ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEBR003

WHITE RABBIT TIMING: THE NEW CERN ACCELERATOR TIMING SYSTEM

G. Moscardi*, M. Cejp, A. Dujović, T. Gingold, E. Gousiou,
F. W. Hoguin, I. Kozsar, G. Kruk, A. Zeising
European Organization for Nuclear Research (CERN), Geneva, Switzerland

Abstract

After more than 30 years of service, CERN's accelerator timing system is being renovated, moving from the existing distribution infrastructure based on the RS-485 technology and legacy hardware modules, to a new one based on White Rabbit.

Developed at CERN, White Rabbit Timing (WRT) is a generic toolkit composed of the White Rabbit Event Node (WREN) - a System-on-Chip based hardware module, and the corresponding software stack. WRT allows transmission and reception of messages, along with an arbitrary payload (key-value pairs). The received messages enable the generation of triggers in the form of software interrupts and electrical pulses, with sophisticated and highly configurable triggering patterns. WRT seamlessly integrates time derived from the radio frequency used for particle acceleration, with WRENs capable of locally generating beam orbit and bunch clocks, as well as broadcasting Beam Synchronous Timing (BST) streams over dedicated optical links.

We present the key concepts of WRT, its architecture, multi-layered distribution network layout, functionalities and usage at CERN. We also draw a potential path towards a turn-key timing system based on WRT that could be deployed anywhere for scientific or commercial applications.

THE CURRENT CERN TIMING SYSTEM

A particle accelerator is composed of many pieces of equipment which must work together to produce, accelerate, maintain and collide particle beams. It is necessary to synchronize the operation of devices such as magnets, beam instrumentation or radio-frequency (RF) cavities.

At CERN, synchronization is achieved by sending messages on dedicated timing networks. These messages are used to generate electrical pulses triggering accelerator equipment as well as to instruct the corresponding software to perform appropriate actions depending on the beam parameters. To cover the wide range of required timing resolution (from milliseconds down to about 25 ns), multiple timing systems are currently in operation (Fig. 1).

General Machine Timing

General Machine Timing (GMT) distributes so-called *Central Timing Events* (CTIMs), such as *Cycle Start*, *Beam Injection*, *Cycle End*, etc. These events are received by Front-End Computers (FECs) equipped with a Central Timing Receiver (CTR) module [1].

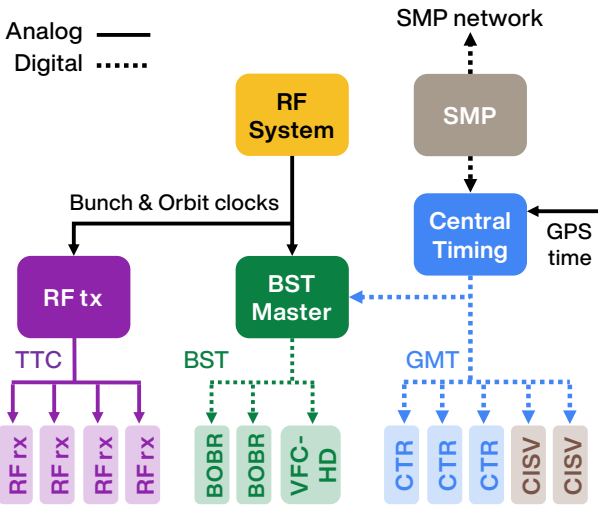

Figure 1: Overview of the current timing systems at CERN.

CTRs are used to decode GMT frames and to generate *Local Timing Triggers* (LTIMs) relative to the Central Timing Events in the form of pulses or software interrupts, that can be used to precisely control software running on a FEC or hardware devices connected to it.

At the physical level, GMT is based on a RS-485 serial link running at 500 kbit/s on a copper cable [1], with one transmitter, hundreds of receivers, as well as signal amplifiers and converters.

Radio-Frequency Clocks

Many equipment types require synchronization with Radio-Frequency (RF) signals so as to be synchronous with the circulating beam. These signals consist of the *Bunch Clock* and the *Turn/Orbit Clock* (also known as F_{rev}) and, in the case of the LHC, they are distributed as analog signals over a dedicated optical network.

