Receiver Nodes of the General Machine Timing System for FAIR and GSI

Mathias Kreider, Jioani Bai, Dietrich Beck, Cesar Prados, Wesley Terpstra, Stefan Rauch, Marcus Zweig

Abstract

This document summarizes the "Timing Receivers", which are part of the component “General Machine Timing System” of the accelerator control system.

Table of Contents

List of Tables

List of Figures

1. Purpose and Classification of the Document

The purpose of this document is to describe the work package “Timing Receivers”, which are used by the accelerator control system component “General Machine Timing System”.

1.1. Classifications of Requirements

The following definitions of requirement classifications are being used throughout the document:

- “**Must**” or “**shall**” or “**is required to**” are used to indicate mandatory requirements, strictly to be followed in order to conform to the standard and from which no deviation is permitted.
- “**Must not**” or “**shall not**” mean that the definition is an absolute prohibition of the specification.
- “**Should**” or “**is recommended**” are used to indicate that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others or that a certain course of action is preferred but not required.
- “**Should not**” or “**is not recommended**” mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighted before implementing any behavior described with this label.
- “**May**”, which is equivalent to “**is permitted**”, is used to indicate a course of action permissible within the limits of the standard.

2. Scope of the Technical System

2.1. System Overview

The FAIR General Machine Timing (GMT) system is a component of the accelerator control system and has the responsibility of synchronizing accelerator actions throughout the whole accelerator facility [3]. The GMT utilizes the White Rabbit PTP (WR-PTP) protocol [4], [10], [16].

The GMT will be based on a network with a tree topology with a timing master on top, consisting of switches and Timing Receivers. All participants synchronize their internal oscillators and local time to the clock master node using White Rabbit (WR).

This document focuses on these Timing Receivers, which are called FAIR Timing Receiver Nodes (FTRN). FTRNs receive and decode broadcast network messages in real time. These are sent by a data master, which is a component of the timing master, located on top of the WR network topology using the EtherBone [14] protocol.

Each FTRN is configured by third party software, typically Front-end System Architecture (FESA) [6], to execute a set of actions known in advance. These actions are scheduled for execution by the data master, which broadcasts commands containing absolute timestamps (such as Temps Atomique International – TAI) denoting the time of action execution and references to an action. When a broadcast command is received, FTRNs enqueue the designated action to be executed locally at the specified time.

Furthermore, all FTRNs shall be able to send and receive over the WR network low priority network messages for management, diagnostics and maintenance. Support for sending high priority network messages shall also be present, e.g. for beam dump notification.

Additional bus interfaces and local controls are also foreseen in many form factors.

FTRNs have the following features:

- clock and time synchronization with a clock master according to the WR specifications
- ability to record arrival time of a digital pulse (e.g. LVTTL)
- ability to generate a digital pulse or a pulse sequence at a given time
- ability to generate clock signals with a given frequency and phase
- ability to generate specified interrupts at given times

Figure 1 depicts the system architecture of a FTRN and its hosting platform.

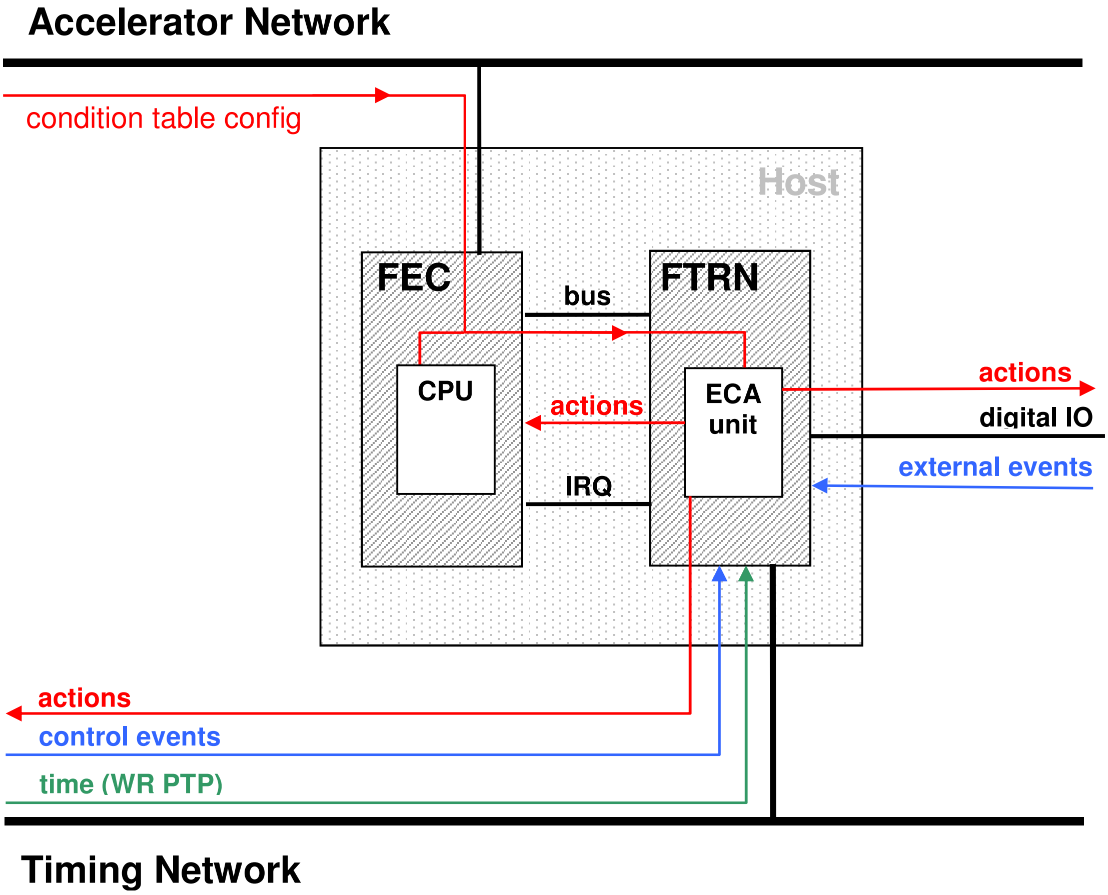


Figure 1: Hosted FTRN system architecture

The FTRN and a **front-end controller** (FEC) are embedded in a host. The FEC is connected to the ACCelerator NETwork (ACCNET) while the FTRN is connected to the dedicated timing network. FEC and FTRN are interconnected via the host system bus and interrupt lines. FTRN time synchronization is achieved by the WR-PTP protocol over the timing network. Coded instructions for FTRNs are typically generated by the data master (not shown) and are called **commands**. Commands come in using EtherBone as messages broadcasted over the timing network or via FEC request. Execution of actions follows the **Event-Condition-Action (ECA)** model [7]: **Conditions** define how FTRNs map the event[1] of an incoming **command** to an **action**. **Actions** drive accelerator equipment by generating digital output, FEC bus interrupt or timing network messages. FTRN actions are typically configured via the FEC CPU. The **ECA unit** executes actions on time. The time of execution is specified by a timestamp that is part of the command received from the data master.

