ft2u: FTSW version 2.1 firmware for users

Mikihiko Nakao, KEK

January 3, 2014

- version 0.3 (2013.10.22) — for ft2u 0.35 first SVN release
- version 0.3.1 (2013.11.18) — for ft2u 0.38/ tt4r 0.18,
- version 0.3.2 (2014.1.3) — for ft2u 0.43/ tt4r 0.19, statft / stat4r

1 Introduction

The **ft2u** firmware for the **FTSW** (Frontend Timing SWitch) module version 2.1 provides communication between FTSW and frontend boards based on the **b2tt** (Belle II trigger timing) protocol, and communication between FTSW and TT-RX mounted on the COPPER board. The TT-RX uses the **tt4r** firmware which is included in this package and briefly described in this document, while the **b2tt** receiver firmware is described in a separate document.

The **ft2u** firmware runs either under a standalone mode or a controlled mode. The standalone mode requires no controlling VME CPU module, and is still capable to test the frontend system, to provide an external trigger, and to readout data from multiple boards with COPPER through Belle2link. The controlled mode requires a VME CPU and has access to all functions of **ft2u**, and is needed if it has to be controlled via NSM2.

There are quite a few functions of the **ft2u** firmware as listed below, and many of them can be used without an external VME control. The functions with **(*)** require a VME CPU.

System clock

- Generation and distribution the system clock of 127.216 MHz
- Alternatively distribution of a system clock generated elsewhere

Trigger signal

- Encoding the trigger signal from elsewhere as an LVDS signal
- Getting the trigger signal from the "TLU" module of the EUDAQ system (*)
- Or otherwise generation of a dummy trigger by the FTSW (*)
- Distribution of the trigger timing and trigger type with the event time and the trigger tag.

Synchronization

- System time distribution
- Initialization of the system time to a reference (otherwise time starts from 0 aka the beginning of year 1970 aka the Epoch) (*)
- Reset signal distribution (*)

Flow control

- Collecting all status information from connected frontend systems and backend COPPER systems to control the data flow

JTAG signals

- Parallel JTAG programming of all connected systems
- Parallel JTAG programming of selected systems (*)

Monitoring

- Collecting all other information that can be done with the **b2tt** protocol (*)

2 Firmware files

There are two FPGAs on FTSW version 2.1 and one FPGA on TT-RX version 4.5. All FPGAs have to be reprogrammed from what were previously there.

Spartan-3 — **ft2s** firmware has to be programmed from JTAG. The latest version is 0.46, or **ft2s046.bit**. It is not expected to be frequently updated.

Virtex-5 — **ft2u** firmware is the main firmware, which can be stored into configuration flash XCF16P from JTAG, or can be overwritten from VME. The latest version is 0.43 or **ft2u043.bit** as of writing, but it is expected to be frequently updated.

XCF16P — **ft2u043.mcs** has to be generated by Xilinx `impact` program, and programmed into XCF16P. The **Parallel Mode** option has to be enabled in the programming properties.

Building from source — provided distribution should have necessary options enabled, but do not forget to enable **Drive Done Pin High** option in the generation file properties, or otherwise the generated bit file cannot make the FPGA into DONE.

TT-RX — **tt4r** firmware for TT-RX version 4.5 (on the board it just says V4). The latest version is 0.19, or **tt4r019.bit**. Tools to download and control TT-RX is provided as **ttrxprogs** in a separate package.

3 On-board setup

As shown in Fig. 1, there are several switches and jumper pins that are needed for the **b2tt** firmware. Here is the summary.

S2,S1 rotary switches to define the board id in hexadecimal, which should coincide with the decimal board id label on the front panel. This is also a lower 8-bit of the VME base address. Do not touch.

S4,S3 rotary switches should be 0,3 to define the higher 8-bit of the VME base address. Do not touch unless it conflicts with other boards on the same VME bus.

S5 rotary switch is used to choose the clock source. 0 (default) to use the on-board clock, 1 to use external clock.

J1,J6,J7 jumpers are to include / exclude some of the devices. Do not change unless JTAG chain has to be investigated.

J5 jumpers have to be installed at 1P and 1N positions. 2P and 2N positions are unused.

J2 jumpers have to be inserted to use remote JTAG link through RJ-45. They have to be removed when JTAG programming is done through P5 connector.

J4,J3 are input pins of Xilinx flying wire pins of the JTAG cable. Connect TCK to TESTPIN-0, TMS to TESTPIN-1, TDI to TESTPIN-2, TDO to TESTPIN-3, VREF to J3-3.3V, and GND to J3-GND.