Beam-Synchronous Timing

The Beam-Synchronous Timing (BST) system is CERN-specific and solely dedicated to the SPS and LHC accelerators. Its purpose is to broadcast timing signals that are required to synchronize the different beam instrumentation systems with the circulating beam. For this purpose, messages are encoded using the Bunch Clock provided by the RF system. Additionally, the BST system rebroadcasts selected GMT events and data, for the benefit of those receivers that are not simultaneously connected to the GMT network.

BST is distributed to hundreds of receivers over dedicated optical fibers and specialized boards developed at CERN.

* giorgio.moscardi@cern.ch

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEBR003

Safe Machine Parameters

For the safe operation of the accelerator complex, several mission-critical parameters (such as beam intensity or machine energy) must be distributed around the SPS and LHC. These are called *Safe Machine Parameters* (SMP) and are sent via a dedicated network to critical receivers (interlock systems), but they are also rebroadcast on the GMT network [2] in order to reach a wider audience, despite not strictly constituting a timing system.

Issues and Limitations

The current timing system has evolved gradually since serial transmission came into use in the 1980s [3] using a combination of off-the-shelf technology and in-house solutions. While it has been successfully used in operation for over 30 years, some of its limitations are becoming increasingly apparent.

GMT, in particular, has very low bandwidth and only part of that is usable for timing data, allowing for the transmission of eight 32-bit frames per millisecond [4]. New equipment is constantly put into service on the accelerators and occasionally this requires new events to be transmitted, resulting in complex workarounds to overcome this limitation. Moreover, the RS-485 standard limits the length of the physical links to around 1.2 km: this is not sufficient to cover the distance between some CERN installations, so dedicated devices must be used to amplify the signals or to convert them to optical signals and back. The standard also limits the number of nodes per network segment to 32, with *repeaters* required to overcome this limitation. Due to the extensive deployment, the number of such devices is large and the incurred maintenance costs are significant.

The length of the cabling also introduces delays in signals that cannot be ignored for larger accelerators (SPS, LHC). In order to compensate for these, it is necessary to run a manual calibration procedure for every FEC, comparing the local clock signal with that generated by a very stable portable cesium clock [5].

The above limitations imply that dedicated cables must be setup to carry timing information for every accelerator, effectively making the GMT network a collection of different physical networks. Part of the equipment requires connections to multiple networks, creating yet more complications in cabling layout.

Event frames are sent in due time and, because of their small size, cannot carry all necessary contextual information (cycle type, particle type, beam destination, etc). The latter is sent in separate frames and must be associated with events by the FEC library, contributing to the overall complexity.

The CTR modules used to receive timing are also not without limitation: most outboard equipment only understands simple electrical pulses, yet the module provides only 8 outputs with this capability. When more output signals are needed, multiple CTR modules need to be installed in a single FEC, increasing costs and reducing the number of available slots in FEC crates.

Maintenance of the CTR modules has also become problematic with some of the components now obsolete, making repairs or production of new boards difficult.

Ultimately, the proliferation of different timing networks, along with the necessity of re-broadcasting information between them, presents significant opportunities for optimization and improvement.

WHITE RABBIT TIMING

The White Rabbit project was started at CERN in 2008, aiming at developing new technology that could be used to renovate the CERN timing system [6]. The outcome of this effort is a deterministic Ethernet network capable of achieving sub-nanosecond time synchronization among its nodes [7]. In 2020, White Rabbit became part of the Precision Time Protocol standard (IEEE 1588-2019) as a new PTP Profile named High Accuracy.

White Rabbit Timing (WRT) builds on top of this network to enable distributing the information necessary to synchronize processes in a particle accelerator. It borrows some concepts from the existing timing system at CERN, while taking advantage of the vastly increased bandwidth and processing power of the network nodes to provide more abstraction and flexibility.

At its core, WRT is a generic messaging system, allowing to broadcast messages (events) containing their future due time along with an optional set of parameters (fields) constituting contextual information.

Core Concepts

Contextual information is carried in WRT by **fields**. A field is a variable defined by its name, type and either a range (for numeric fields) or a set of enumeration values. Examples of fields are `BEAM_ID`, defined as a 32-bit integer, and `PARTICLE` defined as one of `PROTON` (when accelerating protons), `PB82` (when accelerating lead ions), etc.

In a **context**, concrete values are assigned to fields for a given interval of time, for example, one *basic period*[1] in a cycling accelerator (most fields tend not to change during this time). A context is identified through its *ID*, an 8-bit rolling counter.