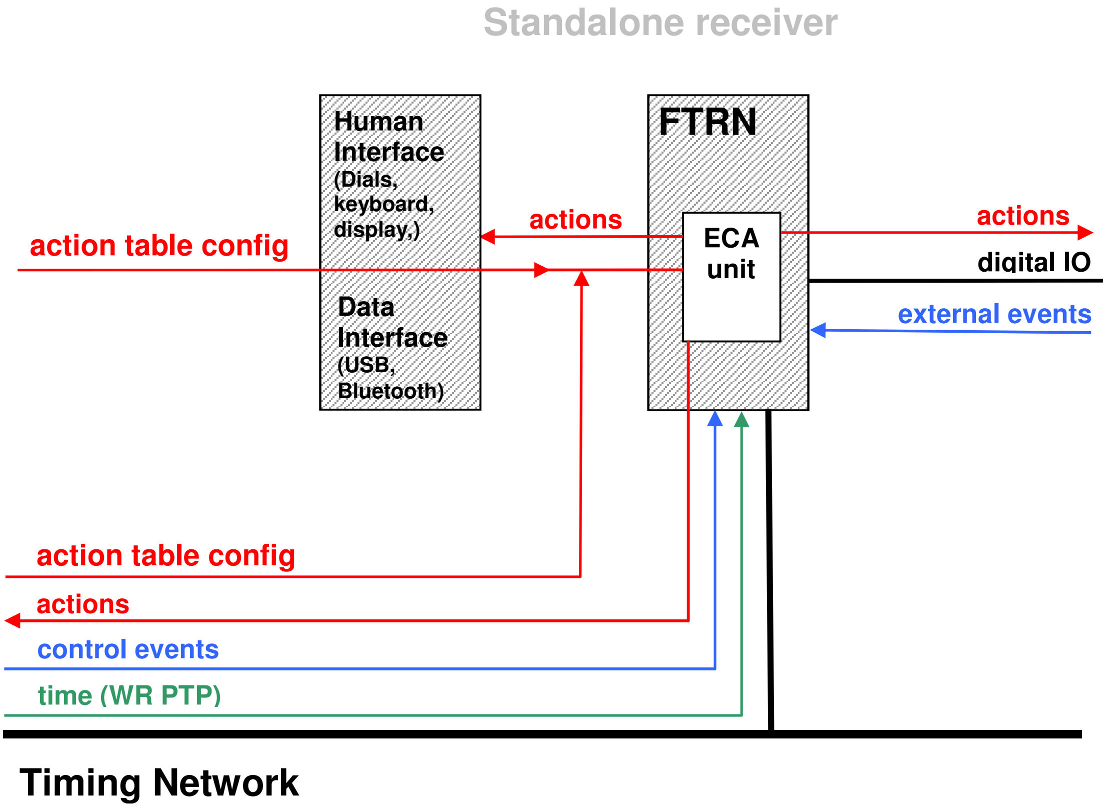


Figure 2: System architecture of a standalone FTRN.

FTRNs shall be available in different form factors for operation within different types of crates or PCs and as standalone devices. The latter are shown in Figure 2. Here, the FEC is replaced by a human interface (knobs, display,…). A standalone FTRN is not embedded in a host system.

[1] In the document the word „event“ is used in the original sense: *„event“ = „something happens“*

2.2. Limits of the System and Environment

This section describes what is not in the scope of the work-package and the constraints imposed by the environment.

2.2.1. Compatibility to PCIe FTRN PEXARIA5

The existing FTRN PEXARIA5 [24] and its mezzanine board PEXARIA5DB [32] is the reference implementation for all FTRNs covered by this document, see section 3.2.

2.2.2. Interfaces

Everything on the far side of the main FTRN interfaces is out-of-scope. The four main interfaces are

- the WR network layer
- the bus interface
- the host independent mezzanine board, if supported by the carrier
- the host dependent I/O connectors (e.g. LEMO Series 00)

2.2.3. Limits

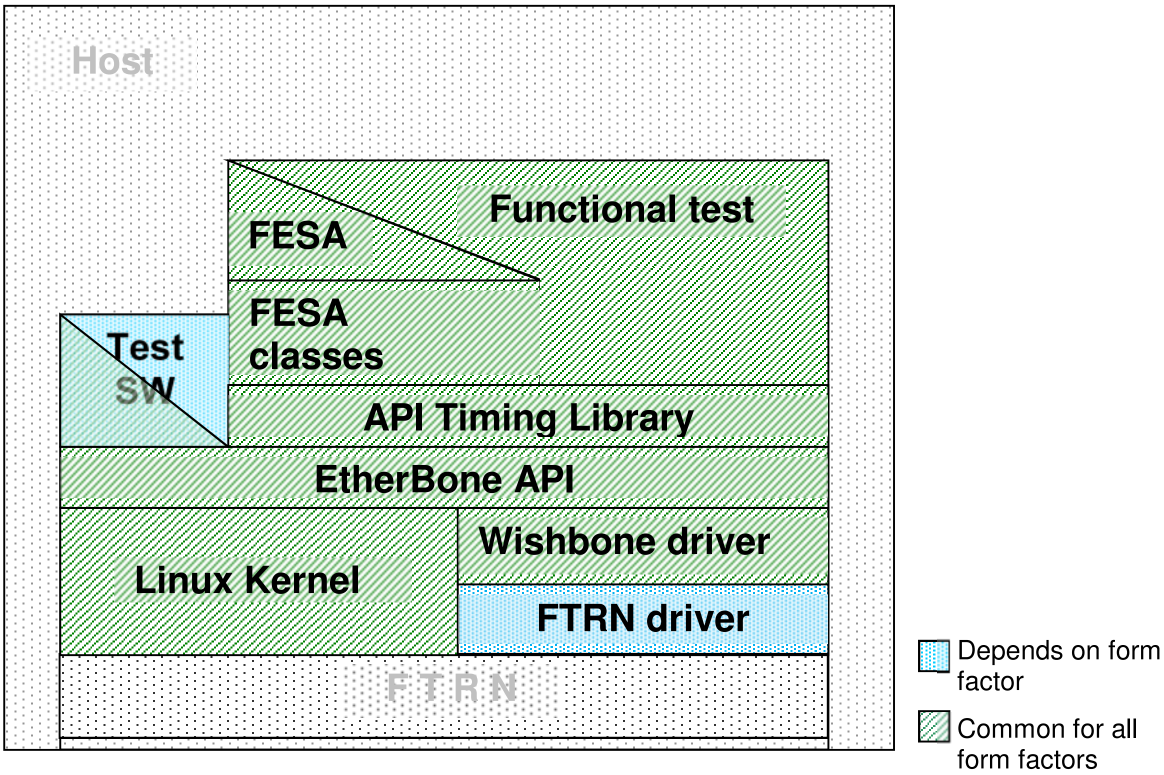


Figure 3: Software Boundaries on the FTRN host system.

Software

Figure 3 shows the software stack of the FEC. The FTRN Linux device driver uses resources of the Linux Kernel (driver subsystem) to communicate with the FTRN over the host bus (which is form factor specific). It provides low level access to the FTRN including the Wishbone bus [21] and interrupt handling. Furthermore, it has an interface to a form factor independent Wishbone driver.

The Wishbone driver provides a userland software Wishbone (WB) interface including a 32-bit memory map, cycle error and control lines.

Abstraction to the communication layer is realized by the EtherBone API, which provides register I/O to Wishbone devices and serves as **the** interface towards userland applications. It provides abstract methods to access components of the FTRN WB bus.

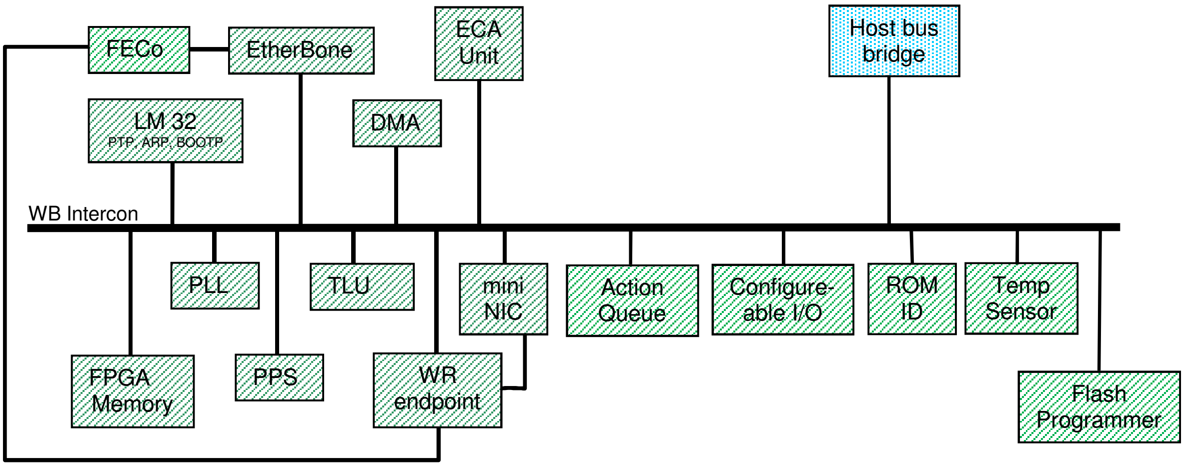
An API Timing Library (ATL) implements an interface towards third party user software such as a FESA class. The ATL abstracts the FTRN by hiding its Wishbone architecture and address layout. It provides high level functionality of the FTRN. Moreover, the ATL only provides the functionalities of the FTRN which is required by the concept of the control system. This implies that other functionalities already implemented by the FTRN may intentionally be hidden.

