ATLAS Internal Note
ATL-D-ES-0012
Version 1.1
5 February 2003

TTCvi Library for ROD Crate DAQ

Authors : R. Spiwoks
Keywords : TTCvi, ROD Crate DAQ

Abstract

This note documents the library for the TTCvi in ROD Crate DAQ.

1 Introduction

This note documents the library for the TTCvi in ROD Crate DAQ. The TTCvi itself is documented in Reference [1]. The library is based on the VME bus library/driver of ROD Crate DAQ and its C++ wrapper [2].

2 Package

The library for the TTCvi in ROD Crate DAQ is contained in package **RCDTtc** of the TDAQ software repository [3]. In order to build the package the instructions in the reference have to be followed.

The RCDTtc package uses the RCDVme, vme_rcc, rcc_error, rcc_time_stamp, io_rcc, cmem_rcc, RCDMenu and RCDUtilities packages which are all contained in the TDAQ ROS software repository.

3 Application Program Interface

3.1 Overview

The following sections list the public data types, constants and methods of the TTCVI class.

3.1.1 Data Types

The TTCVI class defines two data types which are used for B-channel commands:

- LongCommand
- ShortCommand

3.1.2 Constants

The TTCVI class defines constants for use with the methods. The constants are documented with the methods where the constants are used.

3.1.3 General Methods

Global methods

The TTCVI class provides a unique constructor, a unique destructor and other global methods which apply to the whole TTCvi:

- TTCVI
- ~TTCVI
- reset
- dump

Configuration/Identification

The TTCVI class provides methods which allow to obtain information concerning the configuration/identification of the board:

- manufacturerGet
- boardIdentifierGet
- boardRevisionGet
- mkTypeGet

BC and ORBIT Methods

The TTCVI class provides a method to read the BC delay from the CSR1, as well as methods to handle the ORBIT generation:

- bcDelayGet
- orbitInputSet
- orbitInputGet

Trigger Word Methods

The TTCVI class provides methods to handle the Trigger Word and Event/Orbit counter transfer after each L1A:

- triggerWordEnable
- triggerWordDisable
- triggerWordGet

Event/Orbit Counter

The TTCVI class provides methods to handle the Event/Orbit counter:

- counterValueGet
- counterSelectionSet
- counterSelectionGet
- counterReset

3.1.4 Level-1 Accept methods

L1A Generation

The TTCVI class provides methods to handle the L1A generation:

- l1aInputSet
- l1aInputGet
- l1aRandomSet
- l1aRandomGet
- l1aGenerate

L1A FIFO

The TTCVI class provides methods to monitor and control the L1A FIFO:

- l1aFifoEmpty
- l1aFifoFull
- l1aFifoReset

3.1.5 B-Channel Methods

B-Go Channels

The TTCVI class provides methods to handle the B-Go channels:

- bgoModeSet
- bgoModeGet
- bgoCommandPut
- bgoGenerate

B-Go Inhibit Signals

The TTCVI class provides methods to handle the INHIBT signals associated to the B-Go channels:

- bgoInhibitOn
- bgoInhibitOff
- bgoInhibitGet

B-Go FIFOs

The TTCVI class provides methods to monitor and control the FIFOs associated to the B-Go channels:

- bgoFifoEmpty
- bgoFifoFull
- bgoFifoRetransSet
- bgoFifoRetransGet
- bgoFifoReset

Asynchronous Commands

The TTCVI class provides a method to generate asynchronous B-channel commands:

- asyncCommand
- asyncPendingGet

The following remarks apply to all functions defined in the API:

- The reading and writing of the VMEbus master mapping for the TTCvi uses the safe access functions, cf. [2].
- The methods return an unsigned integer (u_int). This contains the return value used by the VMEbus library, cf. [2]. The value 0 stands for a successful termination of the method while any value different from 0 indicates an error.
- Some methods generate informational print-out to **cout**. In case of an error all methods generate error print-out to cerr.
- The package contains a **DEBUG** flag to be set in the **cmt/requirements** file. When the package is compiled with this flag on, then the TTCvi library will generate debug print-out to **cout**.
- The TTCVI class, its methods, constants, and data types are defined in the RCD namespace.

