ACDC DAQ Programers Manual

Jonathan Eisch, Miles Lucas, Eric Oberla and Others

Contents

1 Introduction

This document is intended to document the commands sent to the ACC and ACDC cards.

It includes source listings of the source and destinations for each command as they were when this documentation was written. New implementations may have been written since, but that is outside the scope of this document.

The command protocol for each 32-bit instruction set is shown in Figure 1.

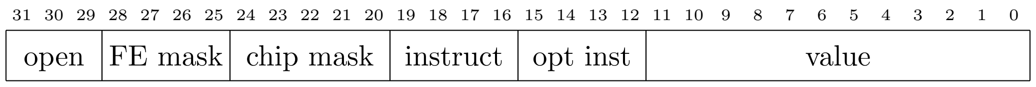

Figure 1: Command protocol

This is parsed on the front-end board like this

```
--parse 32 bit instruction word:
INSTRUCT_PSEC_MASK       <= xINSTRUCT_WORD(24 downto 20);
INSTRUCTION              <= xINSTRUCT_WORD(19 downto 16);
INSTRUCTION_OPT          <= xINSTRUCT_WORD(15 downto 12);
INSTRUCT_VALUE           <= xINSTRUCT_WORD(11 downto 0);
```

Table 1 lists all the instruction flags passed by software.

The instruction for setting self-trigger instructions has many optional instructions, listed in Table 2.

Table 1: List of Instructions

Bits	Description
0x0	Do nothing
0x1	Set dealy-locked loop VDD control voltage
0x2	Calibration pulse switch enable
0x3	Set pedestal
0x4	Reset DLL
0x5	Reset internal trigger
0x6	Set self trigger mask
0x7	Set self trigger instructions
0x8	Set trigger threshold
0x9	Adjust ring oscillation frequency
0xA	Enable/disable on-board LEDs
0xB	(DC) Central card FIFO toggle
0xB	(CC) Central card done
0xC	(CC) Central card read mode
0xD	(CC) Align/setup SERDES
0xE	(CC) USB trigger
0xF	(CC) Sync USB

Table 2: Optional instructs for 0x7 - set self-trigger instructions

Bits	Description
0x0	enable
0x1	wait for system trigger
0x2	Measure rate only
0x3	Trigger sign

2 Instructions

2.1 Template

2.1.1 Bit Fields

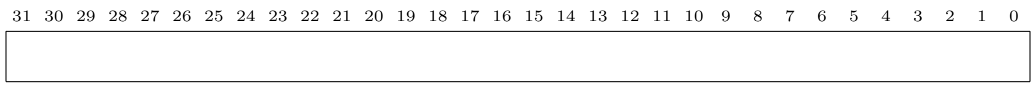

Figure 2: Command ... bit fields

Bits	Name	Description
31-0	Bits	

2.1.2 Source

project:path/to/filename.cpp

2.1.3 Destination

project:path/to/filename.vhd

2.2 0xF - Sync Usb

This command does something regarding to syncing the usb

2.2.1 Bit Fields

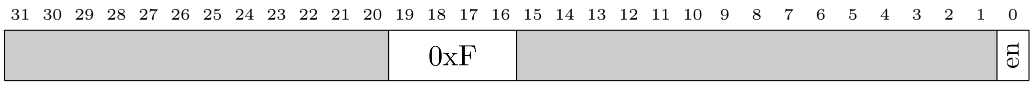

Figure 3: Command 0xF bit fields

Bits	Name	Description
19-16	**0xF**	Command marker
0	enable	Enable USB sync

2.2.2 Source

acdc-daq:src/DAQinstruction.cpp

```
void SuMo::sync_usb(bool SYNC) {
      createUSBHandles();
      if(SYNC != false) { //enable USB_SYNC
           usb.sendData((unsigned int)0x000F0001);
           if(mode == USB2x) usb2.sendData((unsigned int)0x000F0001
              ↪ );
      } else { //disable USB_SYNC
           usb.sendData((unsigned int)0x000F0000);
           if(mode == USB2x) usb2.sendData((unsigned int)0x000F0000
              ↪ );

      }
      closeUSBHandles();
}
```

2.2.3 Destination

2.3 0xD - Align LVDS

This command aligns the LVDS system between the central card and any acdc boards. The LVDS system is the RJ-45 connection.