The Test SoftWare (TSW) shall be developed and implemented for testing gateware and hardware of a FTRN. The TSW uses the EtherBone API as an interface to the FTRN. The GSI Timing Team has already implemented the EtherBone API, Wishbone driver and FTRN drivers for PCIe, VME and USB interfaces. Moreover, **all** FTRN form factors can be accessed through the EtherBone API via the White Rabbit network and USB.

Of the software stack of the FEC, only the form factor Dependent Test SoftWare TSWD and the FTRN driver depend on the form factor. All software must be developed for Scientific Linux 6 or later and support 32bit and 64bit. All software must be compatible to real-time patches.

HDL

Boundaries on FTRN Hardware Description Language (HDL) are illustrated in Figure 4. It remains to implement host bus to WB master and slave bus interfaces and to connect it to one of the WB crossbars.



░ To be done for some form factors

▨ GSI Timing Team

Figure 4: Boundaries on the FTRN HDL. Only the host bus for some form factors remains to be done. (this figure shows only most relevant modules)

All WR switches are out-of-scope; however they are needed as fundamental infrastructure for the implementation of nodes. When a FTRN is connected to a switch, it shall establish a link according to the WR protocol. Furthermore, it shall integrate into the General Machine Timing System seamlessly, as described later in this document. Thus, FTRNs shall be developed in close coordination with the GSI Timing Team. Hardware and software developments shall be implemented in accordance with the CERN Open Hardware licensing scheme [5] and GPL [9] respectively.

2.2.4. Environment

The environment of the FTRNs beyond the four main interfaces consists of

- the WR timing network (~3000 FTRNs, switches …)
- the host environment: crates, PCs or other devices in which FTRNs are integrated, e.g. receivers in VME form factor operate within VME crates
- accelerator equipment connected to FTRN I/O connectors

The network environment is based on Gigabit Ethernet and has some special features such as

- deterministic delivery of control messages,
- reliable delivery, with packet loss rate less than 10^{-12},
- geographical extensions of 2 km and
- approximately five layers of WR switches.

The host environment of the FTRN depends on the form factor. Limits on

- power supply
- cooling performance
- mechanical hosting
- bus communication

are imposed by the form factor standards (e.g. VME, MTCA …).

The FTRN I/O connectors can send or receive digital signals. I/O standards are defined in section 3.2.

FTRNs will operate in the environmental conditions specified for their host systems. Exposure to significant radiation levels is not considered. Environmental endurance for standalone FTRNs is yet to be specified.

Section 2 assumes that FTRNs are connected to WR switches using optical links.

2.3. Basis of Concept

The FAIR timing system is specified in the FAIR Detailed Specification “General Machine Timing System” [3].

Relevant to Timing Receivers are the following technical specification documents:

- WR specifications [10]: describe the WR network, in particular the clock synchronization protocol, known as WR-PTP [4], [16].
- WR node functional specifications [11]: describe the Timing Receiver functionality for a general purpose node.
- FAIR Common Specifications "Accelerator Control System" [2].

2.3.1. Functional Requirements

The "Timing Receivers" fulfil the following functions requirements.

Number	Description of the Requirement
TR_000	**Scheduled digital signal generation** – Digital signals (e.g. TTL, LVDS) must be available on connectors. FTRNs shall act as programmable wave form generators with a WB slave interface. When requested by the ECA unit, they shall produce a pulse or a pulse sequence as well as clock signals of variable phase and frequency. It must be possible to run slow serial protocols (I2C, SPI, UART, MIL).
TR_009	**Actions** – FTRNs shall support at a minimum the following actions: host system messaging (TR_010-TR_013), dispatching network messages (TR_180), and waveform generation (TR_130, TR_000).
TR_010	**Action queue** – The FTRN shall maintain a time stamped buffer for messages from the ECA unit to the FEC. The queue shall be readable from the FEC, which then marks entries as processed.
TR_011	**Overflow detection** – The FTRN device driver must detect action queue overflow and report this to the API timing library.
TR_012	**Interrupt generation** – FTRNs shall generate interrupt signals to the FEC of the host system, when the action queue is not empty. Interrupt masking shall be supported by a register in the FTRN. This enables performing actions on the host system synchronized with accelerator operations.
TR_013	**Polling mode** – The ATL must implement a "polling mode", if the host bus does not support interrupts.
TR_020	**ECA unit** – The Event-Condition-Action (ECA) unit processes incoming commands and initiates appropriate actions. Conditions and actions in the ECA are programmed over the WB bus / FEC by FESA. They specify which commands are transformed to actions on this FTRN, taking into account local information.
TR_030	**Action Interface** – The ECA unit outputs actions conforming to the action interface format. Receiving components accept actions in this format.
TR_040	**Host Bus Bridge** – A host bus to Wishbone bridge shall be implemented. This allows the FEC to configure WB slaves in the FTRN.
TR_050	**USB-Interface** – All FTRNs must have a USB slave interface. As a WB master, it configures WB slaves in the FTRN.
TR_060	**Manual controls** – In case the FTRN is not connected to a host-system, a Human Interface Device (HID) like manual knobs or displays should allow configuring WB slaves in the FTRN. The assignment of HID components to configuration parameters must be programmable.

TR_070	**Logging** – FTRNs shall implement an interface that allows to record their command and action history for post mortem analysis and other purposes.
TR_080	**Sharing** – The ATL must support simultaneous access from multiple userland applications on the FEC.
TR_100	**External events** – A FTRN should be able to handle events from an external interface like a digital input. As an example, this allows triggering actions by beam monitors. As part of the gateware, a configurable component should be developed that generates Wishbone "writes" upon a signal received over a digital input line.
TR_110	**Delays** – It shall be possible to delay all input and output signals.
TR_120	**I/O selection** – All bi-directional connectors shall be flexibly configured to be either an input or an output.
TR_130	**Configuration block of waveform generators** – The waveform generator produces signals on the output connectors. A waveform is configured with an FPGA-like composition of AND/OR/MUX/… components linking action inputs (from the ECA unit) to digital outputs. Examples of waveforms include "Single Gate", ”Single Burst", "Toggling Gate", "Toggling Burst" and "1/N Bursts of N Pulses" as described in [28]. Whenever the configurable components are connected in series, there is a register between them.
TR_140	**Status register** – In order for the FEC to verify correct operation, FTRNs shall have an accessible status register and shall issue an interrupt if particularly critical conditions occur (e.g. timing synchronization was lost). The ATL must inform the front-end software if a Timing Receiver ceases to work correctly or the communication is disrupted.
TR_150	**LEDs** – Various LEDs shall be provided to allow easy installation, operation start up and diagnostics. In particular, it is necessary to have general error LEDs and WR network LEDs showing status of the link, of the clock synchronization and of the PLL. Additional LEDs or a tiny front display should provide additional information. See section 3.1.
TR_160	**Reset** – FTRNs shall be resettable via EtherBone. A reset button on the front panel is not required.
TR_170	**Gateware and firmware update** – It shall be possible to update the gateware and firmware via EtherBone (WR network, host bus bridge, USB,...) or directly (e.g. via JTAG).
TR_180	**Dispatching network messages** – A FTRN shall be able to send messages of high priority via the timing network. This is triggered by an action.
TR_190	**Timestamp latching** – A FTRN shall be capable of latching a timestamp, if it is configured accordingly. Latching a timestamp can be triggered by an electrical signal (LVTTL, LVDS…) at one of the input connectors.
TR_195	**Timestamp availability** – It shall be possible to query a timestamp containing the actual time via EtherBone.
TR_200	**BuTiS [19] clock generation** a) A FTRN should be able to synthesize a **w0** signal that has the same clock and phase as the BuTiS **T0** clock [19]. For this, it receives tuning words from the data master (connected to BuTiS). The accuracy of the generated w0 clock shall be sufficient for the identification of **c2** clock cycles, see TS_430 and TS_410 [3].

	b) A FTRN should be able to synthesize a **w2** clock signal related to the BuTiS **c2** clock. That clock signal should be frequency and phase matched to the **c2** clock generated by a BuTiS receiver. Info: The WR clock of the GMT is phase locked to the BuTiS clock.
TR_210	**Timestamp distribution** – It shall be possible to provide timestamps via a serial protocol on an output connector (LVTTL, LVDS…). As an example, this could be encoded between the generated w0 pulses ("At the next tone, it will be …").
TR_220	**Form factor dependent I/O** – Some form factors shall implement I/O connectors on the carrier board. Those will be specified together with the GSI Timing Team before the Printed Circuit Board (PCB) layout.
TR_230	**Identification of hardware, gateware and firmware** – Every FTRN and mezzanine shall provide a unique serial number, model and revision. Every gateware and firmware (bitstream, soft-CPU code…) shall provide information including type, version number, author, fitness for the particular hardware and possible unauthorized changed. The ATL provides means of verifying the compatibility between the components.
TR_240	**Persistence** – All FTRN configurations shall be saveable locally. This is required to fulfill requirement TS_190 of the GMT [3].
TR_250	**WR** – All FTRNs shall be compliant to the WR specifications [10].
TR_260	**EtherBone** – All FTRNs shall be compliant to EtherBone [14].
TR_270	**Forward Error Correction (FECo)** – All FTRNs shall support FECo for data received from and send to the timing network.
TR_280	**IP Addresses** – All FTRNs obtain their IP addresses via BOOTP.
TR_290	**Verification of gateware and firmware** – It should be possible to detect run-time modification of firmware and gateware. Such a run-time check shall be investigated, see section 3.7.2.