3.2 Data Types

TTCVI::LongCommand

Synopsis

```
in vme_rcc.h:

typedef struct {
    u_short             address;
    bool                external;
    u_short             sub_address;
    u_char              data;
} LongCommand;
```

Fields

u_short address	14-bit TTCrx address (0x0000 to 0x3fff)
bool external	flag for external sub-address in associated front-end electronics
u_short sub_address	8-bit sub-address (0x00 to 0xff)
u_char data	8-bit data (ox00 to 0xff)

Description

The LongCommand type is used for long-format individually addressed or broadcast (with *address* = 0) B-Channel commands.

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::ShortCommand

Synopsis

```
in vme_rcc.h:

typedef u_char ShortCommand;
```

Fields

none

Description

The ShortCommand type is used for short-format (8-bit) broadcast B-Channel commands.

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

3.3 General Methods

TTCVI::TTCVI()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
TTCVI::TTCVI(u_int vmebus_address);
```

Parameters

u_int vmebus_address	in	VMEbus base address of TTCvi

Description

The TTCVI constructor creates a TTCVI object. It allocates a VMEbus master mapping starting at VMEbus address *vmebus_address*. The size of the master mapping is constant (0x100). The constructor also obtains the type of the TTCvi (Mk I or Mk II) by testing the trigger word address register which only exists for the Mk II.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::~TTCVI()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
TTCVI::~TTCVI(void);
```

Parameters

none

Description

The TTCVI destructor deletes the TTCVI object. It releases the VMEbus master mapping allocated for the TTCvi by the constructor, cf. page 8.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::reset()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::reset(void);
```

Parameters

none

Description

The *reset*() method resets the TTCvi by writing to the software module reset register.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::dump()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::dump(void);
```

Parameters

none

Description

The *dump*() method dumps all registers, interprets the values and prints the results to **cout**.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

This method uses many of the other “*Get*” methods.

TTCVI::manufacturerGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::manufacturerGet(u_int* manufacturer);
```

Parameters

u_int* manufacturer	out	manufacturer identifier (0x00800030)

Description

The *manufacturerGet*() method gets the manufacturer identifier from the configuration/identification EEPROM. The value of the manufacturer identifier is constant (0x00800030).

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/RCDTtcvi.cc, TTCVI::dump()* method.

Notes

none

TTCVI::boardIdentifierGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::boardIdentifierGet(u_int* board_identifier);
```

Parameters

u_int* board_identifier	out	board identifier (ttttssss, see description)

Description

The *boardIdentifierGet*() method gets the board identifier/serial number from the configuration/identification EEPROM. The value of the board identifier contains four digits for the date when the board was tested and four digits for the serial number (ttttssss).

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/RCDTtcvi.cc*, *TTCVI::dump()* method

Notes

none

TTCVI::boardRevisionGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::boardRevisionGet(u_int* board_revision);
```

Parameters

u_int* board_revision	out	board revision (yyyymmdd)

Description

The *boardRevisionGet*() method gets the board revision from the configuration/identification EEPROM. The value of the board revision is the date (yyyymmdd) when the configuration/ identification EEPROM was programmed.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/RCDTtcvi.cc*, *TTCVI::dump()* method.

Notes

none

TTCVI::mkTypeGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::mkTypeGet(int* mk_type);
```

Parameters

int* mk_type	out	Mk type of the board

Description

The *mkTypeGet*() method reads the type of the board obtained by the constructor, cf. page 8. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used:

TTCVI::MK_TYP1	Mk I type
TTCVI::MK_TYP2	Mk II type

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bcDelayGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bcDelayGet(int* bc_delay);
```

Parameters

int* bc_delay	out	BC delay in ns

Description

The *bcDelayGet*() method reads the BC delay from the CSR1 register. The value read is given in nanoseconds.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::orbitInputSet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::orbitInputSet(u_short orbit_input_selection);
```

Parameters

u_short orbit_input_selection	in	Orbit input selection

Description

The *orbitInputSet*() method selects the input for the ORBIT generation. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used:

TTCVI::ORB_INT	internal ORBIT generation
TTCVI::ORB_EXT	external ORBIT generation (front panel)

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::orbitInputGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::orbitInputGet(u_short* orbit_input_selection);
```

Parameters

u_short* orbit_input_selection	out	Orbit input selection

Description

The *orbitInutGet*() method reads the input selection for the ORBIT generation. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used:

TTCVI::ORB_INT	internal ORBIT generation
TTCVI::ORB_EXT	external ORBIT generation (front panel)

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *menuRCDTtcvi.cc*.

Notes

none

TTCVI::triggerWordEnable()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::triggerWordEnable(LongCommand& command);
```

Parameters

LongCommand& command	in	data structure used for address, external flag and sub-address

Description

The *triggerWordEnable*() method sets the address, external flag and sub-address for the transfer of the Trigger Word and of the Event/Orbit counter value after each L1A. Only the bits [7..2] of the sub-address are actually being used; the other bits [1..0] are set to zero by the TTCvi. The method enables the transfer of the Trigger Word and Event/Orbit counter value after each L1A by setting the size field of the trigger word register to 1.

Return Values

EPERM	This method is not permitted for an Mk I.
all return values of the RCDVme library, cf. [2]	

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::triggerWordDisable()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::triggerWordDisable(void);
```

Parameters

none

Description

The *triggerWordDisbale*() method disables the transfer of the Trigger Word and of the Event/ Orbit counter value after each L1A by setting the size field in the trigger word register to 0.

Return Values

EPERM	This method is not permitted for an Mk I.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::triggerWordGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::triggerWordGet(LongCommand* command, bool* enable_flag);
```

Parameters

LongCommand& command	out	data structure used for address, external flag and sub-address
bool enable_flag	out	flag to enable (true) or disable (false) the trigger word

Description

The *triggerWordGet*() method reads the address, external flag, sub-address and enable flag (= size field) for the transfer of the Trigger Word and of the Event/Orbit counter value after each L1A. Only the bits [7..2] of the sub-address are actually being used. The other bits [1..0] are set to zero by the TTCvi.

Return Values

EPERM	This method is not permitted for an Mk I.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::counterValueGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::counterValueGet(int* counter_value);
```

Parameters

int* counter_value	out	Event/Orbit counter value

Description

The *counterValueGet*() method reads the value of the Event/Orbit counter.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

For an Mk I the Event/Orbit counter only counts events (=L1As).

TTCVI::counterSelectionSet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::counterSelectionSet(u_short counter_selection);
```

Parameters

u_short counter_selection	in	Event/Orbit counter selection

Description

The *counterSelectionSet*() method selects the type for the Event/Orbit counter. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used:

TTCVI::CNT_ORB	Orbit counter
TTCVI::CNT_L1A	Event counter

Return Values

EPERM	This method is not permitted for an Mk I.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::counterSelectionGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::counterSelectionGet(u_short* counter_selection);
```

Parameters

u_short* counter_selection	out	Event/Orbit counter selection

Description

The *counterSelectionGet*() method reads the type for the Event/Orbit counter. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used:

TTCVI::CNT_ORB	orbit counter selection
TTCVI::CNT_L1A	event (L1A) counter selection

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

For an Mk I this method always returns TTCVI::CNT_L1A.

TTCVI::counterReset()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::counterReset(void);
```

Parameters

none

Description

The *counterReset*() method resets the Event/Orbit counter in the counter reset register.

Return Values

EPERM	This method is not permitted for an Mk I.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

3.4 Level-1 Accept Methods

TTCVI::l1aInputSet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::l1aInputSet(u_short l1a_input_selection);
```

Parameters

u_short l1a_input_selection	in	L1A input selection

Description

The *l1aInputSet*() method selects the input for the L1A generation. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used:

TTCVI::L1A_EXT0	external L1A generation, front panel input L1A<0>
TTCVI::L1A_EXT1	external L1A generation, front panel input L1A<1>
TTCVI::L1A_EXT2	external L1A generation, front panel input L1A<2>
TTCVI::L1A_EXT3	external L1A generation, front panel input L1A<3>
TTCVI::L1A_VME	internal ORBIT generation, using VME command
TTCVI::L1A_RNDM	internal ORBIT generation, using random generator

Return Values

EINVAL	The L1A input selection value is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::l1aInputGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::l1aInputGet(u_short* l1a_input_selection);
```

