Radmu DAQ

Zynq Ultrascale+

Software Reference Manual

INFN Padova

By Lorenzo Castellani

The interface consists of a web page that implements a series of forms for setting the operating parameters, access to all the registers of the various chips and a WebSocket Server (RFC 6455) which responds on port 4444.
LabView library to communicate with the server is available
https://www2.pd.infn.it/~caste/pub/WebSockets.zip
Also there are a servers for RO data on port 3333.

Preliminary commands implemented.

Binary commands accepted by server are:

Commands implemented with WebSocket opt code = binary frame.
The format of the frames: [cmd] [arg1] .... [argn].
The first byte represents the command followed by the arguments.
The following structures describe the arguments for each command, UINT32 represents an integer 32bit (4 bytes) in little-Endian format, FLOAT64 an 8byte floating point in little-Endian format.

Operation, code 0x01

Manual operation for data link phase adjust and enable and disable channels.

```
BYTE cmd;

BYTE op;        //0=dec-delay, 1=inc-delay, 2=inc-pos,  4=enable, 5 disable, 6=enable sync,

                // 7 disable sync, 8 enable test, 9 disable test

BYTE ch;        //channel 0-24
```

Return : the status frame, see Status command

SetID, code 0x31

Set the TTCID connected on the RJ45 channel 0-7

```
BYTE cmd;

BYTE ch;        //channel 0-7

BYTE ID;        //TTC ID from 0 to 32
```

Return :

```
BYTE cmd;       //0xB1

BYTE ID[8];     //the 8 TTC ID (-1 not connected)
```

SetTRG CFG, code 0x34

Set trigger CFG

```
BYTE cmd;

UINT32 cfg;     //trigger CFG

UINT32 en0;     //trigger enable

UINT32 en1;     //trigger enable

UINT32 en2;     //trigger enable

UINT32 en3;     //trigger enable
```

Return : the same frame.

GetTRG CFG, code 0xB4

Get trigger CFG

```
BYTE cmd;
```

Return :

```
BYTE cmd;       //0x34
UNIT32 cfg;     //the GFG
UINT32 en0;     //trigger enable
UINT32 en1;     //trigger enable
UINT32 en2;     //trigger enable
UINT32 en3;     //trigger enable
```

SetTTC CFG, code 0x32

Set TTC CFG

```
BYTE cmd;
UINT32 id
UINT32 enable[2];       //
BYTE monostable;
BYTE generic;           //struct _genregbits {
                        //BYTE monoretr : 1;
                        //BYTE sync1 : 1;
                        //BYTE sync2 : 1;
                        //BYTE sync3 : 1;
                        //BYTE pllclk : 1;
                        //BYTE test1 : 1;
                        //BYTE test2 : 1;
                        //BYTE test3 : 1;}
BYTE winwhidth;
BYTE winbefor;
UINT16 tp[2];
BYTE dummy;
BYTE Si5344reset;
```

Return : the same frame.

SetTTC IN Enable, code 0x32

Set TTC input Enable

```
BYTE cmd;

UINT32 id

UINT32 enable[2];        //
```

Return : the same frame of **SetTTC CFG** command

GetTTC CFG, code 0xB2

Get TTC CFG

```
BYTE cmd;

UINT32 id
```

Return :

```
BYTE cmd;                //0x32

UINT32 id

UINT32 enable[2];        //

BYTE monostable;

BYTE generic;            //struct _genregbits {
                         //BYTE monoretr : 1;
                         //BYTE sync1 : 1;
                         //BYTE sync2 : 1;
                         //BYTE sync3 : 1;
                         //BYTE pllclk : 1;
                         //BYTE test1 : 1;
                         //BYTE test2 : 1;
                         //BYTE test3 : 1;}

BYTE winwhidth;

BYTE winbefor;

UINT16 tp[2];

BYTE dummy;

BYTE Si5344reset;
```

Enable, code 0x27

Enable channel

BYTE cmd;

BYTE ch; // channel:

```
//ENABLE(31)  Enable Merge TTC AB from GTT
//ENABLE(30)  Enable L1A output to GTT (0=Three-state)
//ENABLE(29)  Sel L1A 0=L1A from Artix7 1=CPU generated by pos_pulse(31)
//ENABLE(28)  Disable L1A veso Artix7 tramite TTC locale (verso GTT
//sempre abilitato)
//ENABLE(27)
//ENABLE(26)
//ENABLE(25)
//ENABLE(24)
//ENABLE(23-0)      Enable Link
```

Return : the same frame.