Table 1: List of FTRN and supporting HDL/software functional requirements.

2.3.2. Non-functional Requirements

The "Timing Receivers" fulfil the following non-functions requirements.

Number	Description of the Requirement
TR_310	**Maintenance** a) All form factors embedded in a host system shall support loading of gateware, firmware and configuration by the host system. b) All form factors shall support loading their gateware and firmware from flash and shall support gateware and firmware update via EtherBone.
TR_320	**Safety** – FTRNs shall be able to send safety related information to the host bus, to the WR network as high priority messages and on the I/O connectors as digital signals. This is of interest for special cases like transmitting beam dump notifications to the post mortem system.
TR_325	**I/O GMT compliance** – The signal properties at I/O connectors of FTRNs shall be compliant to fulfil the requirements of the GMT [3]. The PCB layout and electronic circuit design connecting I/O connectors and FPGA pins shall consider the rise-time at outputs, the edge-detection at

	inputs and the intrinsic delays. Specifically, the following requirements of the GMT shall be supported. a) The jitter should be 100 ps or less. Jitter shall be 500 ps or less (TS_390). b) The precision shall be 1 ns or better (TS_400). c) The accuracy shall be sufficient for unambiguous identification of BuTiS c2 clock cycles (TS_410). d) BuTiS clock signals and their properties (TR_200 and TS_430). The terms jitter, precision and accuracy are defined within the specifications of the GMT [3]. The intrinsic delays for I/O signals between the connectors and the FPGA pins as well as the delays due to signal processing in the FPGA shall be determined and made known to the GSI Timing Team.
TR_330	**Real time reliability** – FTRNs shall have prior knowledge of the intrinsic delays for actions connected to relevant commands, in order to guarantee timely execution or to signal a missed deadline. Sources of intrinsic delays include HDL execution, signal propagation in PCBs and electronic circuits connecting FPGA pins to I/O connectors.
TR_335	**Local delay compensation** – FTRNs shall be able to delay or pre-trigger the execution of actions within limits defined by the GMT. Local delays are typically requested by third party software such as a FESA class to compensate for timing issues outside the scope of the main interfaces, see section 2.2.2.
TR_340	**Clock synchronization** – Each FTRN shall provide a PPS output in order to verify the clock synchronization between WR devices. A synchronization of 1 ns or better shall be achieved.
TR_350	**Scalability** – It shall be possible to operate many FTRNs on the same host bus
TR_360	**Debugging** – It shall be possible to use signal analyzers to debug HDL and debuggers on soft CPU processes.
TR_370	**Monitoring** – It shall be possible to monitor FTRNs on the WR network by means of a management interface.
TR_380	**QR-code** – Each FTRN shall have a QR sticker, recognized by notebook / smartphone for configuration or diagnosis.
TR_390	**Design concept** – Functionality should be implemented in the FPGA, not via dedicated electronics outside the FPGA.
TR_400	**Hardware follows gateware** – The design of the hardware must suffice gateware already developed by the GSI Timing Team, see section **Error! Reference source not found.**.

Table 2: List of Timing Receiver Non-functional Requirements

2.3.3. General Constraints

The following constraints apply.

- All development and production tools must be compatible with GSI infrastructure.
- VHDL shall be the preferred HDL programming language

- Only widely supported tools shall be used
- All project material shall be kept under version control

2.3.4. Architectural Principles

The Software Architecture Guideline for the Control System [20] fully applies.

3. Technical Specifications

The FTRNs shall be based on and compliant to WR technology.

3.1. Common Features

3.1.1. Hardware

All FTRNs shall feature a unique FTRN carrier board serial number, which shall be obtained via a one-wire temperature sensor.

All FTRNs have a unique vendor specific MAC address. Private MAC addresses must not be used. On each FTRN, the MAC address is stored in an EEPROM on the carrier board. The type of EEPROM is specified by the GSI Timing Team and the supplier will deliver FTRNs with MAC addresses and make the MAC address for each device known to the GSI Timing Team.

A “Mean Time Between Failure” (MTBF) calculation for each FTRN type shall be performed.

The mechanical design shall allow quick replacement of FTRN carrier boards or mezzanine boards by a technician using standard tools (e.g. screwdriver) without the need for special tools or soldering.

Other mandatory common features are listed in section 3.2.1.

The hardware design of the different FTRN form factors should avoid redundant developments; all circuits and components should be as similar as possible.

3.1.2. Start up, Gateware, Firmware and Configuration

FTRNs need two kinds of start-up data: gateware and firmware (mandatory) and configuration (optional). The gateware and firmware is intended to be delivered and modified only by developers, while the configuration consists of parameters saved by the user.

All form factors support loading their gateware and firmware from flash and support gateware and firmware update via EtherBone.

All form factors embedded in a host system shall support loading of gateware, firmware and configuration by the host system.

All form factors shall support saving/loading configuration data on/from FTRN EEPROM.

FTRNs shall provide two different regions in the flash memory. One stores the gateware and firmware loaded on normal power up. The other stores a factory default gateware and firmware which is loaded either when the normal power-up gateware is broken or when one of the buttons on the PCB is pressed. Both types of gateware and firmware shall include the following:

- Wishbone firmware programmer
- Programmer for on-board gateware and firmware flash
- EtherBone networking
- Host bus bridging support

In case a FTRN features a host bus bridge chip, a hardwired connection to the FPGA programmer is required (see also section 3.6.4). WR calibration data and MAC address are stored in an on-board EEPROM on the carrier board.

3.1.3. Core Modules

FTRNs in all form factors share a set of core modules provided by the WR community and by the GSI Timing Team. An overview of the provided core modules can be found in Figure 4.

3.1.4. Configuring Actions

Configuration of actions is mainly performed over the bus, accessible from FEC, human interface, timing network.

Configuration of actions shall always be based on EtherBone.

All configurable gateware components relevant for implementation of actions (e.g. pulse generators) shall be Wishbone devices.

It shall be possible to schedule action executions simultaneously by means of commands with the same execution timestamp, provided the actions are handled by distinct ECA channels or timing receivers.

It shall be possible to disable/enable the ECA unit.

3.1.5. Waveform Generators

All FTRNs shall have waveform generators. Each waveform generator shall provide the features described in Table 1.

3.1.6. Timestamp Latch Unit

An external signal from an input port shall trigger the Timestamp Latch Unit (TLU) to store a timestamp of the current time. The TLU has multiple channels. Timestamps are stored in a FIFOs and can be retrieved via EtherBone.

The TLU can also be used as a configurable Message Signalled Interrupt (MSI) source. By this, it can be used to trigger another component upon an incoming digital signal as shown in Figure 5.