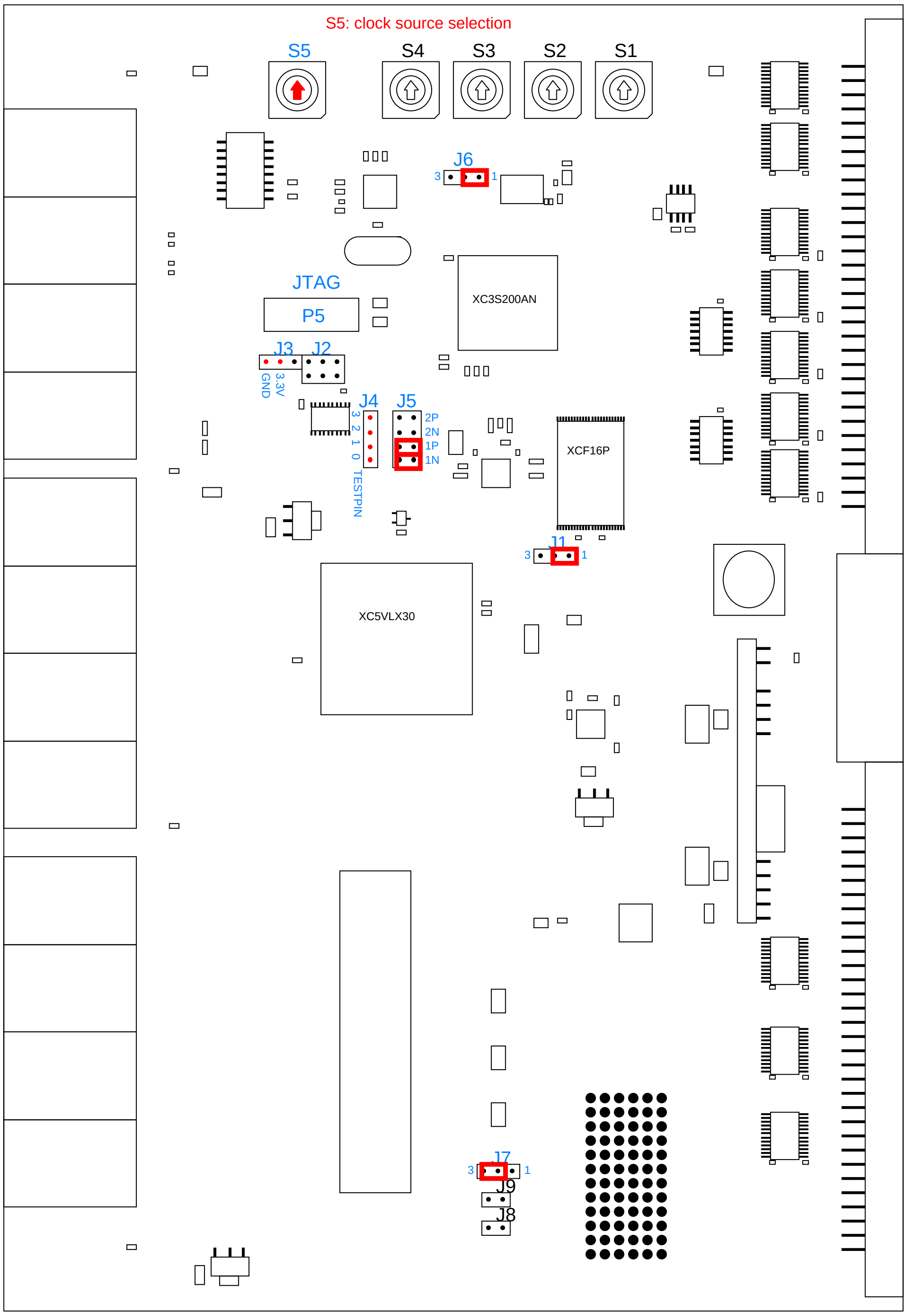

Figure 1: FTSW on-board setup for the `ft2u` firmware.

4 Front-panel: Cables, Connections, LEDs

FTSW port definition is given in Fig. 2, where **O**n is labeled as **OUT**-n on the front panel. Many FTSW boards of **type-R** do not have **O17–20** and some **type-P** FTSW do not have **O13–16** either, but the rest of functions are the same.