Disable, code 0x28

Disable channel

BYTE cmd;

BYTE ch; // channel :

```
//ENABLE(31)  Enable Merge TTC AB from GTT
//ENABLE(30)  Enable L1A output to GTT (0=Three-state)
//ENABLE(29)  Sel L1A 0=L1A from Artix7 1=CPU generated by pos_pulse(31)
//ENABLE(28)  Disable L1A veso Artix7 tramite TTC locale (verso GTT
//sempre abilitato)
//ENABLE(27)
//ENABLE(26)
//ENABLE(25)
//ENABLE(24)
//ENABLE(23-0)      Enable Link
```

Return : the same frame.

Soft L1A, code 0x29

Generate software L1A to TOF measure

BYTE cmd;

Return : the same frame.

Read TOF, code 0xB0

Read TOF measure, input delay used to measure resolution <1ns

```
BYTE cmd;
```

Return :

```
BYTE cmd;       //0x30
UNINT32 tof;    //bits 0-8 input delay, bits 9-25 delay ns
```

TTC broadcast, code 0x11

Set TTC broadcast cmd, use 0x04 for test-pulse

```
BYTE cmd;
BYTE val;       //TTC broadcast value
```

Return :

```
BYTE cmd;       //0x11
BYTE val;       //TTC broadcast value
```

TTC broadcast, code 0x25

Set TTC broadcast cmd, use 0x04 for test-pulse

```
BYTE cmd;
UNIT32 val;     //TTC broadcast value
```

Return :

```
BYTE cmd;       //0x11
UNIT32 val;     //TTC broadcast value
```

TTC, code 0x22

Set TTC cmd

```
BYTE cmd;
UINT32 val;     //TTC value
```

Return :

```
BYTE cmd;       //0x11
UINT32 val;     //TTC value
```

Link scan, code 0x06

Find and set delay for the data link (place link in test mode before send command)

BYTE cmd;

Return :

0x07,<byte Interface>,<uint32 Totalsize>,

<uint32 Namesize>,<Cstring>,

<uint32 Datasize>,<Data>,

<uint32 Namesize>,<Cstring>,

<uint32 Datasize>,<Data>,

…

…

<uint32 Maxrecord>

Totalsize: is the size in byte of data transferred, tag 0x02 included, in little Endian format.

Namesize: is the size in byte of the string name that identified the data, end string included (char=0) , in little Endian format.

Cstring: is the byte array contained the name of data .

Datasize: is the data size in byte.

Data: is the vector array. Vector_0 [x, y],Vector_1[x,y]…..

x and y are in double float precision (8 byte) and little Endian format.

Maxrecord: is the maximum number of vector return in data.

Status, code 0x82

Get the links status, if scanning is running return also scan data frame

```
BYTE cmd;
```

Return :

```
BYTE cmd;          //0x02
INT32 spydata[24];        //format 0xBBVVPDDD, BB = BER median on 1s, VV=data value, P= position
                          DDD = delay
INT32 enable;   //
INT32 sync;      //
INT32 test;      //
INT32 ErrFlag;  //
INT32 errcnt[24];
INT32 idtdc[8];           //tdc id assignement
INT32 tof;
BYTE pll_status;
INT32 pll_lose_lock;
BYTE pll_input;
```

Save, code 0x17

Save default configuration

```
BYTE cmd;
```

Return :

```
BYTE cmd;          //0x17
```

Si5338 PLL status and Xilinx Pll, code 0x9D

Get Si5338 PLL status

```
BYTE cmd;

BYTE reset;       //reset nLoseLock;
```

Return :

```
BYTE cmd;         //0x1D

BYTE pllstatus;  //0x00 = OK pll locked

INT32 nLoseLock;           // Lose lock number

BYTE pll_input;            //0=local 1=GTT
```

Temperature, code 0x9E

Get temperature sensor

```
BYTE cmd;
```

Return :

```
BYTE cmd;         //0x1E

FLOAT32 pl;       //Programmable Logic Temperature

FLOAT32 ps;       //Processor Syntem Temperature

FLOAT32 rem;   //Remote Temperature

FLOAT32 phy;   //Ethernet PHY Temperature
```

Select filtered Pll on articx7, code 0x24

```
BYTE cmd;

UNINT32 val;    //1 select PLL
```

Return : the same frame.