Only one context can be in force at any given moment. This establishes the idea of a "current context", permitting applications to retrieve the current values of timing fields at any time, for example in reaction to an external trigger (timer, user action).

Events constitute an essential part of the timing system. In WRT, an event has a name and a nanosecond-resolution timestamp known as the *due time*; since the WR network ensures that all nodes have a common notion of time[2], this timestamp is sufficient for global synchronization. Moreover, each scheduled event contains a reference to a context from

[1] A basic period is the smallest time unit for accelerator scheduling at CERN and corresponds to a duration of 1.2 seconds.

[2] Specifically, International Atomic Time – TAI, which can be converted to UTC after accounting for leap seconds.

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEBR003

which it inherits all field values. An event can also contain additional field values of its own.

Finally, to account for different needs of different accelerators, and to partition the timing information, each field, context and event belongs to a **domain**. On a conceptual level, domains are isolated from each other: references between events, contexts and fields are constrained to the same domain. Typically, equipment is triggered by events from a single domain (linked to the accelerator it is installed in), although in some cases it must rely on events from two or more domains, as it is in case of devices installed in beam transfer lines. Figure 2 shows more concrete examples of the basic timing objects.

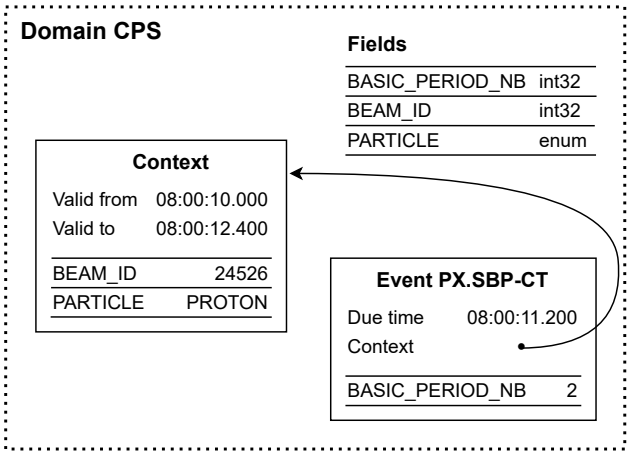

Figure 2: Example of the CPS domain (CERN Proton Synchrotron) with a selection of fields, a context and an event. Due to inheritance of field values, the event has effective values of `BEAM_ID` = `24526` and `PARTICLE` = `PROTON`.

Distribution of Timing Information

Events and contexts, along with the corresponding field values, are sent over the WR network in Ethernet frames containing their identifiers and values (Fig. 3). At CERN, in the cycling accelerators[3], the actual accelerator cycles are scheduled dynamically a few seconds in advance, hence the contexts and events are broadcast to receivers immediately after being programmed. However, this time advance can be adjusted according to the particular application, as long as it is greater than the maximum network latency (Fig. 4).

The definitions of events and fields (containing IDs) must be known by both the transmitter and receiver software. This information is static during operation, making it relatively easy to distribute, for example using a network file system.

Frames are transmitted by a *source*, which is identified by its *domain* and a *transmitter ID*: the former allows receivers to filter out timing data from accelerators they are not interested in, while the latter uniquely identifies the different senders within a single domain.

[3] Cycling accelerators work based on predetermined *supercycles*, i.e. an arrangement of cycles that is played in a loop, usually defined by means of a *Beam Coordination Diagram*.

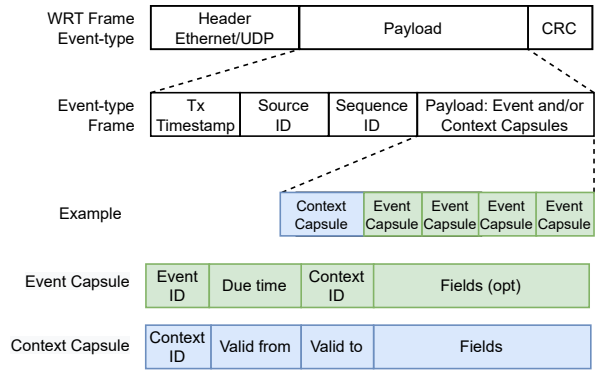

Figure 3: Structure of a WRT frame.