2.3.1 Bit Fields

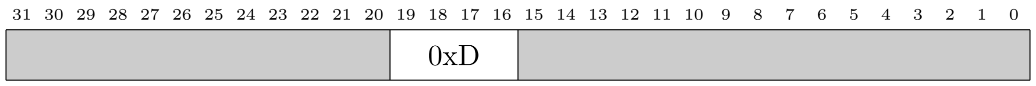

Figure 4: Command 0xD bit fields

Bits	Name	Description
19-16	**0xD**	Command marker

2.3.2 Source

acdc-daq:src/DAQinstruction.cpp

```cpp
void SuMo::align_lvds()
{
	createUSBHandles();
	usb.sendData((unsigned int)0x000D0000);   //toggle align process
	if(mode == USB2x) usb2.sendData((unsigned int)0x000D0000);
	closeUSBHandles();
}
```

2.3.3 Destination

2.4 0xA - Toggle LED

This command toggles the LED on all connected boards.

2.4.1 Bit Fields

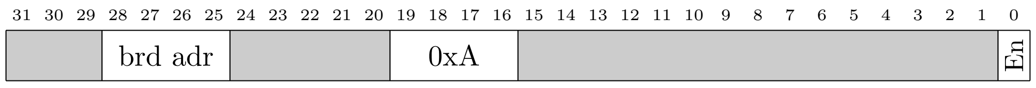

Figure 5: Command 0xA bit fields

Bits	Name	Description
28-25	board address	The address of the board. Default is 0xF
19-16	**0xA**	Command marker
0	enable	Enable the leds

2.4.2 Source

acdc-daq:src/DAQinstruction.cpp

```cpp
void SuMo::toggle_LED(bool EN)
{
    unsigned int boardAdr_all = 15;

    createUSBHandles();
    unsigned int send_word = 0x000A0000;
    send_word = send_word | boardAdr_all << boardAdrOffset;

    if(EN != false){
      usb.sendData(send_word | 0x1);
      if(mode == USB2x) usb2.sendData(send_word | 0x1);
    }
    else{
      usb.sendData(send_word);
      if(mode == USB2x) usb2.sendData(send_word);
    }
    closeUSBHandles();
}
```

2.4.3 Destination

2.5 0x7 - Set Self-trigger Lo

This command sends certain trigger commands

2.5.1 Bit Fields

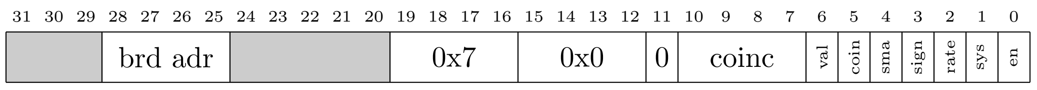

Figure 6: Command 0x7 bit fields

Bits	Name	Description
28-25	board address	The address of the board. Default is 0xF
19-16	**0x7**	Command marker
15-12	**0x0**	Optional command marker
10-7	coincidence window	The window of unknown units for the coincidence (<15)
6	use trig valid	Use trig valid as a reset on AC/DC
5	use coincidence	Use channel coincidence
4	use board sma trig	Use SMA input on AC/DC board for trigger
3	trig sign	1 for rising edge, 0 for falling edge
2	rate only	
1	sys trig option	
0	trig enable	Enables self-trigger

2.5.2 Source

acdc-daq:src/DAQinstruction.cpp

```
void SuMo::set_self_trigger_lo(       bool ENABLE_TRIG,
                                      bool SYS_TRIG_OPTION,
                                      bool RATE_ONLY,
                                      bool TRIG_SIGN,
                                      bool USE_BOARD_SMA_TRIG,
                                      bool USE_COINCIDENCE,
                                      bool USE_TRIG_VALID_AS_RESET,
                                      unsigned int coinc_window,
                                      unsigned int boardAdr,
                                      int device)
{
    const unsigned int hi_cmd = 0x00070000;
    unsigned int send_word = hi_cmd | 0 << 11
       | USE_TRIG_VALID_AS_RESET << 6
       | USE_COINCIDENCE << 5
       | USE_BOARD_SMA_TRIG << 4
       | TRIG_SIGN << 3 | RATE_ONLY << 2
       | SYS_TRIG_OPTION << 1 | ENABLE_TRIG
       | coinc_window << 7
       | boardAdr << boardAdrOffset;
    //printf("%i\n", send_word);

    createUSBHandles();

    if(device == 0)                         usb.sendData((unsigned int)
       ↪ send_word);
    if(device == 1 && mode == USB2x) usb2.sendData((unsigned int)
       ↪ send_word);

    closeUSBHandles();
}
```

