

White Rabbit - Architecture proposal

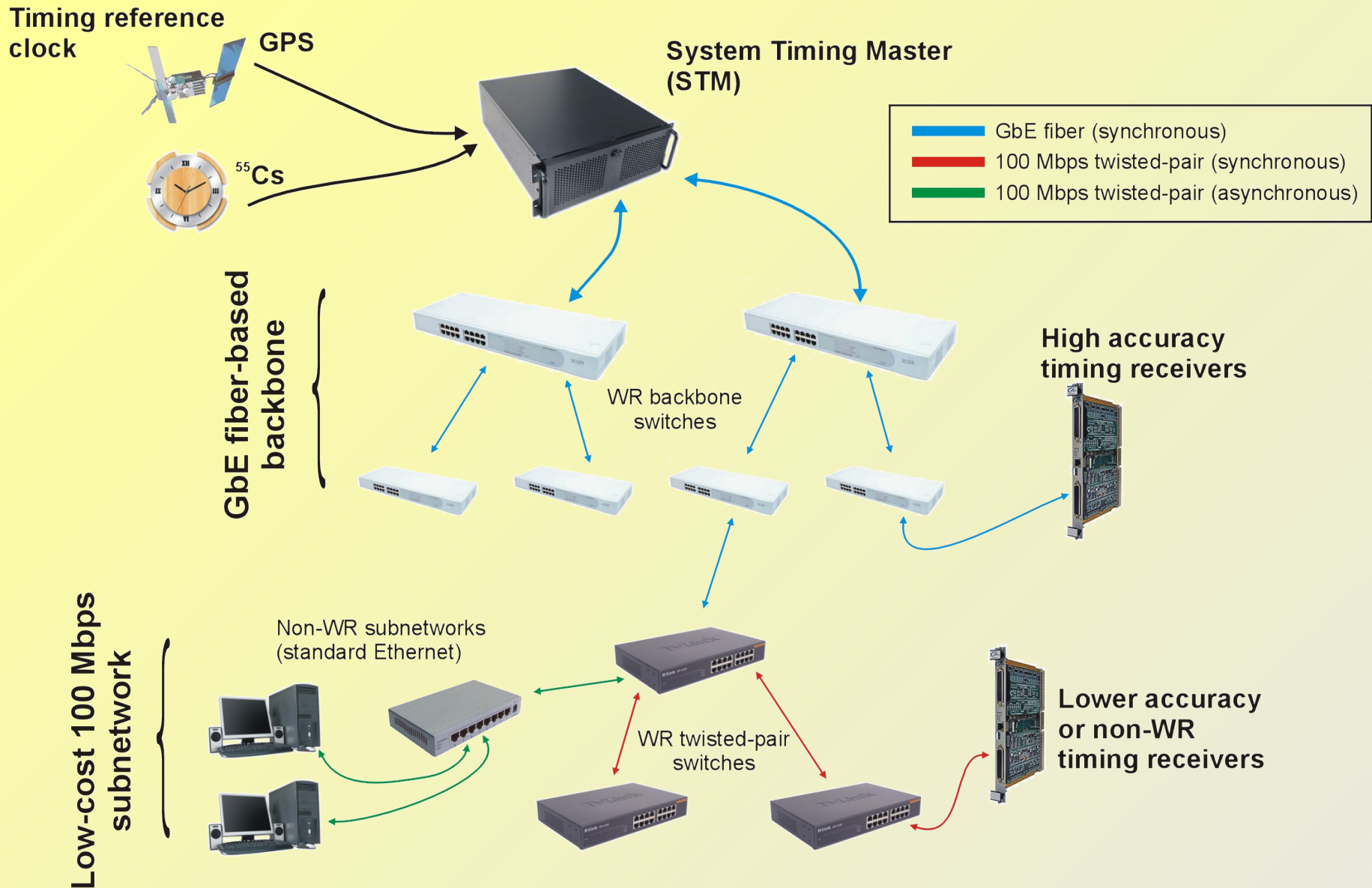


Tomasz Włostowski
CERN AB-Co-HT

Outline:

- **White Rabbit network architecture**
- Core component – WR backbone switch
- White Rabbit Protocol

WR network architecture



Topology

- multilayer star-like topology
- alternate paths for fault tolerance
- media: single-mode fiber or CAT5 twisted pair
- synchronous operation

System Timing Master

- single device for the whole network
- can be backed up by alternate STM
- provides time and frequency reference for all nodes in the network
- manages high-priority (HP) traffic – allows or disallows nodes to send HP frames
- sends time-critical timing and control messages like current CERN timing master does
- checks and maintains network integrity

Backbone switches

- gigabit Ethernet – based network backbone
- transmission medium – single mode WDM fiber (single fiber for both TX/RX)
- provide high throughput and very accurate (sub-nanosecond) timing
- usually work as network backbone, but can be interconnected directly with nodes which require high timing accuracy ($< 1\text{ns}$)

Twisted-pair WR switches

- low-cost, 100 Mbps twisted-pair solution
- for devices which are satisfied with lower timing precision
- interoperable with standard Ethernet gear (non-WR switches, routers, etc.)