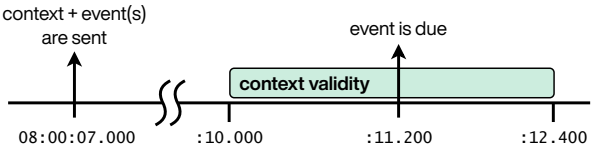

Figure 4: Timeline of the situation in Fig. 2.

It is important to note that the network does not inherently provide real-time reception guarantees. Reliable operation can nevertheless be achieved through a combination of deterministic maximum propagation time and tight control over the number of independent transmitters.

The exact maximum message propagation latency at CERN is still to be measured, however with cable lengths reaching tens of kilometers and several layers of WR switches, it is estimated to be around 500 µs.

BST and SMP Traffic

Besides events and fields, WRT is able to provide information for the distribution of RF clocks for the SPS and the LHC, and can replace the existing BST system altogether. Since 2018, the SPS Low Level RF Beam Control system is based on White Rabbit technology [8] and is capable of injecting Frequency Tuning Words (FTWs) into the WRT network. In addition, work is ongoing to redesign the RF Beam Control system of the LHC [9] for the same purpose.

Similarly, an upgrade of the SMP system (SMPv2) will inject WRT frames into the same network in the form of ordinary timing events [10].

Event Delivery Offset

Some types of equipment need to start acting before an event is due; for instance, the current in a magnet cannot change instantaneously but must be gradually ramped up by a power converter. In GMT, where events are sent in due time, this issue is addressed by sending dedicated "warning" events, preceding the actual event (e.g. a beam injection warning event is sent 100 ms before the injection event). As mentioned earlier, WRT events and contexts can be sent out on the wire in advance and the resulting time margin can be exploited to trigger actions with a predefined anticipation, rendering warning events unnecessary (Table 1).

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEBR003

Table 1: Example of Event Delivery with a Negative Offset

PX.SBP-CT (example)		
Due time	08:00:11.200	
Delivery offset of subscriber	−160	ms
Delivery time	08:00:11.040	

WHITE RABBIT EVENT NODE

The *White Rabbit Event Node* (WREN) was conceived as a general-purpose WRT node that can act either as a receiver or as a transmitter [11]. It is available in VME and PCIe form factors and a PXIe variant is at the prototype stage.

WREN modules are also able to decode the RF frequency tuning words and to reconstruct the RF Orbit and Bunch clocks. These clocks can then be combined with timing information to generate BST signals to be provided to existing beam instrumentation equipment.

Local Timing

A WREN receiver is able to produce Local Timing triggers. These are essentially highly-configurable trains of pulses accompanied by optional software interrupts (Fig. 5). LTIMs can be *loaded* immediately upon configuration (in which case they are called *Immediate*) or upon the reception of a Central Timing event (*Event-bound*). In the latter case, an optional *condition* can be specified in order to restrict the loading only to events matching user-defined criteria, consisting of up to three field predicates arbitrarily combined with boolean operators.

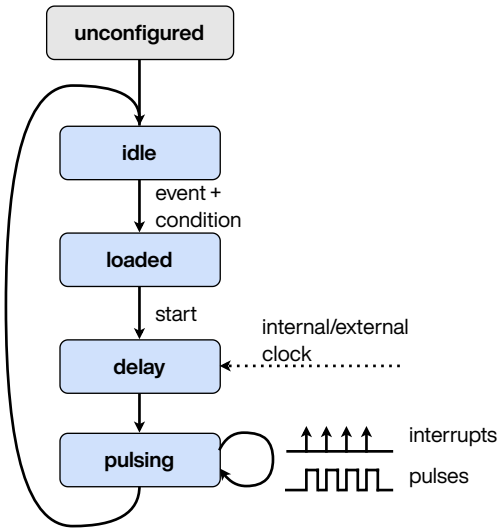

Figure 5: LTIM state diagram for the general case. In practice, many LTIMs will use simpler configuration with automatic start (effectively skipping the *loaded* state) and produce a single pulse or interrupt.

Once loaded, LTIMs can be *started* either automatically, by the WR-provided Pulse-Per-Second (PPS) signal, by a pulse on an input connector or by another LTIM. After an optional delay, the LTIM will then produce pulses with the desired width and frequency until the specified number of pulses has been reached or until a stop signal is asserted (pulse on an input connector or another LTIM).