Parameters

u_short* l1a_input_selection	out	L1A input selection

Description

The *l1aInputGet*() method reads the input selection for the L1A generation. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used:

TTCVI::L1A_EXT0	external L1A generation, front panel input L1A<0>
TTCVI::L1A_EXT1	external L1A generation, front panel input L1A<1>
TTCVI::L1A_EXT2	external L1A generation, front panel input L1A<2>
TTCVI::L1A_EXT3	external L1A generation, front panel input L1A<3>
TTCVI::L1A_VME	internal ORBIT generation, using VME command
TTCVI::L1A_RNDM	internal ORBIT generation, using random generator

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *menuRCDTtcvi.cc*.

Notes

none

TTCVI::l1aRandomSet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::l1aRandomSet(u_short l1a_random_selection);
```

Parameters

u_short l1a_random_selection	in	L1A random generator frequency selection

Description

The *l1aRandomSet*() method selects the frequency for the L1A random generation. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used:

TTCVI::RNDM_1HZ	random generator frequency of 1 Hz
TTCVI::RNDM_100HZ	random generator frequency of 100 Hz
TTCVI::RNDM_1KHZ	random generator frequency of 1 kHz
TTCVI::RNDM_5KHZ	random generator frequency of 5 kHz
TTCVI::RNDM_10KHZ	random generator frequency of 10 kHz
TTCVI::RNDM_25KHZ	random generator frequency of 25 kHz
TTCVI::RNDM_50KHZ	random generator frequency of 50 kHz
TTCVI::RNDM_100KHZ	random generator frequency of 100 kHz

Return Values

EINVAL	The random generator frequency selection value is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::l1aRandomGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::l1aRandomGet(u_short* l1a_random_selection);
```

Parameters

u_short* l1a_random_selection	out	L1A random generator frequency selection

Description

The *l1aRandomGet*() method reads the frequency selection for the L1A random generation. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used:

TTCVI::RNDM_1HZ	random generator frequency of 1 Hz
TTCVI::RNDM_100HZ	random generator frequency of 100 Hz
TTCVI::RNDM_1KHZ	random generator frequency of 1 kHz
TTCVI::RNDM_5KHZ	random generator frequency of 5 kHz
TTCVI::RNDM_10KHZ	random generator frequency of 10 kHz
TTCVI::RNDM_25KHZ	random generator frequency of 25 kHz
TTCVI::RNDM_50KHZ	random generator frequency of 50 kHz
TTCVI::RNDM_100KHZ	random generator frequency of 100 kHz

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::l1aGenerate()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::l1aGenerate(void);
```

Parameters

none

Description

The *l1aGenerate*() method generates an L1A if the L1A input selection allows this (TTCVI::L1A_VME), cf. *l1aInputSet*() on page 26.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::l1aFifoEmpty()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::l1aFifoEmpty(bool* fifo_empty);
```

Parameters

bool* fifo_emtpy	out	empty flag of L1A FIFO

Description

The *l1aFifoEmpty*() method reads the empty flag of the L1A FIFO from the CSR1 register.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::l1aFifoFull()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::l1aFifoFull(bool* fifo_full);
```

Parameters

bool* fifo_full	out	full flag of L1A FIFO

Description

The *l1aFifoFull*() method reads the full flag of the L1A FIFO from the CSR1 register.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::l1aFifoReset()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::l1aFifoReset(void);
```

Parameters

none

Description

The *l1aFifoReset*() method resets the L1A FIFO in the CSR1 register.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

3.5 B-Channel Methods

TTCVI::bgoModeSet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoModeSet(int channel_number, u_short channel_mode);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)
u_short channel mode	in	B-Go channel mode

Description

The *bgoModeSet*() method selects the mode for the B-Go channel number *channel_number*. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used. They are inverted with respect to what gets written to the register because of the negative definitions chosen for the different fields of the register:

TTCVI::BGO_ENABLE	enable external B-go input, front panel B-Go input
TTCVI::BGO_SYNC	generate synchronous cycle, at the end of inhibit signal
TTCVI::BGO_SINGLE	generate single cycles, as opposed to repetitive cycles
TTCVI::BGO_FIFO	generate cycle as soon as FIFO non-empty
TTCVI::BGO_CALIB	calibration mode, only for B-Go channel 2, others must be set to “0”

Return Values

EINVAL	The B-Go channel number or mode are not valid.
EPERM	The B-Go channel mode (calibration) is not permitted for the B-Go channel or for the Mk I.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoModeGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoModeGet(int channel_number, u_short channel_mode);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)
u_short* channel mode	out	B-Go channel mode

Description

The *bgoModeGet*() method reads the mode for the B-Go channel number *channel_number*. The following symbolic constants, defined in *RCDTtc/RCDTtcvi.h*, have to be used. They are inverted with respect to what get written to the register because of the negative definitions chosen for the different fields of the register:

TTCVI::BGO_ENABLE	enable external B-go input, front panel B-Go input
TTCVI::BGO_SYNC	generate synchronous cycle, at the end of inhibit signal
TTCVI::BGO_SINGLE	generate single cycles, as opposed to repetitive cycles
TTCVI::BGO_FIFO	generate cycle as soon as FIFO non-empty
TTCVI::BGO_CALIB	calibration mode, only for B-Go channel 2

Return Values

EINVAL	The B-Go channel number is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoCommandPut()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoCommandPut(int channel_number, LongCommand& command);
u_int TTCVI::bgoCommandPut(int channel_number, ShortCommand& command);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)
LongCommand& command or ShortCommand& command	in	B-Go channel command

Description

The *bgoCommandPut*() method loads the B-channel command *command* into the B-Go channel FIFO *channel_number*.

Return Values

EINVAL	The B-Go channel number is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoGenerate()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoGenerate(int channel_number);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)

Description

The *bgoGenerate*() method generates a B-channel command in the B-Go channel number *channel_number* if it is set up to do so, i.e. TTCVI::BGO_ENABLE is NOT selected, cf. *bgo-ModeSet*() on page 34.

Return Values

EINVAL	The B-Go channel number is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoInhibitOn()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoInhibitOn(int channel_number, u_short inhibit_delay,
u_short inhibit_duration);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)
u_short inhibit_delay	in	B-GO inhibit delay in BCs (0 to 0xfff)
u_short inhibit_duration	in	B-Go inhibit duration in BCs (0 to 0xff)

Description

The *bgoInhibitOn*() method switches the inhibit signal associated to B-Go channel number *channel_number* on. It first sets the delay to *delay*, then the duration to *duration*. This avoids spurious inhibit signals at *delay = 0*.

Return Values

EINVAL	The B-Go channel number, the inhibit delay or the inhibit duration are not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoInhibitOff()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoInhibitOff(int channel_number);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)

Description

The *bgoInhibitOff*() method switches the inhibit signal associated to B-Go channel number *channel_number* off. It first sets the duration to 0, then the delay to 0. This avoids spurious inhibit signals at *delay = 0*.

Return Values

EINVAL	The B-Go channel number is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoInhibitGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoInhibitGet(int channel_number, u_short* inhibit_delay,
u_short* inhibit_duration);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)
u_short* inhibit_delay	out	B-GO inhibit delay in BCs, (0 to 0xfff)
u_short* inhibit_duration	out	B-Go inhibit duration in BCs (0 to 0xff)

Description

The *bgoInhibitGet*() method reads the duration and delay parameters for the inhibit signal associated to B-Go channel number *channel_number*.

Return Values

EINVAL	The B-Go channel number is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoFifoEmtpy()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoFifoEmpty(int channel_number, bool* fifo_empty);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)
bool* fifo_emtpy	out	empty flag of L1A FIFO

Description

The *bgoFifoEmpty*() method reads the empty flag of the FIFO associated to the B-Go channel *channel_number* from the CSR2 register.