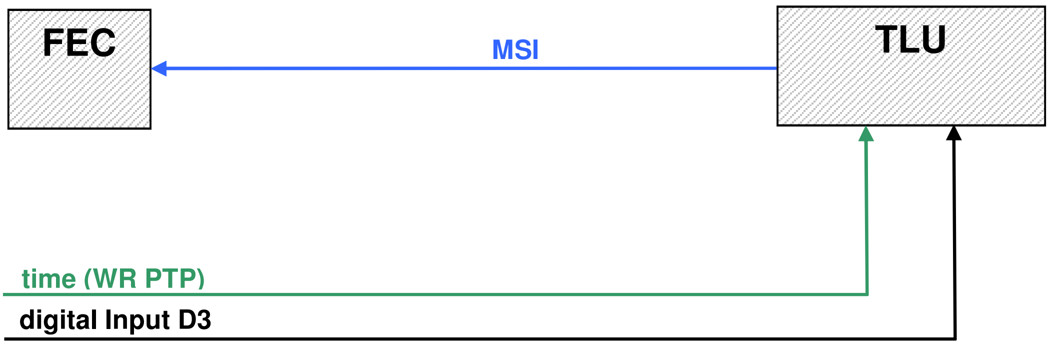


Figure 5: Example: A FEC receives a MSI from the TLU.

3.1.7. Signal Generation

I/Os are connected to the FTRN's FPGA and only digital signals are used. Dedicated circuits for each I/O serve to configure the direction and provide a buffer to protect the FPGA. In the following, some background information is given.

System Clocked

Here, the logic is implemented in the same clock domain as the 125 MHz system clock. This is the most general and flexible case. If a appropriate VHDL component is implemented, direction, source and sink of signals can be configured on-the-fly. Sources and sinks could include items such as I/O connectors, trigger input to the TLU, output of ECA action channels or input and output of simple logic operations (AND, OR, NOT). Due to the 125 MHz clock, signals can be generated and processed within 8 ns. As an example the length of gate signals is a multiple of 8 ns and the period of generated clock signals have a multiple of 16 ns.

Driven by Differential I/O Buffers

System clocked signals can possibly be individually phase shifted by using the output buffers of differential signals on the FPGA, which can be clocked to 1.25 GHz. Using this technique, the implementation of fine delay with 1ns resolution is straight forward for input as well as for output signals. For output, the fine delay value can be changed on-the-fly within one 125 MHz clock cycle for all outputs of the concerned ECA channel. For input, signals of different connectors are sampled individually.

Driven by WR PLL

If higher granularity or phase shifting is required, the PLL of the WR clock can be used. This is as special case and imposes limitations on the number of signals that can be generated and it is not guaranteed that this resource is available at all. This method allows generating clock signals with a period of 1 ns multiples and a defined phase relative to the 125 MHz system clock. Besides restrictions in the PLL itself, the generated signals must be connected in a clock domain different from the 125 MHz system clock, which prevents relevant I/O to be configurable on-the-fly. As an example, this method can be applied for generating the BuTiS w2 clock. Another example would be the implementation of a fine-delay option for output signals with a granularity of 125 ps. As a drawback, a set-up time of up to 10 µs has to be considered when changing the phase. Furthermore, connections between PLL and output connectors are defined by VHDL and cannot be changed after the synthesis. This prevents using concerned connectors for other purposes.

Driven by Delay Lines with Differential Output Buffers

Combining the existing 1ns delay at the differential output buffers with delay chains might possibly allow fine delay with granularity of down to 25 ps. This needs to be investigated, but it is most likely only available on the FPGAs of the ArriaV GX family.

Driven by a Dedicated PLL

For more freedom in the frequency of generated clock signals, a dedicated PLL can be used. However, the generation of clock signals from the system clock must suffice certain ratios and the specified frequency range of involved oscillators. Like for the WR PLL, this imposes restrictions on the configurability of I/Os. The availability of this resource is even more restricted and it depends on the schematics of a FTRN, to which I/O connector the signal can be connected. It must be stated that the implementation of such signals is clearly outside the specifications of GMT and FTRNs. It is mentioned here only for completeness.

3.1.8. Monitoring

FTRNs should implement a management interface (e.g. SNMP).

3.1.9. Interface to the Host

On some kinds of host systems, the FEC needs a bus controller in order to communicate with FTRNs. Normally this bus controller is integrated in the CPU module, which is also the bus master.

The engineering specification shall clarify this issue, especially for VME, MTCA and PMC, because FTRN drivers depend on the bus controller. However, the FTRN should depend on as few of the peculiarities of a given bus controller as possible, adhering strictly to the relevant standards and thus maximizing compatibility.

3.1.10. Drivers and Test Software

The “Timing Receivers” require the following drivers and test software.

Number	Description of the Requirement
TR_1000	**VME** – A FTRN driver for a VME host bus (see [25]) has been developed by the GSI Timing Team. Status (February 2014): The driver has been derived from the CERN VME64x driver; release candidate.
TR_1010	**USB –** A FTRN driver for the USB bus common to all form factors shall

	be developed. Status (February 2014): done and released
TR_1020	**PCI/PCIe –** A FTRN driver for the PCI/PCIe bus common to many form factors (see [23], [24] and section 3.2.3) shall be developed. Status (February 2014): done and released
TR_1030	**VME64x** – A FTRN driver for the specified VME64x form factor (see section 3.2.2) is required. It shall be confirmed, if the existing VME driver (TR_1000) can be used. If not, the FTRN driver for VME64x shall be derived from the VME driver TR_1000.
TR_1040	**PMC** – A FTRN driver for the specified PMC form factor (see section 3.2.4) is required. It shall be confirmed, if the existing PCIe driver (TR_1020) can be used. If not, the FTRN driver for PMC shall be derived from the PCIe driver TR_1020.
TR_1050	**MTCA –** A FTRN driver for the specified MTCA form factor (see section3.2.3) is required. • It shall be confirmed, if the existing PCIe driver (TR_1020) can be used. If not, the FTRN driver for MTCA shall be derived from the PCIe driver TR_1020. • Crate management (IPM) is considered to be part of the MTCA standard. It shall be confirmed, that all requirements by MTCA.0 are fulfilled. If not, they have to be derived or developed.
TR_1060	**Performance of host bus to Wishbone bridges –** Performance of bridges is tested using the form factor independent Test SoftWare (TSWI). Tests always include the whole stack a) Userland test program b) EtherBone API c) Wishbone driver d) FTRN driver e) Kernel f) Bus access g) Host-bus to Wisbhone bridge on the FTRN h) Wishbone device on the FTRN Tests are done for issues including reliability, addressable range and latency. The following numbers serve as a guideline. • Robustness. An error-rate better than 10^{-12} must be achieved, see [3], TS_070. • Latency. Examples are given in section 3.6.4. • Address Range. The full 32 bit address range of the FTRN's Wishbone bus must be accessible without re-configuring the host bus bridge.
TR_1070	**Test SoftWare (TSW)** – Test Software for testing the FTRN's hardware shall be developed and implemented on top of the EtherBone API. It shall be possible to perform tests using the standard FTRN gateware and firmware.
TR_1080	**Environment** – All software must support 32bit and 64bit operating systems. Scientific Linux 6 or later with real-time patches must be supported.

Table 3: FTRN drivers (status February 2014).

3.2. FTRN Form Factors

The typical front-end controller at FAIR will be the Scalable Control Unit (SCU) with integrated FTRN functionality as described in [23]. It is expected that about 1000-1500 SCUs will be installed. Moreover, FTRNs of the three form factors VME [25], PCIe [24] and standalone [15] have already been developed by the GSI Timing Team of which about 500 will be employed by different users at FAIR. Hence, the FTRNs delivered by the control system component "Timing Receivers" are only a minor fraction of the total number of FTRNs on the FAIR campus. The FTRNs delivered by the work package "Timing Receivers" shall be compatible to the FTRNs already developed.

The existing FTRN PEXARIA5 [24] and its mezzanine board PEXARIA5DB [32] is the reference implementation from which all FTRNs shall be derived. The design of the other FTRNs should be compatible to the designs of the GSI Timing Team in many aspects including parts of the schematics as well as electronic components. As an example, all FTRNs delivered will be based on the Arria V GX FPGA family from Altera.

Two other detailed specifications, "Motion Control FECs" [26] and "Serial Equipment Control FECs" [27] mention timing receiver interfaces for those two types of FECs. If an additional Timing Receiver form factor for those FECs would be required, the development and production of this form factor must be agreed upon in written.

Table 4 gives an overview of the form factors, their FMCs and their digital I/Os.