The delay and pulse frequency can either be based on an internally-produced clock rate (PPS-aligned 1 Hz, 1 kHz, 1, 10 and 40 MHz, 1 GHz) or on an externally-provided signal. Thanks to the automatic link delay calibration feature of White Rabbit, the internally-produced clocks are guaranteed to be synchronized on all the receivers.

Once configured, LTIMs are handled completely by the WREN, guaranteeing timely load, start and stop operations. WREN modules support up to 32 logical channels, each of which can host one or more LTIMs. Such channels are grouped in four blocks of eight, whose outputs can further be combined through an AND, NAND, OR or NOR logic gate and connected to a number of physical output connectors.

A single WREN module is equipped with 6, 14 or 38 integrated connectors, depending on the form factor, with some freely configurable in input or output mode. It also provides an interface for an external patch panel with 32 output connectors.

For setups requiring more than 32 LTIM channels or input/output connectors, multiple WREN modules can be installed on the same FEC and controlled independently.

Event Tables

Event tables are a feature of the WREN when used as a transmitter. They provide a mechanism for pre-loading a sequence of events to be transmitted upon an external trigger – either an electrical pulse on an input connector or another event. An event table may contain two kinds of entries: `EVENT` and `WAIT`. An example is provided in Table 2.

An `EVENT` entry sends out an event with the specified due time and any provided fields. Since a table can be reused many times, only a relative due time is specified; at the time of sending, it will be summed with the current time to compute the absolute due time which is sent on the network. For the same reason, the event's context is not specified when loading the table; the transmitter keeps a history of recently sent contexts and automatically assigns the context which will be valid at the event's due time.

The second type of entry, `WAIT`, can be used to postpone the sending of subsequent events, since it is not always desirable to send all events at once.

When required, event tables can also be started programmatically.

THE SOFTWARE STACK

The software stack was designed from the ground up, deliberately avoiding dependencies on legacy APIs or constraints imposed by previous architectural designs. This approach enabled the creation of a clean, modular, and well-structured set of C++ classes and libraries.

One key architectural decision was to clearly separate the mechanisms considered universally applicable from those

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEBR003

Table 2: Example of a Timing Table. The Left Half Shows the Table Definition As Loaded Into the Timing Transmitter, While the Right Half Shows Example Timestamps if the Table Was Started at 09:00:00.000

Command	Event name	Due time (rel.)	Wait amount	Execution time	Due time (abs.)
`EVENT`	HX.ABC-CT	200 ms		09:00:00.000	09:00:00.200
`WAIT`			100 ms	:00.000	
`EVENT`	HX.DEF-CT	200 ms		:00.100	:00.300
`WAIT`			50 ms	:00.100	
`EVENT`	HX.GHI-CT	200 ms		:00.150	:00.350
`END`				:00.150	

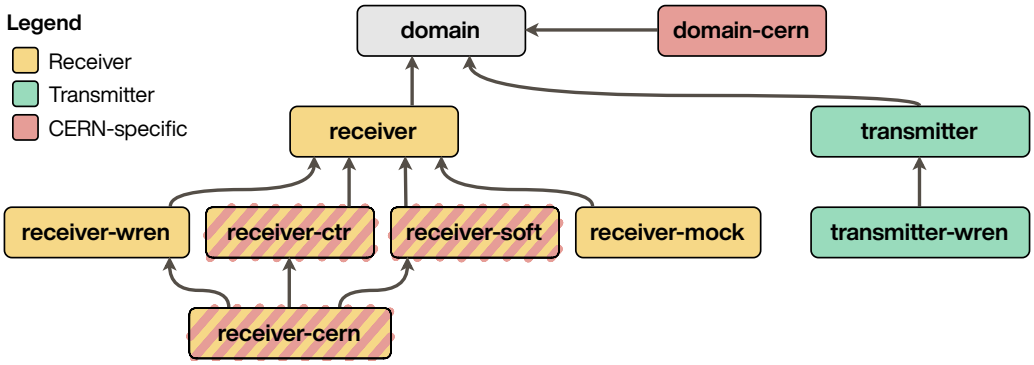

Figure 6: Dependency tree of the packages comprising the White Rabbit Timing software stack.

tailored to CERN's specific use of the timing system. For example, the timing distribution API is kept distinct from the LTIM management API. This separation enables the system — whether in part or in its entirety — to be reused beyond CERN, supporting other scientific or commercial applications, either as a dedicated timing solution or more broadly as a messaging system capable of generating triggers.