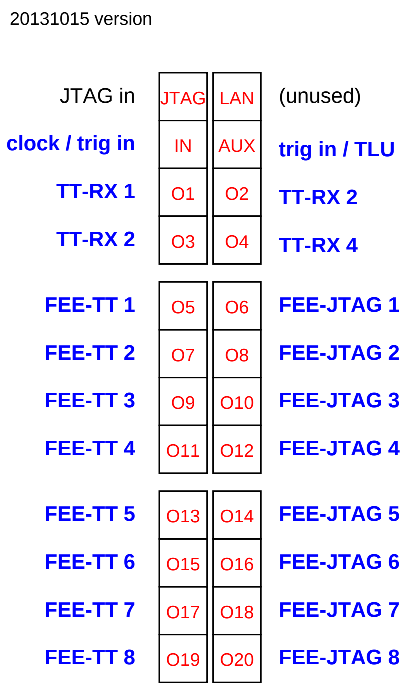

Figure 2: FTSW port definitions for the `ft2u` *firmware.*

The system clock **(sysclk)** of 127 MHz is generated inside FTSW, or an external clock can be fed from the 7-8 pair of **IN** port. The **sysclk** source is chosen by a rotary switch **S5** on the board: 0 for the on-board clock, 1 for the external clock. When a valid clock is not fed into FTSW, it may still look like working, but a self-oscillating clock of about 123 MHz is generated by the on-board jitter cleaner. Upper yellow LED at **AUX** indicates the clock status: LED is **on** when an external clock is chosen and valid, **blinking** (at 2 Hz) if the clock is not valid, and **off** when the on-board clock is chosen.

An external trigger signal to FTSW is an LVDS signal longer than one **sysclk** cycle of 7.8 ns. Only the rising edge is used, so it can be any length. Without a VME control, it is taken from **AUX** 7-8 pair. The **AUX** port can be also used for a connection to the TLU module from which the trigger is sourced. In this case the trigger signal is not mere a pulse, so a separate control is needed from VME.

The first four **O1–4** ports are for connections to the TTD ports of TT-RX cards on COPPER boards. Connections to the front-end electronics are taken from odd number (left-side) **O5–19** ports, while JTAG connections are to be taken from even number (right-side) **O6–20** ports. Each HSLB card requires **sysclk** at the RJ-45 port, which can be taken from any of **O1–4** or odd number **O5–19** ports.