Clear Error Flag , code 0x26

```
BYTE cmd;

BYTE ch;          //link channel 0 24
```

Return : the same frame

Check PLL CFG, code 0xB3

BYTE cmd;

UNIT32 id;   //TTC id

Return :

BYTE cmd;   //0x33

BYTE err;

Artix7 Temperatute, code 0xB5

BYTE cmd;

UNIT32 id;   //TTC id

Return :

BYTE cmd;   //0x35

FLOAT32 temperature;

Init Artix7, code 0x23

Reconfigure PLL and Artix7 registers

BYTE cmd;

UNIT32 id;   //TTC id

Return:

BYTE 0x27;

Delay TP L1A, code 0x36

BYTE cmd;

BYTE delay;

Return: the same frame

Read Delay TP L1A, code 0xB6

BYTE 0x36;

BYTE delay;

SetFilter DisableW, code 0x39

BYTE cmd;

INT32 disablew[16];

Return: the same frame

GetFilter DisableW, code 0xB9

BYTE cmd;

Return:

BYTE cmd;   //0x39

INT32 disablew[16];

Copy TDC, code 0x38

BYTE cmd;

UINT32 id;   //id TDC

UINT32 enable[2];   //

BYTE monostable;

```
BYTE generic;             //struct _genregbits {
                          //BYTE monoretr : 1;
                          //BYTE sync1 : 1;
                          //BYTE sync2 : 1;
                          //BYTE sync3 : 1;
                          //BYTE pllclk : 1;
                          //BYTE test1 : 1;
                          //BYTE test2 : 1;
                          //BYTE test3 : 1;}
```

```
BYTE winwhidth;

BYTE winbefor;

UINT16 tp[2];

BYTE dummy;

BYTE Si5344reset;
```

Return:

BYTE cmd;   //0x38

Restore, code 0x37

```
BYTE cmd;
```

Return:

```
BYTE cmd;        //0x37
```

Write configuration File, code 0x41

Save configuration file

```
BYTE cmd;

CSTRING filename;       //C string with termination char 0. If string is empty write default
                        //configuration file

BYTE data[size];        //the configuration data
```

Return the same frame;

Read configuration File, code 0xC1

Read configuration file

```
BYTE cmd;

CSTRING filename;       //C string with termination char 0. If string is empty write default
```

Return:

```
BYTE cmd;               //0x41

CSTRING filename;       //C string with termination char 0. If string is empty is default configuration

BYTE data[size];        //the configuration data
```

Set configuration, code 0x42

Set configuration

```
BYTE cmd;

BYTE data[size];        //the configuration data
```

Return the same frame;

Get configuration, code 0xC2

Get configuration

```
BYTE cmd;
```

Return :

```
BYTE cmd;               //0x42

BYTE data[size];        //the configuration data
```

Return : Return data on error , code 0xFF:

```
BYTE cmd;       //0xFF

INT32 errorcode;  // Error code
```

Error code:

-1	Not Authorized
-22	Invalid Value
-5	I/O Error
-9	Unknow command (connection will be closed)
-2	no such file
-13	Permission denied
-16	Busy

See linux c/c++ error base for undefined number (errno-base.h)

TEXT commands accepted by server are:

Temperature?	Return the temperature
Version?	Return the App version string
Save [<filename>]	Save settings
Load [<filename>]	Load settings

LsCfg?	List setting config file
Restore	Restore memory actual setting (force reconfiguration of TDC)
SiStatus?	Show PLL status
VersionFPGA?	Return the FPGA version string an implementation type
VersionTDC?	Return all TDC FPGA version string and implementation type
VersionTDC <ID>	Return the TDC FPGA version string and implementation type
TTCDly?	Return the TTC input delay
TTCDly <ID> <dalay>	Set TTC input delay
LsFirm?	List DAQ firmware
Reboot <firmware>	Load and reboot with new firmware
Ls	list TDC firmware. Need authentication
auth	authentication (command implementation on web socket terminal)
ProgramTDC <id> <filename>	reprogram FLASH TDC firmware. Need authentication
WDisable <ch> <val>	Filter wire disable
PllChgIn	Change PLL input

DATA FORMAT

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	DAQ FF	TDC FF	BRD ID					CH						TIME(ns)															

DAQ FF: is the DAQ FIFO Full Flags and indicate that data are lose at DAQ level

TDC FF: is the TDC FIFO Full Flag and indicate that data are lose at TDC level

Event TAG (trigger MODE)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	Event number																												

Event Time(trigger MODE)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	TIME(ns) bit[2-0] always 0															

Dummy word

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Orbit TAG (all hit MODE)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	DAQ FF	TDC FF	BRD ID					CH						Orbit number															