Nodes

There are 2 kinds of nodes:

- dedicated hardware devices or standard PCs with special drivers for access to WR features
- non-WR devices – nodes and network infrastructure without access to WR features

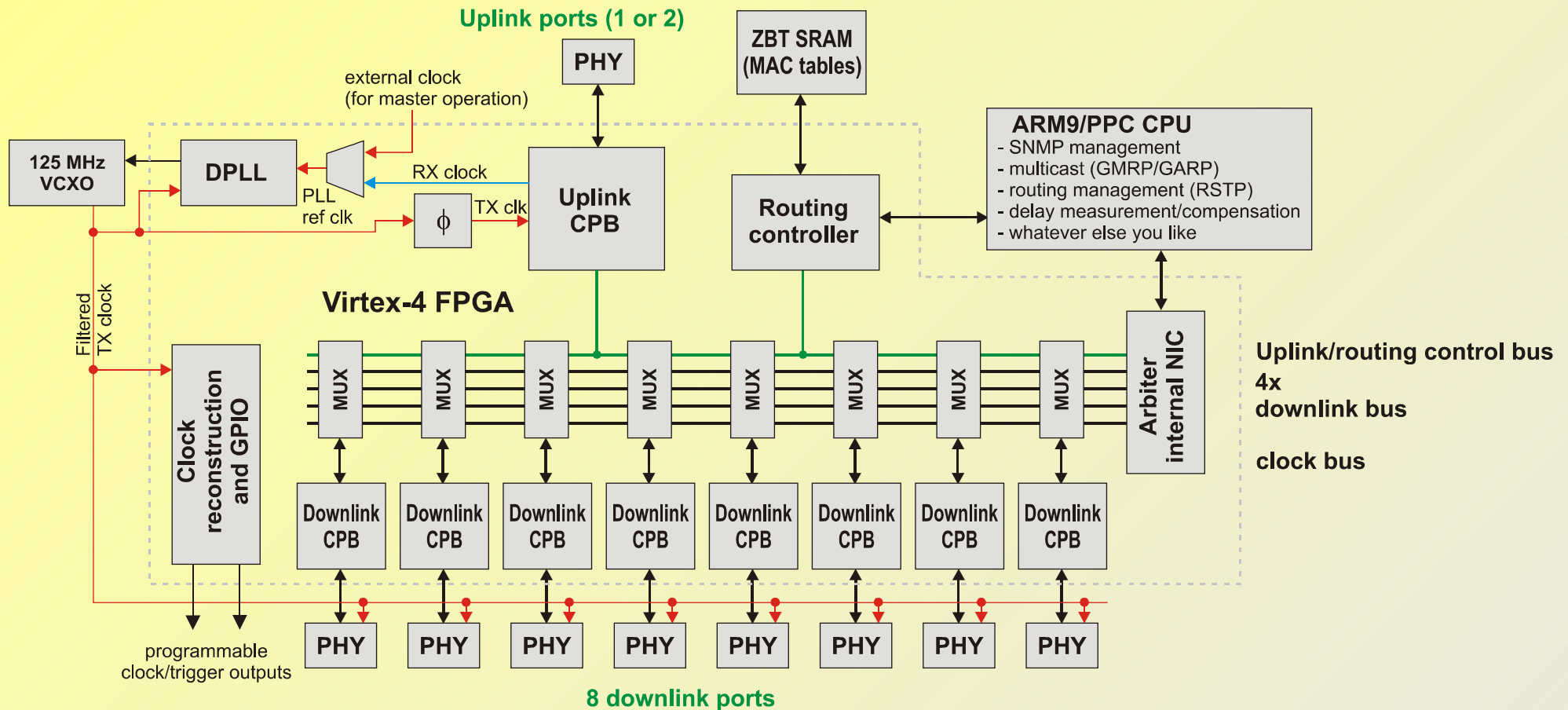
Outline:

- White Rabbit network architecture
- **Core component – WR backbone switch**
- White Rabbit Protocol