2.5.3 Destination

2.6 0x7 - Set Self-trigger hi

This command sends certain trigger commands

2.6.1 Bit Fields

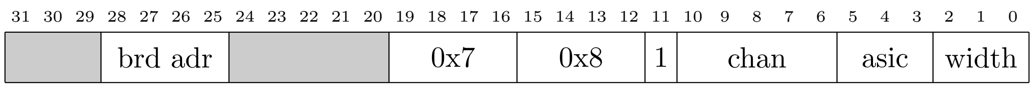

Figure 7: Command 0x7 bit fields

Bits	Name	Description
28-25	board address	The address of the board.
19-16	**0x7**	Command marker
15-12	**0x8**	Optional command marker
10-6	channel coincidence min	Number of coincident channels to enable trigger (<30)
5-3	asic coincidence min	Number of coincident asic chips to enable trigger (<5)
2-0	coincidence pulse width	Width of coincidence pulse in unknown units (<7)

2.6.2 Source

acdc-daq:src/DAQinstruction.cpp

```
void SuMo::set_self_trigger_hi(unsigned int coinc_pulse_width,
        unsigned int asic_coincidence_min,
        unsigned int channel_coincidence_min,
        unsigned int boardAdr,
        int device)
        {
        const unsigned int hi_cmd = 0x00078000;
        unsigned int send_word = hi_cmd | 1 << 11
        | channel_coincidence_min << 6
        | asic_coincidence_min << 3
        | coinc_pulse_width
        | boardAdr << boardAdrOffset;
        //printf("%x\n", send_word);

        createUSBHandles();
        if(device == 0) usb.sendData((unsigned int)send_word);
        if(device == 1 && mode == USB2x) usb2.sendData((unsigned
            ↪ int)send_word);

        closeUSBHandles();
}
```

2.6.3 Destination

2.7 0xA - Read ACDC Ram

2.7.1 Bit Fields

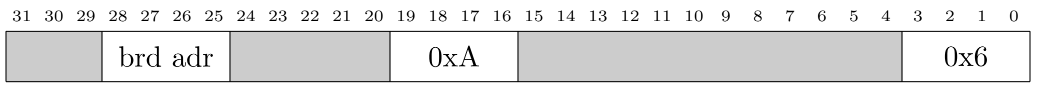

Figure 8: Command 0xA bit fields

Bits	Name	Description
28-25	board address	The address of the board (default = 15)
19-16	**0xA**	Command marker
3-0	**0x6**	Optional command marker

2.7.2 Source

acdc-daq:src/DAQinstruction.cpp

```
void SuMo::readACDC_RAM(int device, unsigned int boardAdr)
{
        unsigned int boardAdr_override  = 15;

        createUSBHandles();
        unsigned int send_word = 0x000A0006;
        send_word = send_word | boardAdr << boardAdrOffset;

        if(device == 0)                       usb.sendData(send_word);
        if(device == 1 && mode==USB2x)        usb2.sendData(send_word);

        closeUSBHandles();
}
```

2.7.3 Destination

project:path/to/filename.vhd

```
```

2.8 0x2 -Toggle Cal

2.8.1 Bit Fields

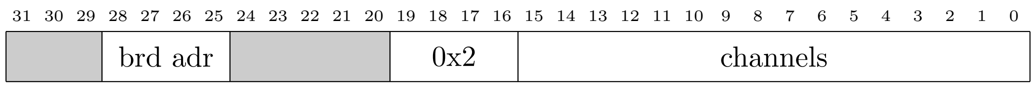

Figure 9: Command 0xA bit fields

Bits	Name	Description
28-25	board address	The address of the board (default = 15)
19-16	**0x2**	Command marker
3-0	channels	Channels (default = 0x7FFF)

2.8.2 Source

acdc-daq:src/DAQinstruction.cpp

```
void SuMo::toggle_CAL(bool EN,  int device)
{
        createUSBHandles();

        unsigned int send_word = 0x00020000;
        unsigned int channels  = 0x7FFF;
        unsigned int boardAdr  = 15;

        if(EN){
                send_word = send_word | boardAdr << boardAdrOffset | channels;
                if(device == 0)                 usb.sendData(send_word);
                if(device == 1 && mode==USB2x)  usb2.sendData(send_word);
        } else{
                send_word = send_word | boardAdr << boardAdrOffset;
                if(device == 0)                 usb.sendData(send_word);
                if(device == 1 && mode==USB2x)  usb2.sendData(send_word);
```

```
            }
closeUSBHandles();
}
```

2.8.3 Destination

project:path/to/filename.vhd

```
```