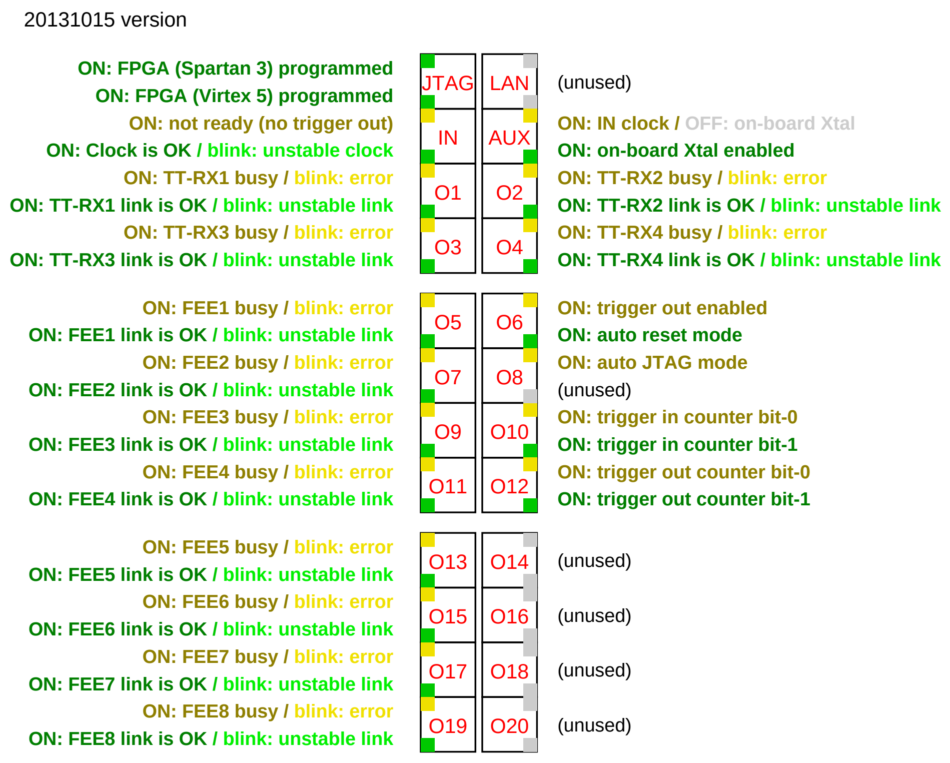
20131015 version
ON: FPGA (Spartan 3) programmed
ON: FPGA (Virtex 5) programmed
ON: not ready (no trigger out)
ON: Clock is OK / blink: unstable clock
ON: TT-RX1 busy / blink: error
ON: TT-RX1 link is OK / blink: unstable link
ON: TT-RX3 busy / blink: error
ON: TT-RX3 link is OK / blink: unstable link
JTAG
LAN
IN
AUX
O1
O2
O3
O4
(unused)
ON: IN clock / OFF: on-board Xtal
ON: on-board Xtal enabled
ON: TT-RX2 busy / blink: error
ON: TT-RX2 link is OK / blink: unstable link
ON: TT-RX4 busy / blink: error
ON: TT-RX4 link is OK / blink: unstable link
ON: FEE1 busy / blink: error
ON: FEE1 link is OK / blink: unstable link
ON: FEE2 busy / blink: error
ON: FEE2 link is OK / blink: unstable link
ON: FEE3 busy / blink: error
ON: FEE3 link is OK / blink: unstable link
ON: FEE4 busy / blink: error
ON: FEE4 link is OK / blink: unstable link
O5
O6
O7
O8
O9
O10
O11
O12
ON: trigger out enabled
ON: auto reset mode
ON: auto JTAG mode
(unused)
ON: trigger in counter bit-0
ON: trigger in counter bit-1
ON: trigger out counter bit-0
ON: trigger out counter bit-1
ON: FEE5 busy / blink: error
ON: FEE5 link is OK / blink: unstable link
ON: FEE6 busy / blink: error
ON: FEE6 link is OK / blink: unstable link
ON: FEE7 busy / blink: error
ON: FEE7 link is OK / blink: unstable link
ON: FEE8 busy / blink: error
ON: FEE8 link is OK / blink: unstable link
O13
O14
O15
O16
O17
O18
O19
O20
(unused)
(unused)
(unused)
(unused)

Figure 3: LED definitions for the `ft2u` *firmware.*

40 LEDs on 20 RJ-45 ports provide information about the FTSW, as given in Fig. 3.

- Both LEDs at **JTAG** should be **on**, or otherwise the board has to be re-programmed.
- Upper LED at **IN** should be **off**, or otherwise FTSW is **busy** and no trigger will be generated.
- Lower LED at **IN** should be **on**, or otherwise something is wrong in clock-ing.
- Upper LED at **AUX** tells the choice of the clock source.
- Lower LED at **AUX** tells the on-board oscillator is enabled. It is possible to disable the on-board oscillator when an external clock is provided, but it should be done with a great care.
- Upper LED at **O1–4** tells that the connected TT-RX is generating a busy signal (such as fifo is almost full). This could be one of the reason of FTSW **busy**.
- Lower LED at **O1–4** tells that the connection to TT-RX is established.
- Upper LED at odd-number **O5–19** tells that the connected front-end electronics is generating a busy signal (such as fifo is almost full) or an error. This could be another reason of FTSW **busy**.
- Lower LED at odd-number **O5–19** tells that the connection to the front-end electronics is established. If blinking, the frontend is receiving **sysclk** and saying that the **b2tt** link is not established there.
- Upper LED at **O6** tells that the trigger output is enabled. This should be enabled if it is not controlled from VME.

- Lower LED at **O6** tells that it is in the auto-reset mode, meaning that a reset signal is generated every time when a new connection is established. This is enabled when there is no control from VME.
- Upper LED at **O8** tells that it is in the auto-JTAG mode, meaning that the JTAG signals are distributed to all even-number **O6–20** ports, but the TDO output is taken only from the O6 port. This is enabled when there is no control from VME.
- Two LEDs at **O10** show the lowest 2-bit of the trigger **input** counter. These may be blinking too fast at a high rate trigger.
- Two LEDs at **O12** show the lowest 2-bit of the trigger **output** counter. These may be blinking too fast at a high rate trigger.

5 Software Tools

Software tools are provided to control the **b2tt** firmware, if a VME A32D32 (32-bit address 32-bit data) program-I/O access is possible from a VME controller that runs Linux or other Unix-like operating system. The tools are developed and tested on GE-Fanuc VMIVME 7807 CPU with the VMISFT-7433-3.6 driver, and it should be readily used on other supported VMIVME series.