Another significant design choice was to define a basic `Receiver` interface with multiple implementations to address different needs: in addition to the WREN implementation, one serves as a wrapper for the legacy CTR libraries, and another is dedicated to testing. This approach streamlined the migration from GMT, enabling users to adapt their software to the new API ahead of the full deployment of WRT, without the burden of maintaining separate versions.

An overview of the packages is given in Fig. 6 and more details are provided in the following sections.

domain

This package defines classes corresponding to the core domain objects: `Domain`, `Field`, `Context` and `Event`, as well as functions for lookup of these objects by name or ID.

domain-cern

This package contains definitions of the domains, fields and events used at CERN. For each of those, a corresponding global variable is defined, allowing names to be looked up and verified at compile time and ensuring all references are handled in a type-safe manner. The package consists predominantly of code automatically generated from a central timing configuration database.

The library is distributed in the form of a shared object on a network file system, which allows adding new events or fields without recompilation of client code. However, it also places stringent constraints on the maintenance of this library, since backward compatibility must be preserved.

receiver

This package defines a `Receiver` interface enclosing the common operations that all receivers must support, such as:

- subscribing to/unsubscribing from contexts and events;
- waiting for events and contexts to be received;
- retrieving the current time.

Notably, LTIM manipulation is not part of the interface.

receiver-wren

This package contains the `WrenReceiver` class, which implements the `Receiver` interface for WRT using a low-level driver library to interface with the WREN. This class is enriched with APIs allowing to manipulate LTIMs and I/O pins, to retrieve diagnostic information (network and time synchronization status, version numbers) and to fetch the contents of various history buffers used for troubleshooting.

receiver-ctr

This package enables generic client software to work transparently with WRT as well as with the legacy GMT system, without the need to maintain dedicated versions, therefore simplifying the GMT-to-WRT migration.

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEBR003

receiver-mock

This module contains the `MockReceiver`, also derived from `Receiver`, which is meant for simulations and tests of client software without depending on specific hardware. It takes event definitions with time offsets from an XML file or through the API and simulates receiving them a specified number of times or until stopped.

receiver-soft

This is a CERN-specific `Receiver` implementation meant for applications not requiring the highest timing accuracy or where a WREN receiver cannot be used due to incompatibility of the form factor or other limitations. This includes System-On-Chip (SoC) platforms, which are expected to grow in number throughout the accelerator complex in the coming years.

Contexts and events are transmitted in advance over a TCP connection via a standard Ethernet network from a server hosting a *Timing Distributor* service. The main factor determining the overall accuracy is the receiver's host system clock synchronization, which can be improved by using PTP.

transmitter

Similarly to *receiver*, this module defines a `Transmitter` interface for transmitting timing data on WRT.

transmitter-wren

This module uses the low-level WREN driver to implement the actual WRT transmitter. It is used by the central timing software to send events and contexts and takes care of assembling them into WRT frames and transmitting them on the network.

TESTING AND DIAGNOSTICS

Since both the hardware and software stack are completely new, significant effort was put into making sure the whole system behaves as expected and into creating tools allowing to investigate and troubleshoot any unexpected behavior.

LTIM Integration Tests

LTIM behavior is controlled by 16 largely independent settings (booleans, integer values and enums). Behavior is also influenced by interactions between LTIMs – for example, one acting as a start trigger, another as a clock, and a third as a stop signal, each with their own settings. While this flexibility supports more use cases, it leads to a large number of possible configurations.

To validate LTIM operation within the timing system, a set of parameterized test cases is executed with varying configurations, producing over 1100 total test cases. These cover a broad range of scenarios, with emphasis on edge cases, to verify correct triggering and data accuracy.

The tests are implemented in Java using a JUnit-based integration framework, which manages data acquisition and uses a timestamp calculator to model expected LTIM behavior. The test suite can run on any network-connected machine, independently of the WREN module testbed.

wrentest

wrentest is a command-line-based, non-interactive diagnostic application for WRT, written in C++. Its feature set ranges from displaying general information and highlighting common configuration problems to providing more specific introspection and manipulation of WREN modules.