Return Values

EINVAL	The B-Go channel number is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoFifoFull()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoFifoFull(int channel_number, bool* fifo_full);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)
bool* fifo_full	out	full flag of L1A FIFO

Description

The *bgoFifoFull*() method reads the full flag of the FIFO associated to the B-Go channel *channel_number* from the CSR2 register.

Return Values

EINVAL	The B-Go channel number is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoFifoRetransSet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoFifoRetransmitSet(int channel_number, bool
retransmit_flag);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)
bool retransmit_flag	in	B-Go channel FIFO retransmission flag

Description

The *bgoFifoRetransSet*() method selects the retransmission mode of the FIFO associated to the B-Go channel *channel_number* in the CSR2 register.

Return Values

EINVAL	The B-Go channel number is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoFifoRetransGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoFifoRetransmitGet(int channel_number, bool
retransmit_flag);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)
bool retransmit_flag	out	B-Go channel FIFO retransmission flag

Description

The *bgoFifoRetransGet*() method reads the retransmission mode of the FIFO associated to the B-Go channel *channel_number* in the CSR2 register.

Return Values

EINVAL	The B-Go channel number is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::bgoFifoReset()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::bgoFifoReset(int channel_number);
```

Parameters

int channel_number	in	B-Go channel number (0 to 3)

Description

The *bgoFifoReset*() method resets the B-Go channel FIFO *channel_number* in the CSR2 register.

Return Values

EINVAL	The B-Go channel number is not valid.
all return values of the RCDVme library, cf. [2]	

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::asyncCommand()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::asyncCommand(LongCommand& command);
u_int TTCVI::asyncCommand(ShortCommand& command);
```

Parameters

LongCommand& command or ShortCommand& command	in	B-channel command

Description

The *asyncCommand*() method generates the asynchronous B-channel command *command*.

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

TTCVI::asyncPendingGet()

Synopsis

```
#include “RCDTtc/RCDTtcvi.h”
u_int TTCVI::asyncPendingGet(bool* vme_pending);
```

Parameters

bool* vme_pending	out	flag for pending asynchronous B-channel command (VMEbus)

Description

The *asyncPendingGet*() method reads the transfer pending flag for asynchronous B-Channel commands from the CSR1 register (VME pending field).

Return Values

all return values of the RCDVme library, cf. [2]

Programming Example

For a programming example see *src/test/menuRCDTtcvi.cc*.

Notes

none

4 Programming Example

A programming example for the use of the TTCvi library of ROD Crate DAQ is available in the test program **src/test/testRCDTtcvi.cc**, see also Section 5.

5 Test Program

testRCDTtcvi provides a text-driven menu for reading and writing a TTCvi. It is intended for low-level interactive communication with the TTCvi.

Typing a number at the command prompt will execute the corresponding method or sub-menu of the menu printed above the command prompt. If parameters are required for the selected method they will be prompted for. The return code of the method will be printed. Ctrl-C can be used at any time to interrupt the execution of a method.

6 Known limitations

- Return values:
  The return values of the class methods shall in the near future be replaced by using the C++ exception mechanism.
- Testing:
  The TTCvi library of ROD Crate DAQ has been tested to work correct as far as transfers over the VMEbus are concerned. The general methods and the inhibit signal handling methods have been tested with an Mk I. B-channel commands and full functionality of an Mk II have **not been tested**!
- Bug reports:
  Please send any bug reports or requests for modifications of the TTCvi library of ROD Crate DAQ to Ralf.Spiwoks@cern.ch.

7 Acknowledgements

The author wishes to acknowledge the effort of M. Joos (VMEbus), P. Gallno (TTCvi) and I. Papadopoulos (TDAQ DataFlow software repository) without which this library could not have been developed.

8 References

[1] P. Gallno, TTC-VMEbus Interface (TTCvi), EDMS project CERN-0000002700, https://edms.cern.ch/project/CERN-0000002700.

[2] R. Spiwoks et al., VMEbus Application Program Interface, ATLAS EDMS document ATL-D-ES-0004, https://edms.cern.ch/document/325729.

[3] The ATLAS TDAQ Read-out System, Code Repository, http://atlas.web.cern.ch/Atlas/GROUPS/DAQTRIG/ROS/code_repository.