The basic VME functions used in these tools are confined in ftswlib.c, and its variant were tested with other VME controllers such as Force CPU-5V running Solaris 2.5 and Motorola MVME187 running VxWorks 5.2. It should not be difficult to tailor for any other controllers. A simple Makefile is provided to build the programs.

Following software tools are provided in the package:

bootft to program the Virtex-5 FPGA of FTSW.

- usage: `bootft` -*id bitfile.bit*
- *id* is the board-id number that is given in the front panel and set in S2–S1 rotary switches. This *id* is also needed in all other tools.
- The latest *bitfile* is ft2u038.bit.
- After booting the *bitfile*, it sets the current time of the host system to FTSW. It waits for the timing when the second changes on the host, so it should have a precision much better than 1 second.

trigft to generate dummy triggers from FTSW.

- usage: `trigft` *-id (-option) type rate (count) (param)*
- *-option* list:
  - *-r* generates `runreset` before start (default)
  - *-n* does not generate `runreset` at start
  - *-b* generates begin-run-trigger (trigger type 0x0f) before start
  - *-f* fills FIFO to be read out **fifoft** or other programs
  - *-w* waits until all the triggers are processed or trigger is stopped for 3 seconds
- *type* is one of *none, aux, tlu, pulse, revo, random, poisson, one, stop*
  - *none* generates no trigger

- *aux* takes triggers from AUX 7-8 pair
- *tlu* takes triggers from AUX 7-8 pair according to the TLU protocol
- *pulse* generates dummy trigger at fixed interval, at all revolution positions. Maximum rate is about 100 kHz. Optionally, one can generate a bunch of several triggers instead of one. Lower 6 bit of *param* is the number of pulse in the bunch, and next 6 bit + 24 clock is the interval.
- *revo* generates dummy triggers at fixed interval, always at a fixed revolution position. Maximum rate is about 100 kHz, i.e., same as the revolution signal. *param* is to set the bunch position.
- *random* generates dummy triggers at random interval, where the random number is uniformly distributed.
- *poisson* generates dummy triggers at random interval, where the random number is distributed according to a Poisson distribution.

fifoft to read out FIFO data.

- usage: `fifoft -`*id num-event* `[-f [`*filename*`]]`
- This is to read FTSW's FIFO, to check the consistency with data read out by Belle2link.
- Program terminates after reading *num-event*
- Default output filename is constructed from date and time, unless `-f` *filename* is provided.
- If `-f` option is specified without *filename*, output is printed on screen.

regft to read and write any register of FTSW.

- The register map is given in the next section.
- Useful command will be
  - `regft -`*id* `200 1` (run reset)
  - `regft -`*id* `200 80000001` (run reset and read no FIFO)
  - `regft -`*id* `180 4` (disable 3rd FEE)
  - `regft -`*id* `1a0 4` (select 3rd FEE for JTAG)

statft to generate summary of **ft2u** registers.

- usage: `statft [-]`*id* `[-v] [-d] [-i] [-c]`
- *-option* list:
  - *-v* generates a full list
  - *-d* generates a dump of serial link
  - *-i* repeatedly show the register dump
  - *-c* color option: cyan when running, yellow while idle, red when busy or other problem
- Description of the registers is given in the next section.

6 VME address map

Under the VME base address, 64K byte memory space is allocated. All the registers are mapped at 16 byte steps, so up to 4096 register space is available. In reality, first 16 register space between 0x0000 and 0x00f0 is allocated for the Spartan-3 FPGA, and a space between 0x0100 and 0x09f0 is allocated for the Virtex-5 FPGA. The rest is currently unused.