The application functionalities are grouped hierarchically into a tree structure of commands and sub-commands. The leaf command of any invocation may accept positional and non-positional options. Bash shell completion is available to aid discovery of commands, their options, and arguments.

The *status* command lists general information, including the current state of the network link, synchronization, and time, while the *events* and *fields* commands show descriptions of CTIM events and their respective fields. The configuration of LTIMs and their relationships to the inputs and outputs of each module, are handled by the *ltim* and *io* commands. The *wait* command allows for the monitoring of CTIM events and LTIM software interrupts as they occur, while the *history* command enables examination of past occurrences of CTIMs, LTIMs, and the rising or falling edges of the signals of each hardware pin used for input and output. If an LTIM is deemed to be misbehaving, a *troubleshooting* command highlights possible problems that may then be resolved via modification of the LTIM and I/O configuration using their respective sub-commands.

WRT AT CERN

As of summer 2025, pilot installations of WRT have been deployed on a few dozen FECs. This has enabled early adopters to gain experience with the system while simultaneously providing valuable feedback. The new system operates in parallel with the legacy one. The migration of the SPS and LHC accelerators is planned to be completed by 2029 (end of Long Shutdown 3). The remaining accelerators will be migrated by the end of 2033 (Long Shutdown 4). By that point, all timing systems should be consolidated to a single WR network as shown in Fig. 7.

LEIR, PSB, PS and SPS

LEIR, PSB, PS and SPS (Fig. 8) are *cycling* accelerators. Events for these accelerators are scheduled by a single *Central Timing* system and are sent out roughly three basic periods in advance, allowing negative delivery offsets of up to 3.6 seconds: this provides a clean way to remove the usage of the "warning" events.

Most fields are grouped into the context and sent once per cycle. Only fields that change during the cycle are transmitted with the individual events.

Besides events scheduled in the supercycle, the accelerators also use *Asynchronous Events*: these are unpredictable

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEBR003

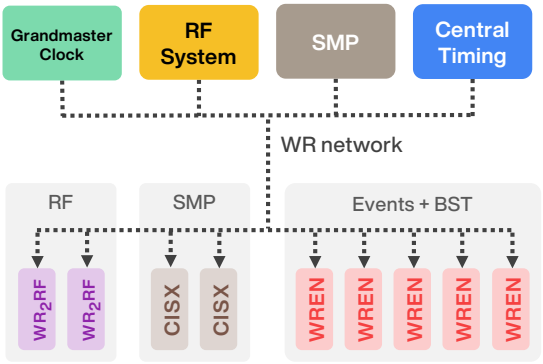

Figure 7: Conceptual diagram of an upgraded timing system based on a common White Rabbit network.

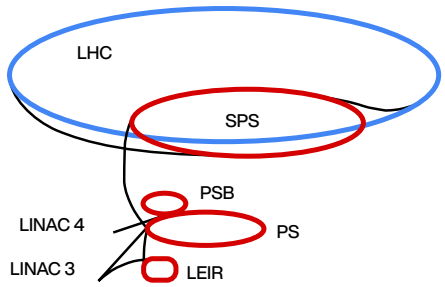

Figure 8: The LHC (in blue) and its injector chain (in red).

events that can be sent either manually by operators or automatically by devices (e.g. *Beam Dump*). These events are sent with a due time in the near future, only accounting for the network propagation time. On the receiver side, they are associated with the context that is valid at their due time, inheriting the corresponding field values.

LHC

Timing in the LHC is not based on supercycles but rather on separate injection, acceleration and collision phases. The transitions among these phases are initiated by operators making use of event tables. As for the other accelerators, asynchronous events are used for sudden conditions such as a beam dump notification coming from the interlock system.

Network Layout

The size of the planned WRT deployment at CERN is rather large, ultimately comprising nearly 2000 receivers interconnected by about 200 WR switches.

Since part of the accelerator equipment requires events from different domains, it was decided to distribute timing data for all the domains on a single network. This network is logically separated from the rest of the WR network using Virtual LANs (VLANs) and appropriate switch port configurations, but it still shares the common time reference provided by the *grandmaster* switch. Should the need for separate networks ever arise, it could be achieved through the use of additional VLANs.