Number	Description of the Requirement
TR_1200	**PCIe –** General purpose I/O shall be implemented on a mezzanine board. About 5 bi-directional I/O signals shall be available.
TR_1210	**Standalone (Handheld) –** At least 6 general purpose I/O shall be implemented on a mezzanine board. In addition a display is required. It shall be investigated, if a user interface via dials/switches is appropriate.
TR_1220	**Standalone (Rackmount) –** At least 6 general purpose I/O, display, dials, knobs,... shall be implemented on the front panel.
TR_1230	**VME64x** – This form factor shall be equipped with a mezzanine board. It shall feature the following connection on the front panel: • Carrier: 2 bi-directional LEMO 00 connectors for LVTTL signals and one JTAG connector in micro-USB format. One of these connectors shall satisfy TR_1260. • Mezzanine: 10 bi-directional LEMO 00 connectors for LVTTL signals. In addition 2 uni-directional LVDS output and 2 uni-directional LVDS input signals shall be provided via a 5x2 IDC connector. • A small display on the front panel. • A USB interface in micro-USB format (should be implemented on the carrier board).
TR_1240	**PMC** – This form factor itself is a mezzanine board and I/O should be implemented on the "main" PCB. 4-5 bi-directional LEMO 00 for LVTTL signals shall be implemented on the front panel.
TR_1250	**MTCA –** As for PMC, the I/O should be implemented on the "main" PCB. The following is required. • 5-6 bi-directional LEMO 00 for LVTTL signals on the front panel.
TR_1260	**Clock Input** – One single-ended I/O on the front panel shall be

	connected to a clock input of the FPGA. If the I/O is used as output, the connection shall be disabled.

Table 4: Form factors and their I/Os.

3.2.1. Common Features for all Form Factors

The design of the FTRNs for the different form factors should avoid diversity as much as possible. The aim is to keep hardware, firmware and gateware as portable as possible between the different form factors. This should be achieved by choosing identical electric components and similar circuit diagrams wherever possible. Compatibility to PEXARIA5 and other form factors by the GSI Timing Team is a severe issue for all form factors and the design of the form factors shall be oriented towards the GSI Timing Team designs, maximizing reusability as much as possible.
All FTRNs shall have the following features independent of their form factor.

Number	Description of the Requirement
TR_2000	**Reference design –** All FTRNs shall be derived from the PCIe PEXARIA5 [24].
TR_2010	**SFP cage –** for the connection to the WR network.
TR_2020	**WR dircuitry –** This can be either implemented directly on the carrier or by a separate add-on board like the WREX1 [30].
TR_2030	**Main JTAG connector –** A micro-USB connector on the PCB of the carrier board is mandatory. If the space permits, it should be placed on the front panel.
TR_2040	**Secondary JTAG connector –** Not mandatory, but may be implemented.
TR_2050	**USB** – A separate micro-USB connector, preferably on the front panel. It is connected to a on-board USB controller.
TR_2060	**Connector for logic analysis**
TR_2070	**MAC address** – A MAC address with Organizationally Unique Identifier (OUI) of the manufacturer is required for each FTRN.
TR_2080	**Power** – By default, FTRNs are powered via the host-bus (except standalone FTRN).
TR_2090	**External power** – Capability to be used as standalone receiver. An additional connector for external power using 12V should be included in the PCB layout of all FTRNs.
TR_2100	**Front panal LEDs** – Depending on the available space, 4-8 LEDs shall be implemented on the front panel and connected to the FPGA via a single-ended line, see TR_2140. Examples are • Power / WR Link Activity • WR Link Up • WR Lock • PPS • Module o.k. • Bus Access – active, if there is a read or write access • IRQ – active, if an interrupt is requested to the host system
TR_2110	**On-Board LEDs –** The LEDs shall be connected to the FPGA via single-ended lines, see TR_2140. • 8 user defined LEDs on the carrier

	- 4-8 user defined LEDs on the mezzanine
TR_2115	**Power LEDs** – Power LEDs for all voltages on carrier and mezzanine shall be implemented. They are connected to the relevant voltages.
TR_2120	**I/O LEDs –** For bi-directional I/O, two LEDs per I/O line shall be implemented on the front panel, see TR_3100. For uni-directional I/O, one LED per I/O line shall be implemented on the front panel, see TR_3570. The LEDs shall be connected to the FPGA via single-ended lines.
TR_2130	**LED colour** – The colour of LEDs matters and is defined by the GSI Timing Team.
TR_2140	**LED functionality** – Defining and implementing the functionality of LEDs (except power LEDs) is done by the GSI Timing Team.
TR_2150	**Push buttons and HEX switch** – All form factors shall have two push buttons and one HEX switch on the PCB. They are connected to the FPGA via a single-ended line. Defining and implementing the functionality is done by the GSI Timing Team ("Reset Button").
TR_2160	**CRC check** – GSI presently (February 2014) investigates the possibility of gateware CRC verification, see section 3.7.2. If this possibility is confirmed, the FPGA CRC_ERROR pin shall be connected to a dedicated pin defined by the GSI timing team.
TR_2170	**Front Panel Text** – Front panel text such as engraving must be approved by the GSI Timing Team.

Table 5: Common features for all form factors

The following electronic components shall be used. Table 6 sometimes uses the word "should" instead of "shall. This is only done to avoid a dead-lock in case a component is deprecated or no longer available. The components "shall" be used, as long as they are available.

Number	Description of the Requirement
TR_2500	**FPGA –** All FTRNs should use 5AGXMA3D4F27I3N FPGAs, but dye revision A must not be used. If required, a model with a different pin-out may be used, but speed grade and temperature range shall not be changed. The connection of signals to FPGA pins should follow the schematics described here [31]. The use of Arria II GX families such as the EP2AGX125EF29C5N is no longer considered.
TR_2510	**Flash memory –** Flash memory for gateware and firmware. Serial flash N25Q512A13GF840E should be used. The design should use the circuitry described in [31]. Informative: For Arria II GX FPGAs, M25P128-VME6GB should be used.
TR_2520	**ROM –** The 1-wire ROM DS18B20U+ should be used as temperature sensor with serial number. Each carrier and each mezzanine board shall be equipped with such a temperature sensor. The design should use the circuitry described in [31] [32].
TR_2530	**EEPROM –** The 1-wire EEPROM DS28EC20P should be used for hardware model, hardware revision number, calibration data and MAC address. Each carrier and each mezzanine shall be equipped with such an EEPROM. The design should use the circuitry described in [31] [32].
TR_2540	**USB controller –** The USB peripheral controller CY7C68013A-

	56BAXC should be used This is required as serial interface for the WR console and for providing a Wishbone to USB bridge at the same time. The design should use the circuitry described in [31].
TR_2550	**Mezzanine connectors –** The connectors QMS-052-05.75-L-D-A should be used to connect mezzanine boards to carrier boards. The design should follow the example described in [31] [32].
TR_2560	**Power supply –** DC-DC regulators such as LTM4619, LTM4620 or LTM8023 should be used. The exact type depends on the power requirements of the FTRN and the voltages provided by the host system. The design should follow the examples described in [31] [32].
TR_2570	**WR circuitry –** The WR circuitry shall use components identical to the ones used by the SCU3 [23] or the WREX1 [30] board. The circuitry may be implmented on-board or via a add-on board.

Table 6: Important electronic components.

Commonly used single-ended I/Os are used to trigger external equipment or as input to the ECA or the TLU as described in section 2.1. These signals shall be available via the front panel of all form factors and to be implemented as follows.

Number	Description of the Requirement
TR_3000	**Reference design –** The implementation of single-ended I/Os should be similar to the one on the mezzanine PEXARIA5DB1 [32].
TR_3010	**I/O is digital.**
TR_3020	**Signal type –** Signal level shall be LVTTL single-ended.
TR_3030	**Connectors –** LEMO 00 type shall be used.
TR_3040	**Bi-directional –** All I/Os shall be bi-directional.
TR_3050	**Connection to FPGA –** • I/Os are connected to the FPGA, but with buffers in between. If an I/O is implemented on a mezzanine board, the buffer shall be implemented on the mezzanine board too. • For each I/O, the buffer is connected with two pairs of differential signals to two pairs of differential FPGA pins. • Termination should be controlled via a single-ended signal. • Direction should be controlled via a single-ended signal.
TR_3060	**Load –** As output, it shall be possible to drive loads of 50 Ω.
TR_3070	**Termination -** As input, a configurable termination – on or off – is required.
TR_3080	**Bandwidth -** I/Os shall support frequencies up to 200 MHz.
TR_3090	**Signal quality -** I/Os shall support the requirements of the GMT, see TR_325.
TR_3100	**LEDs –** Two LEDs shall be used per I/O. One indicates the direction, the other one indicates activity. See TR_2120.