addr	(mode)	**name**	description
0000	(r/w)	**ftswid**	initial value `0x46545357` (ASCII code for "FTSW"). Anything written into this register can be stored and read back to test VME access, but do not change.
0010	(r)	**cpldver**	bit 31..24: fixed value `0x03`, bit 23..16: board serial number, bit 15..0: CPLD firmware version
0020	(r/w)	**conf**	Virtex-5 FPGA configuration (no description is given here, please check the firmware source file and bootft program source file).
0030	(r/w)	**cclk**	bit 0: Virtex-5 CCLK/BUSY for configuration
0040	(r/w)	**clksel**	only for FTSW 2.1 and will be deprecated
0080	(r/w)	**sysrst**	defined for VME SYSRESET, but not tested
0100	(r/w)	**fpgaid**	initial value to identify firmware `0x46543255` ("FT2U") for **ft2u** Anything written into this register can be stored and read back, but do not change.
0110	(r)	**fpgaver**	bit 31..24: fixed value `0x03`, bit 23..16: board serial number, bit 15..0: FPGA firmware version
0120	(r/w)	**settime**	unix-time, i.e., time as the number of seconds since the Epoch, 1970-01-01 00:00:00 +0000 (UTC), as the reference for the time of the system, and also to record when the time was set
0130	(r/w)	**clkfreq**	bit 23..0: lowest 24-bit of 0x7952980 = 127216000 to define the clock frequency to increment utime counted in the unit of second (default value: 0x952980)
0140	(r/w)	**utime**	current unix-time counted by FTSW Writing any value to this register will hold the value of **utime**, **ctime**, **udead**, **cdead**, **tincnt** and **toutcnt** for one second, to properly calculate the trigger rate, dead time, etc. Read CTIME as the last register and checking bit-31 to be '1' to ensure the all register are held
0150	(r)	**ctime**	bit 31: values of **utime**, etc. are held bit 26..0: current sub-second time in clock unit
0160	(r)	**exprun**	bit 31..22: experiment number bit 21..0: run number This 32-bit word is transmitted to the frontend and attached to the data header
0180	(r/w)	**omask**	bit 15..12: COPPER mask (bit '1' to disable) bit 7..0: FEE mask (bit '1' to disable)
01a0	(r/w)	**jtag**	bit 8: to select auto-JTAG mode, in which output to all JTAG ports are enabled, and TDO is taken from **O6** only. bit 7..0: enable JTAG port when not auto-JTAG mode
01c0	(r/w)	**jset**	set jitter cleaner register (all 32 bit) writing `0000000e` to get register 0 at jget writing `0000002e` to get register 1 at jget writing `0000004e` to get register 2 at jget

			writing `4a200d50` to set register 0 (FTSW default) writing `03874061` to set register 1 (FTSW default) writing `0000001f` to store registers into EEPROM
01d0	(r)	**jget**	get jitter cleaner register register 0: 4a200d50 (FTSW), 72a000e0 (before config) (register 0: 4a201d50 to use 254MHz input) register 1: 83874061 (FTSW), 8389a061 (before config) register 2: (don't care)
01e0	(r/w)	**jpd**	bit 31: jitter cleaner power down bit 1: reset auto phase detection bit 0: disable auto phase detection
01f0	(r)	**jstat**	bit 31: jitter cleaner locked bit 30: phase detection DCM locked bit 27–24: clock phase (should be c) bit 23–16: phase reset retry count bit 15–0: phase check count (should be 8000)
0200	(r)	**jstat**	bit 31: (nofifo) no FIFO full to block the trigger bit 30: (autorst) auto reset mode (runreset when connection changes) bit 28: (genbor) generate begin-of-run trigger at run start bit 24: (usetlu) connect to TLU at AUX port bit 16: (regbusy) artificial busy by software bit 12: (clrictrl)reset IDELAYCTRL bit 9: (feereset)FEE reset (write only) bit 8: (b2lreset)Belle2link reset (write only) bit 4: (cntreset)FTSW counter reset (write only) bit 0: (runreset)run reset (write only)
0210	(r)	**utimrst**	time in second when counter or run reset
0220	(r)	**ctimrst**	bit 28: reset source was run reset bit 27: reset source was counter reset bit 26–0: (27-bit) time in clock when counter or run reset
0230	(r)	**utimerr**	time in second when an error occurred
0240	(r)	**ctimerr**	bit 26–0: (27-bit) time in clock when an error occurred
0250	(r)	**errport**	bit 11–0: (12-bit) port where an error occurred bit 31–16: (16-bit) number of occurred errors
0270	(r)	**tlu**	bit 27: (1-bit) busy signal status to TLU bit 25: (1-bit) trigger signal status from TLU bit 24: (1-bit) reset signal status from TLU bit 23–16: (8-bit) number of reset signals from TLU bit 14–0: (15-bit) trigger tag from TLU
0280	(r/w)	**trgset**	write anything: reset dummy trigger cycle bit 31–20: (12-bit) trgopt: optional parameter bit 17–8: (10-bit) rateval: dummy trigger rate (linear) bit 7–4: (4-bit) rateexp: dummy trigger rate (exponent) bit 2–0: (3-bit) seltrg: type of trigger source 0: none 1: IN 1-2 pair 2: AUX 7-8 pair 3: TLU trigger 4: pulse dummy trigger (uniform over revolution) 5: revo dummy trigger (fixed phase in revolution) 6: random dummy trigger (uniform random interval) 7: poisson dummy trigger (poisson interval) See trigft.c for more detail of dummy trigger setting
0290	(r/w)	**tlimit**	0: no trigger is generated 0xffffffff: no limit