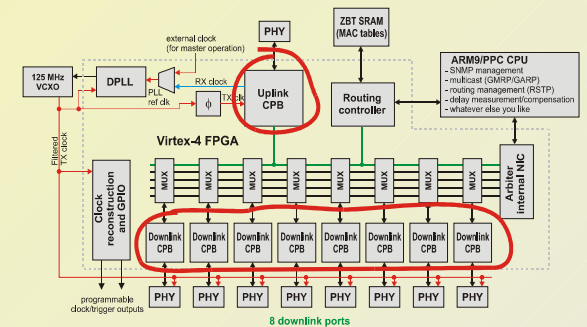
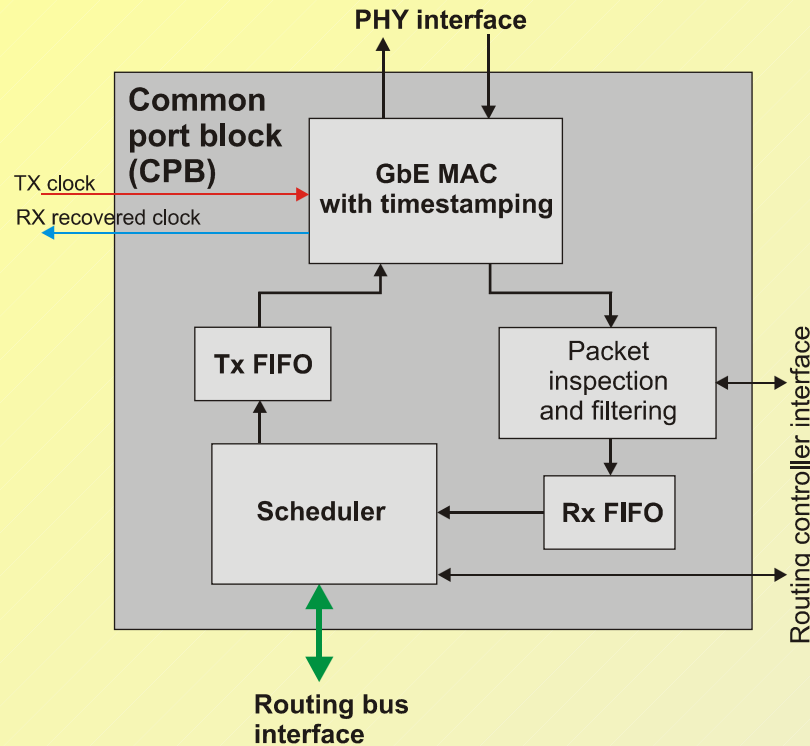
Core component: WR backbone switch

- Standalone gigabit Ethernet fiber-optic switch with 8 downlink ports and 1+1 uplink ports
- for non-WR devices, fully compliant with 802.1x
- synchronous operation for low-latency data/precision timing transmission
- built-in IEEE1588 boundary/master clock daemon

WR backbone switch – block diagram



Common Port Blocks (CPBs)



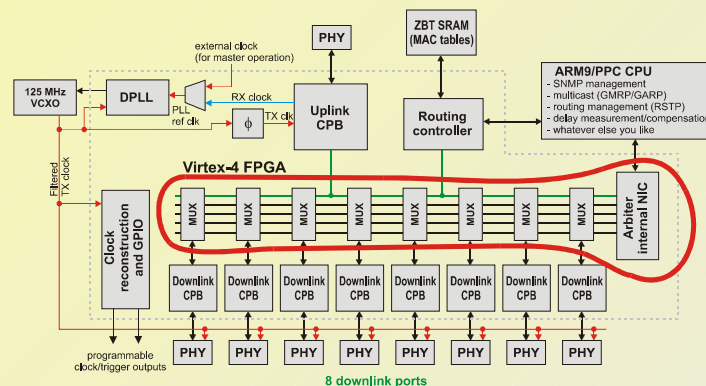
Act as interface between physical ports and internal switch routing circuitry

CPB elements

- **Synchronous GbE MAC** with IEEE1588-compatible packet timestamping – interface to PHY
- **Packet inspector** – detects incoming frames' addresses and types, then sends them to **routing controller**
- **Scheduler** – performs frame ordering, SP/non-WR fragmentation and decides when to send awaiting frames.
- **FIFOs** – TX/RX buffers. For high-priority data, usage of FIFOs is heavily constrained to obtain determinism.

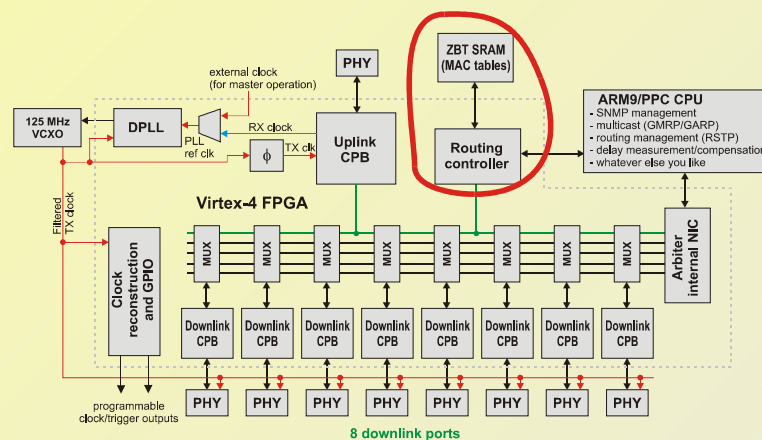
Routing busses

- 5 bidirectional 16-bit busses interconnecting CPBs
- 4 busses for routing between downlink ports and one bus for routing between uplink and downlink ports, also used for HP data.
- Busses are assigned by **bus arbiter** according to routing controller decisions
- Bus arbiter contains internal MAC for local switch CPU