Table 7: Properties of single-ended signals.

To use longer cables or to profit from the precision provided by the GMT, single-ended are not sufficient but differential signals must be used. However, this is not required on all form factors, see Table 1. Differential signals are described in the following.

Number	Description of the Requirement
TR_3500	**Reference design –** The implementation of differential signals should be

	similar to the one on the mezzanine PEXARIA5DB1 [32].
TR_3510	**I/O is digital.**
TR_3520	**Signal type –** Signal level shall be LVDS (pairs)
TR_3530	**Connectors –** not specified.
TR_3540	**Uni-directional –** All I/Os should be uni-directional
TR_3550	**Connection to FPGA –** I/Os are connected to the FPGA, but with buffers in between. If an I/O is implemented on a mezzanine board, the buffer shall be implemented on the mezzanine board too.
TR_3560	**Load –** As output, it shall be possible driving loads up to 50 Ohms.
TR_3570	**LEDs –** One LED shall be used per I/O to indicate the activity. See TR_2120.

Table 8: Properties of differential signals.

3.2.2. VME64x FTRNs

Number	**Description of the Requirement**
TR_4000	**Reference design –** A reference design derived from PEXARIA5 [24] does not exist. On-board circuitry should be derived from PEXARIA5. The host-bus to Wishbone bridge should be derived from the VME board VETAR2 [25].
TR_4010	**Size –** FTRNs in VME form factor shall be 1 slot x 6U x 160 mm VME64x modules that operate in VME64x crates.
TR_4020	**Compatibility –** It shall be possible to operate VME64x FTRNs in standard VME crates with modified rails. The FTRN shall implement two additional HEX switches to select its VME address.
TR_4030	**Power -** All voltages required on the VME carrier board shall be generated from the voltages provided by a standard VME backplane.
TR_4040	**Single-ended I/O –** The majority of I/O shall be implemented on a mezzanine board. A small number (about 2) shall be implemented on the carrier board.
TR_4050	**P1 and P2 –** P1 / P2 VME bus connectors with 5 rows. All pins should be connected to the FPGA, if possible.
TR_4060	**P0 –** A P0 connector is not required.
TR_4070	**Front panel and handles –** Front panel with holes for I/O of carrier and mezzanine. Hot plug 64x handles are required.
TR_4080	**Minimum access modes –** At least A32 (AM: 9) must be implemented. Other VME access modes (A16, A24, BLT, MBLT) shall be implemented, if requested by the GSI Timing Team.
TR_4090	**VME SBC** – The FTRNs in VME form factor are expected to be compatible with VME Single Board Computer (SBC) endowed with the Tundra® TSI 148 PCI/VME bridge . The VME SBC configures the VME Slave FTRNs in order to set the IRQ level/vector, and accesses the WB bus. The VME interface of the FTRNs interfaces a WB master and WB slave. The WB master is used for accessing the WB bus and the WB slave exposes the MSI interrupts to the VME SBC. Support for MSI implies support for legacy interrupts.
TR_4100	**Other VME SBCs** – The FTRNs in VME form factor are also expected to be compatible with other standard VME SBCs like RIO4 (CES) or

	IPV from IOxOS. Here, a FTRN driver or Test SoftWare (TSWD – see TR_1060/1070) is not required.

Table 9: VME64x form factor.

3.2.3. MTCA FTRNs

Number	Description of the Requirement
TR_5000	**Reference design –** The MTCA form factor shall be derived from the PEXARIA5 FTRN as a reference design [24].
TR_5010	**Size –** FTRN in MTCA form factor shall be developed and implemented, single-width, mid-height, with respect to three use cases**.**
TR_5020	**Use Case MTCA.0 –** It shall be possible to configure, monitor a FTRN via PCIe, including IRQ – from any third party userland application such as a FESA class. IRQ generation is the only action towards the userland application that can be triggered by the FTRN via the MTCA.0 backplane. All other actions to the other interfaces of the FTRN must be fully supported. The FTRN must be compatible with MTCA.0, including read/write access as well as IRQ generation via the PCIe bus.
TR_5090	**Single-ended I/O –** Digital I/O lines on the front panel can be implemented using a mezzanine module or directly on the carrier board.

Table 10: MTCA form factor.

3.2.4. PMC Mezzanine WR Receivers

While in all form factors mentioned so far the WR standard and timing functionality is implemented on a main board, PCI Mezzanine Card (PMC) receivers are piggy back boards for other controllers (e.g. VME, I/O controllers). The main function of the PMC form factor board is to generate interrupts on the PCI bus at a given time.

Number	Description of the Requirement
TR_6000	**Reference design –** The implementation of PMC form factor shall be derived from the PEXARIA5 FTRN as a reference design [24].
TR_6010	**Single-ended I/O –** The PMC FTRN shall implement single-ended digital I/Os. The digital I/Os should be implemented directly on the PMC mezzanine.
TR_6020	**Signal type –** Signal level shall be LVTTL.
TR_6030	**Host bus –** A PCI bridge with 33 MHz clock and 32 bit width is sufficient.
TR_6040	**Power –** The PMC mezzanine must support the two supply voltages 3.3V and 5V.

Table 11: PMC form factor.

3.2.5. PCIe FTRNs

FTRNs in PCIe form factor shall be developed featuring

- PCIe bus connector

- Prepared for a mezzanine I/O card
- Every PCIe FTRN must include a mezzanine module for digital I/O.

A prototype of the PCIe form factor is already available [24].

3.2.6. Standalone FTRNs

A host-independent FTRN is called **standalone FTRN**. Such FTRNs shall be provided in two versions: Rack mountable modules and individual units.

- Rack mountable standalone FTRNs shall feature a small display and means for input like switches, buttons, dials or keypad. Display, switches, buttons, I/O... should be implemented using add-on boards. Rack mountable standalone FTRNs shall be developed as 19'' crates and the height should not exceed 2U. It must be possible to access the FTRNs remotely. It must be possible to disable remote access via the front panel.
- Handheld FTRNs could be mounted to racks by cap-rails or used as mobile devices. The form factor of such a FTRN should be similar to the EXPLODER2C developed by the department of Experiment-Electronics (CSEE) at GSI [15]. Power supply voltage shall be 12V over external pins. Fans must not be included and passive cooling should be sufficient.

3.3. Mezzanine Modules

For each form factor supporting mezzanine cards, one mezzanine card for signal I/O must be delivered.

3.4. WR Starter Kit

A WR starter kit is provided by the GSI Timing Team to allow a quick setup of a WR demo for development and integration of gateware, firmware and hardware. The WR starter kit shall include at least two WR nodes and optionally one WR switch.

3.5. Device Drivers

See section 3.1.10.

3.6. Other Issues

This section is mainly informative except section 3.6.4, which needs to be respected.

3.6.1. WR Network on Copper

WR on copper is not developed. If radiation becomes an issue, radiation hard optical fibres shall be used.

3.6.2. Phase of BuTiS w2 Clock

As listed in Table 1, FTRNs shall be able to synthesize a clock signal related to the BuTiS c2 clock. The phase of the synthesized w2 and w0 clock signals must line up with the PPS pulse per default. Phase shifting of the w2 clock signal with respect to the w0 pulse should be possible by a discrete value, if possible. For details on the BuTiS system, see also [19].

3.6.3. microSD Slots

The possibility of loading gateware and firmware from microSD is not required. Gateware and firmware are loaded from the flash memory on the FTRN carrier board.

3.6.4. Host Bus Bridging

Bridge chips between the FPGA and the host bus interface shall not be used.

As a guideline, the following test has been done involving the full stack (see TR_1060). A userland program [34] has been used to read timestamps from the TLU. Within one EtherBone cycle, three 32 bit registers are read and one 32 bit register is written. A few million cycles are performed to determine the average time for the full cycle containing the four operations. The following presents the status as of February 2014.

PCIe

The present driver does not yet support non-blocking read requests or DMA. On a Dell T3500 (Debian Wheezy) the cycle time is about 8.3 µs (120kHz) for the PCIe FTRN [24].

USB

The cycle time is about is about 134 µs (7.5 kHz) for the FTRNs of form factors PCI, VME and standalone [24][25][15], independent of the host computer. This could be improved, although this would break backward compatibility to form factors prior to PCIe FTRN [24]. Older designs miss a clock signal between USB chip and FPGA.