			else: number of triggers to generate
02a0	(r)	**tincnt**	number of input trigger since cnt/runreset
02b0	(r)	**toutcnt**	number of output trigger since cnt/runreset
02c0	(r)	**tlast**	number of triggers yet to be generated
02d0	(r)	**stafifo**	bit 31: FIFO is full bit 30: FIFO overrun error bit 28: FIFO is empty bit 25–24: reading higher (0) or lower (1) 32-bit bit 0: trigger is enabled
02e0	(r)	**fifo**	four 32-bit words are sequentially read out (0xffffffff if empty) word 0 bit 31: (1-bit) 0 if fifo data is valid bit 30–4: (27-bit) trigger time in clock unit bit 3–0: (4-bit) trigger type word 1 bit 31–0: (32-bit) trigger time in second unit word 2 bit 31–0: (32-bit) trigger tag (start from 0) word 3 bit 14–0: (15-bit) TLU tag (start from 1)
0300	(r)	**enstat**	encoder status (for debug)
0340	(r/w)	**baddr**	broadcast / address for reset or other commands
0370	(r)	**lckfreq**	on-board clock frequency
0380	(r)	**destat**	FTSW's and connected systems' status bit 31: (fifoerr) FTSW's FIFO error bit 30: (pipebusy) FTSW's flow control pipeline busy bit 29: (sigbusy) logical-or of all external busy sources bit 28: (busy) logical-or of ALL busy sources bit 27–24: (xnwff) COPPER's length FIFO full bit 23–20: (xbusy) COPPER's slow busy bit 19–12: (obusy) FEE's slow busy bit 11–8: (xbusysig) COPPER's fast busy bit 7–0: (obusysig) FEE's fast busy Fast busy is based on single BUSY octet, slow is busy based on busy bit in the packet.
0390	(r)	**linkup**	b2tt link status bit 23–20: (xalive) Connected COPPER is alive bit 19–12: (oalive) Connected FEE is alive bit 11–8: (xlinkup) Connected COPPER's b2tt link is up bit 7–0: (olinkup) Connected FEE's b2tt link is up (alive: meaningful b2tt packet is received linkup: linkup bit is on in the b2tt packet)
03a0	(r)	**deerr**	decoder error status bit 31–24: (b2lup) Belle2link is up at FEE bit 23–16: (plllk) GTP PLL is locked at FEE bit 15: (staerr) Error somewhere bit 14: (errin) Error somewhere in COPPER or FEE bit 13: (clkerr) Error in clock bit 12: (tshort) Error in short trigger interval bit 11–8: (xerr) Error from COPPER bit 7–0: (oerr) Error from FEE
03b0	(r)	**linkerr**	bit 31–24: (b2lnodn) Belle2link was not lost at FEE bit 23–16: (pllnodn) GTP PLL was not lost at FEE bit 15: (linkerr) B2TT link error has occured bit 14: (b2lor) FEE Belle2link is ready bit 13–12: (staictrl) IDELAYCTRL status