The network layout loosely follows a tree topology over 5 layers. This was dictated both by the logistics of the deployment and by the need to keep the tree as shallow as possible, in order to limit the network propagation time.

As a precaution, traffic is generally only permitted towards the leaves of the tree, where all receivers are connected. This also provides a simple way to inject test traffic limited to the general-purpose network, used for laboratories and development devices. Further precautions were put in place to facilitate network diagnostics and troubleshooting.

A more detailed description of the timing network and its components can be found in [12].

CONCLUSION

WRT was designed to improve upon CERN's legacy timing system, incorporating technological innovations to address its limitations and expand its capabilities.

At CERN, it will be used not only to distribute timing events and fields but also to consolidate distribution of other accelerator-related information (RF clocks and SMP data).

WRT was designed with flexibility and extensibility in mind. It also has relatively modest hardware requirements and benefits from the open nature of the White Rabbit Collaboration[4]. Together, these features make it a highly reusable timing solution. It can serve as a timing system for particle accelerators beyond CERN or function as a versatile messaging and triggering platform.

REFERENCES

[1] J. Serrano, P. Alvarez, D. Dominguez, and J. Lewis, "Nanosecond Level UTC Timing Generation and Stamping in CERN's LHC", in *Proc. ICALEPCS'03*, Gyeongju, Korea, Oct. 2003. https://jacow.org/ica03/papers/MP533.pdf

[2] B. Todd, S. Gabourin, and R. Secondo, *Safe Machine Parameters: Interface to General Machine Timing*, CERN Internal Report EDMS 901688, 2011.

[3] C. G. Beetham, R. J. Lauckner, and C. Saltmarsh, "Overview of the SPS/LEP Fast Broadcast Message Timing System", in *Proc. PAC'87*, Washington D.C., USA, Mar. 1987, pp. 766–769.

[4] J. H. Lewis, J. Serrano, and P. Alvarez, "The LHC Central Timing Hardware Implementation", in *Proc. ICALEPCS'07*, Oak Ridge, TN, USA, Oct. 2007, pp. 400–402. https://jacow.org/ica07/papers/WPPB02.pdf

[5] T. Włostowski, *LHC CTR Timing Delay Calibration Procedure*, Internal CERN access only, 2014. https://confluence.cern.ch/pages/viewpage.action?spaceKey=HT&title=LHC+CTR+Timing+delay+calibration+procedure

[6] J. Serrano *et al.*, "The White Rabbit Project", in *Proc. IBIC'13*, Oxford, UK, Sep. 2013, pp. 936–942. https://jacow.org/IBIC2013/papers/THBL2.pdf

[7] *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Instrumentation and Measurement Society, 2020. doi:10.1109/IEEESTD.2020.9120376

[4] Launched in 2024 to unite all stakeholders of the technology and ensure its sustainability and ongoing development [13]

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEBR003

[8] T. Włostowski *et al.*, "White Rabbit and MTCA.4 Use in the LLRF Upgrade for CERN's SPS", in *Proc. ICALEPCS'21*, Shanghai, China, pp. 847–852, 2022. doi:10.18429/JACoW-ICALEPCS2021-THBR02

[9] T. Levens, "Beam Synchronous Timing over White Rabbit", presented at BI Technical Board GMT/BST via White Rabbit, Geneva, Switzerland, Mar. 2023, unpublished. https://indico.cern.ch/event/1249710/contributions/5251693/attachments/2603199/4495528/BI-TB_BSToWR.pdf

[10] S. Bolton *et al.*, "Development of a second-generation system for the reliable distribution of machine protection parameters", in *Proc. IPAC'24*, Nashville, TN, USA, pp. 3401–3404, 2024. doi:10.18429/JACoW-IPAC2024-THPG60

[11] T. Gingold *et al.*, "WREN: A Versatile White Rabbit Event Node for CERN's Timing System Renovation", presented at ICALEPCS'25, Chicago, IL, USA, Sep. 2025, paper WEPD017, this conference.

[12] M. Suminski, I. Kozsar, M. M. Lipinski, and J. Palluel, "Evolution of White Rabbit Network for large-scale deployment at CERN", presented at ICALEPCS'25, Chicago, IL, USA, Sep. 2025, paper TUPD070, this conference.

[13] The White Rabbit Collaboration, https://www.white-rabbit.tech/.