VME – RIO4

Using a RIO4 SBC from CES and a VME FTRN [25], a cycle time of about 16.5 µs (60kHz) is achieved using A32 access mode. BLT or MBLT access modes will improve the performance for this use case.

VME – MEN A20

Using an A20 SBC from MEN and a VME FTRN [25], a cycle time of about 32.7 µs (30kHz) is achieved using A32 access mode.

3.6.5. Bluetooth

A Bluetooth option for configuring FTRNs is not required.

3.6.6. Fine Delay on FTRNs

Fine delay of 1ns offers a benefit over the 8ns granularity provided the WR clock domain. Fine delay features for input and output are directly implemented in the FPGA, see section 3.1.7. Circuitry using fine delay chips is considered inappropriate by the GSI Timing Team.

3.7. Open Issues

Open issues shall be cleared before design completion.

3.7.1. Rate Divider

This is out-of-scope: If the rate of executed actions by the FTRN does not fit the purpose of the application, it must be investigated what would be the best solution.

- Rate divider functionality as proposed in [28].
- Implementation of „lossy" ECA action channels.
- Polling of Action Queues (instead of IRQs).
- Rate reduction by the ATL.
- Rate divider for digital output channels.

3.7.2. Run-time Gateware Check

This is out-of-scope: Although FTRNs are not intended for use in environment with an increased radiation level, it shall be investigated, if the correctness of the gateware can be verified with techniques like scrubbing or CRC tests.

3.7.3. Local and Remote Operation of Standalone FTRNs

This is out-of-scope: A concept for local and remote operation of rack mountable standalone FTRNs shall be investigated.

4. Acronyms and Definitions

The following acronyms and definitions are used in this document. For a full list of acronyms, please refer to the list of abbreviations for controls.

- Action: This is an activity executed by a FTRN. Activities are scheduled locally by the ECA and are performed by receiving components attached to the ECA. Examples of actions are raising an IRQ line or sending a digital output pulse via a LEMO 00 connector.

- Condition: Conditions define how received commands are mapped to actions by the ECA.

- Command: An input to the ECA. Typically, commands are received from the data master via the timing network.

- Firmware: compiled software code. Firmware is execute by a (Soft-)CPU. For WR designs, a Soft-CPU is frequently embedded in the gateware.

- Gateware: synthesized HDL code. Gateware is configuration for programmable logic and loaded into the FPGA of a timing node, such as a FTRN. In many cases, gateware includes a Soft-CPU such as the LM32 [17].

- GSI Timing Team: The Timing Group within the Controls Department at the GSI Helmholtzzentrum für Schwerionenforschung GmbH (GSI), Darmstadt, Germany.

- Form Factor Specific Test Facility: Allows mass testing of FTRNs for each form factor
- Test-Bed: provided by the GSI Timing Team. The test-bed system is a complete timing system including a timing master, a timing network and Timing Receivers.
- Clock Master: in charge of providing exact timing
- Data Master: in charge of defining real time operation constraints and distributing commands within these
- Management Master: in charge of network supervision and slow control
- Timing Master: Includes the clock master, data master and management master
- ACCNET - ACCelerator NETwork
- AOI - Automated Optical Inspection
- API - Application Programming Interface
- ATL - API Timing Library
- CPU - Central Processing Unit
- ECA unit - Event Condition Action unit
- FAT - Factory Acceptance Test
- FEC - Front-End Controller
- FECo - Forward Error Correction
- FESA - Front-End Software Architecture
- FMC - FPGA Mezzanine Card
- FTRN - FAIR Timing Receiver Node
- GMT - General Machine Timing system
- GSI - GSI Helmholtzzentrum für Schwerionenforschung GmbH
- HDL - Hardware Description Language
- HID - Human Interface Device
- MAC - Media Access Control address
- MTBF - Mean Time Between Failure
- OS - Operating System
- OUI - Organizationally Unique Identifier, part of a MAC address
- PCB - Printed Circuit Board
- PMC - PCI Mezzanine Card
- PTP - Precision Time Protocol
- PWM - Pulse Width Modulation
- SAT - Site Acceptance Test
- SCU - Scalable Control Unit

- SFP - Small Form-factor Pluggable
- SPEC - Simple PCI express card
- SVEC - Simple VME FMC Carrier
- TAI - Temps Atomique International
- TLU - Timestamp Latch Unit
- TSWD- Test SoftWare, form factor Dependent
- TSWI - Test SoftWare, form factor Independent
- USB - Universal Serial Bus
- VHDL - Very High Speed Integrated Hardware Description Language
- WB - WishBone bus
- WR - White Rabbit
- WR-PTP - White Rabbit PTP

I. Attached Documents

List of abbreviations for controls (Abbreviations_Controls.pdf).

II. Related Documentation

[1] F-GS-B-01e, FAIR General Specifications

[2] F-CS-C-01e, FAIR Common Specification "Accelerator Control System"

[3] F-DS-C-05e, FAIR Detailed Specification "General Machine Timing System"

[4] WR-PTP Repository: White Rabbit core collection – http://www.ohwr.org/projects/wr-cores

[5] CERN Open Hardware License – http://www.ohwr.org/projects/cernohl

[6] F-DS-C-01e, FAIR Detailed Specification "FEC software framework (FESA)"

[7] The architecture of an active database management system,
Dennis McCarthy and Umeshwar Dayal, SIGMOD '89 Proceedings of the 1989 ACM SIGMOD international conference on Management of data

[8] F-DS-C-03e, FAIR Detailed Specification "Settings management system"

[9] GNU General Public License – http://www.gnu.org/copyleft/gpl.html

[10] WR Specifications – http://www.ohwr.org/documents/21. Draft for comments: http://www.ohwr.org/attachments/306/WhiteRabbitSpec.pdf.

[11] WR Node Functional Specifications – http://www.ohwr.org/

[12] WR PCIe Node SPEC, repository – http://www.ohwr.org/projects/spec

[13] Open Hardware Repository – http://www.ohwr.org

[14] Etherbone core for WR, repository – http://www.ohwr.org/projects/etherbone-core

[15] EXPLODER2C, http://www.gsi.de/work/organisation/wissenschaftlich_technologische_abteilungen/experiment_elektronik/digitalelektronik/digitalelektronik/module/font_end_module/exploder/exploder2c.htm.

[16] M.Sc. Thesis of T. Wlostowski, Warsaw University of Technology, 2010/2011 – http://www.ohwr.org/documents/80

[17] LatticeMico32 soft CPU for Xilinx platforms – http://www.ohwr.org/projects/lm32

[18] FAIR Technical Note, No 20100510, "Integration and Linking between FAIR Timing System and BuTiS".

[19] BuTiS – Bunchphase Timing System
http://www-bd.gsi.de/dokuwiki/lib/exe/fetch.php?media=meetings:moritz_butis_technikforum_2010_10_28.pdf

[20] F-DG-C-03e "Software Architecture Guideline"

[21] Wishbone B4 Documentation
http://cdn.opencores.org/downloads/wbspec_b4.pdf

[22] F-TG-ET-01e, FAIR Technical Guideline “Electrical Design Rules and Regulations”

[23] F-DS-C-02e, FAIR Detailed Specification “Equipment Interface and Control”.

[24] PEXARIA5.
https://www.gsi.de/work/fairgsi/common_systems/csee_electronics/digitalelektronik/digitalelektronik/module/pci_pci_e/pexaria/pexaria5.htm

[25] VETAR2.
https://www.gsi.de/work/fairgsi/common_systems/csee_electronics/digitalelektronik/digitalelektronik/module/vme/vetar/vetar2.htm

[26] F-DS-C-16e, FAIR Detailed Specification “Motion control FECs”

[27] F-DS-C-17e, FAIR Detailed Specification “Serial equipment control FECs”

[28] Technical Note: LOBI FAIR Timing Receiver Node Requirements

[29] WR VME Node SVEC, repository – http://www.ohwr.org/projects/svec

[30] WREX1 White Rabbit add-on board – see
http://indico.cern.ch/event/238322/contribution/7/material/slides/0.pdf

[31] Schematics of PEXARIA5.

[32] Schematics of PEXARIA5DB.

[33] http://www.ipc.org.

[34] https://www-acc.gsi.de/svn/bel/timing/trunk/development/eb_demo/eb_tlu_demo.c.

III. Document Information

- 7 July 2014: destilled from the detailed specifications of the “Timing Receivers”, F-DS-C-06e.