			bit 11–8: (xlinkerr) COPPER link error bit 7–0: (olinkerr) B2TT link error
03c0 **03e0** **:** **04e0**	(r) (r) (:) (r)	**sta1a** **sta2a** **:** **sta8a**	status of **O1–O8** ports bit 31–12 addr bit 11: feeerr bit 10: tagerr or fifoerr bit 9: seuerr bit 8: busy bit 7–4: tagdiff bit 3–0: b2ldiff
03d0 **03f0** **:** **04f0**	(r) (r) (:) (r)	**sta1b** **sta2b** **:** **sta8b**	status of **O1–O8** ports bit 31–16: b2l write count bit 15–0: id, etc
0500 **05a0**	(r/w) (r)	**latency** **udead**	bit 31–24: maximum trigger allowed before latency (default: 12) bit 17–0: maximum allowed latency per event in clock (default: 0x8000) total deadtime in second since counter / run reset
05b0	(r)	**cdead**	total deadtime in clock since counter / run reset
05c0	(r)	**pdead**	deadtime due to pipeline (this and following dead time has bit 31:16 count in second and bit 15:0 count in 2^{11} clocks = 16 μs)
05c0	(r)	**pdead**	deadtime due to pipeline
05d0	(r)	**edead**	deadtime due to error
05e0	(r)	**fdead**	deadtime due to FTSW's FIFO full
05f0	(r)	**rdead**	deadtime due to software register
0600 **:** **0670**	(r) (:) (r)	**odead1** **:** **odead8**	deadtime due to FEE-1 busy deadtime due to FEE-8 busy
0680 **:** **06b0**	(r) (:) (r)	**xdead1** **:** **xdead4**	deadtime due to COPPER-1 busy deadtime due to COPPER-4 busy
06c0 **:** **06f0**	(r) (:) (r)	**xbcnt1** **:** **xbcnt4**	bit 31–16 slow-busy count bit 15–0 fast-busy count
0700 **:** **0770**	(r) (:) (r)	**odbg1** **:** **odbg8**	bit 31–30 packet error count bit 29 FTSW bit-slip status bit 28 valid octet bit 27–16 FEE idelay bit 15 invalid octet count MSB bit 14–12 invalid octet count 3-LSBs bit 11–0 FTSW idelay
0780 **:** **07b0**	(r) (:) (r)	**xdbg1** **:** **xdbg4**	ttrx debug info to be updated in next release
07c0	(r)	**odelay**	bit 31–24 O1–O8 manual idelay setting mode bit 23–16 O1–O8 manual slip setting bit 15–8 O1–O8 clear delay bit 7–0 O1–O8 increment delay
07d0	(r)	**odelay**	bit 19–16 X1–X4 debug selector bit 15–12 X1–X8 manual idelay setting mode bit 11–8 X1–X8 manual slip setting bit 7–4 X1–X8 clear delay bit 3–0 X1–X8 increment delay
07e0	(r)	**dbg**	more debug registers

07f0	(r)	**dbg2**	
0800 **:** **0890**	(r) (:) (r)	**dump** **:** **:**	snapshot of out-going b2tt packet
08a0	(r)	**acksig**	bit 15–12: (xack) Raw ACK signal from COPPER bit 7–0: (oack) Raw ACK signal from FEE
0900 **:** **0990**	(r) (:) (r)	**dump2** **:** **:**	snapshot of out-going b2tt stream

7 TT-RX

Connection between FTSW and TT-RX should be made between FTSW's **O1–4** port and TT-RX's **TTD** port. This connection should made with a shielded CAT7 cable, or if shorter than 5 m, shielded CAT5e/6 cable.

Optionally TT-RX can generate dummy triggers, which can be delivered to FTSW. To do this, another connection from TT-RX's **EXT** port to FTSW's **AUX** port is needed. This connection can be a shielded or unshielded CAT 5/6/7 cable.

The **tt4r** firmware has to be downloaded with the **bootrx** command. Then there is basically nothing to control. There are some TT-RX registers that can be monitored with the **regrx** command, and a slightly better summary is produced by a **stat4r** command.

Trigger is generated by a **trigrx -localbusy -fifo=no** command. The localbusy option is needed as HSLB is not designed to generate event-by-event trigger-busy handshake. The fifo=no option is not needed at a low trigger rate, but it will probably needed at a high rate (especially when the COPPER CPU is the slow Radisys EPC-6315), as it slows down the trigger generation by reading the TT-RX FIFO. Two trigger types, **pulse** and **random**, are provided. Just run **trigrx** with no argument to show the usage. For example, 1000000 events at about 1000 Hz with fixed interval can be generated by

```
trigrx -localbusy pulse 1000 1000000
```

Programs other than **stat4r** are included in the **ttrxprogs** package.

8 Missing Items

There are still missing items in version 0.38 of **ft2u** firmware. some of which are already described above. Here is the most likely incomplete list of items to be implemented at some point.

reset from TT-RX — even in the standalone mode there is a control line from the COPPER CPU through TT-RX to FTSW. Currently TT-RX can be used for generating dummy triggers, but it would be useful if more functions are implemented, such as generating run reset.

Point-to-point control — to one of the connected frontend systems. Mechanism is partially implemented, but a serious test is yet to be made and tools are yet to be prepared.

JTAG programming without Xilinx impact — it was previously studied to download a bit file to FPGA without using Xilinx products, but this function is not implemented. It was tested for two cases: using the XSVF player and by a fully custom-made program.

many more — some of them are already known and just not listed here, some of them are yet to be identified.