    1    1   Enable Parity Error report
serr_en                 09  4  00000004   00000100    1    1   Enable System Parity Error report
FirmwareVersion         09  4  00000008   0000ffff    1    0   Firmware revision
BAR                     09  4  00000010   ffff8000    1    1   BAR0 Programmable Content
GEN                     09  4  0000002c   0000ffff    1    1   Packet length for Pattern generator
CommandStatusRegister   09  4  00000038   ffffffff    1    1   Command Status Register
C_RESET                 09  4  00000038   00000001    1    1   Clear FIFO
C_TEST                  09  4  00000038   00000002    1    1   Enable test mode
C_REPEAT                09  4  00000038   00000004    1    1   Enable auto-repeat
C_TRIG                  09  4  00000038   00000008    1    1   Trigger one transmit cycle
C_STOP                  09  4  00000038   00000010    1    1   Stop FIFO Reading
```

Item Name (used in C++)

VME Address Modifier

Register Address

Width (bytes)

Bit Mask

Read Write Permission Bits

Description

```
/*
 * Sample HAL code...
 */

/*--- required includes ---*/
#include "VMEDevice.hh"
#include "VMEAddressTable.hh"
#include "VMEAddressTableASCIIReader.hh"
#include "CAENLinuxBusAdapter.hh"


/*--- initialization ---*/
    /* connect to PCI VME bridge */
    HAL::CAENLinuxBusAdapter* busAdapter  ( V2718, unit, chain);

    /* create address table reader */
    HAL::VMEAddressTableASCIIReader addressTableReader( "TTCviAddressMap.dat");

    /* read the address table */
    HAL::VMEAddressTable addressTable( "Test address table", addressTableReader );

    /* create VME Device */
    HAL::VMEDevice TTCviCard(addressTable, *busAdapter, base);

/*--- read/write to items... ---*/
    TTCviCard.write( "Item name”, value);
    TTCviCard.read( “Item name”, value);
```

Of course, there are many more options than just the simple read() and write()... see docs

Public Member Functions

virtual void	**unmaskedWrite** (string item, unsigned long data, **HalVerifyOption** verifyFlag=HAL_NO_VERIFY, unsigned long offset=0) const=0 throw (NoSuchItemException, IllegalOperationException, VerifyException, BusAdapterException, AddressOutOfLimitsException) *Write a data word without applying the mask for the item.*
virtual void	**write** (string item, unsigned long data, **HalVerifyOption** verifyFlag=HAL_NO_VERIFY, unsigned long offset=0) const=0 throw (NoSuchItemException, IllegalOperationException, VerifyException, BusAdapterException, MaskBoundaryException, AddressOutOfLimitsException) *Write a new value for an item into the hardware device.*
virtual void	**writePulse** (string item, unsigned long offset=0) const=0 throw (NoSuchItemException, IllegalOperationException, BusAdapterException, AddressOutOfLimitsException) *Perform a write access in order to trigger side effects.*
virtual void	**unmaskedRead** (string item, unsigned long *result, unsigned long offset=0) const=0 throw (NoSuchItemException, IllegalOperationException, BusAdapterException, AddressOutOfLimitsException) *Read a data word without applying the items mask.*
virtual void	**read** (string item, unsigned long *result, unsigned long offset=0) const=0 throw (NoSuchItemException, IllegalOperationException, BusAdapterException, MaskBoundaryException, AddressOutOfLimitsException) *Read an item from the hardware device.*
virtual bool	**check** (string item, unsigned long expectation, string faultMessage="", unsigned long offset=0, ostream &os=cout) const=0 throw (NoSuchItemException, IllegalOperationException, BusAdapterException, AddressOutOfLimitsException) *Read an item from the hardware and check the result.*
virtual void	**readPulse** (string item, unsigned long offset=0) const=0 throw (NoSuchItemException, IllegalOperationException, BusAdapterException, AddressOutOfLimitsException) *Perform a read access in order to trigger side effects.*