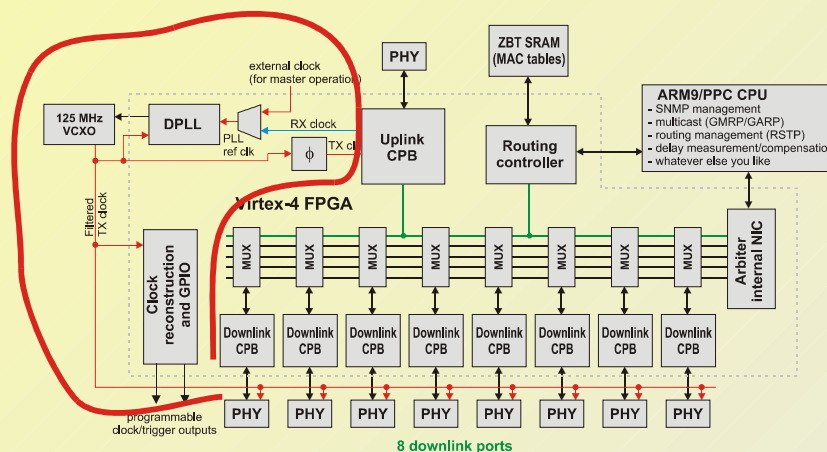
Routing controller

- Decides to which port incoming frame shall be sent
- Associates source addresses with respective ports
- Manages frame ordering and fragmentation



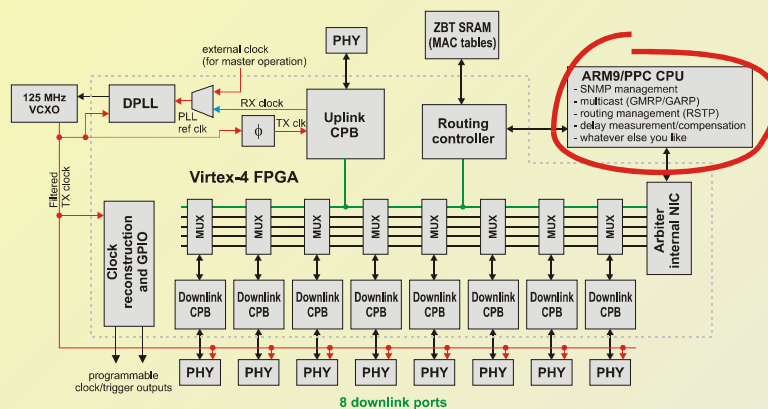
Clocking system

- Recovers reference clock embedded into Ethernet carrier from uplink port. This clock after delay compensation and filtering by DPLL is propagated to lower layers of network
- Lock-to-external reference mode for testing purposes
- Performs delay compensation between local clock and master clock
- Can sustain stable clock for short time (e.g. several hundred microseconds) in case of temporary link failure
- Can generate 16 synchronized programmable clocks/trigger signals



CPU

- ARM9-based external CPU with network controller built into switch fabric, using the same network as other WR devices
- Performs all delay measurement and compensation stuff. Runs local PTPv2 boundary clock daemon and WRP protocol daemon
- Handles allocation of multicast traffic/groups
- Handles network management and maintenance (SNMP, remote console).
- Changes uplink port in case of primary uplink failure
- **Does not** participate in routing. This is done entirely in FPGA hardware.



Outline:

- White Rabbit network architecture
- Core component – WR backbone switch
- **White Rabbit Protocol**

White Rabbit Protocol (WRP)

Our needs: Enable network nodes to use special, realtime features of White Rabbit network without breaking compatibility with standard Ethernet for non-WR devices.

Solution: WRP protocol!

White Rabbit Protocol (WRP)

- Simple, message-based protocol operating on **layer 2** (MAC) of OSI model.
- Easy to implement in hardware
- It's **not** a complete solution for control network. It only provides reliable, deterministic transport layer for data and timing **without** breaking compatibility with standards.

Main WRP tasks

- Providing reliable deterministic one-way transmission channels (without handshaking) for low-latency control and timing messages
- Precise delay measurement and reporting, fine (sub-nanosecond scale) transparent time transmission based on PTPv2 protocol and synchronous Ethernet
- Detection of WR-compliant devices in the network

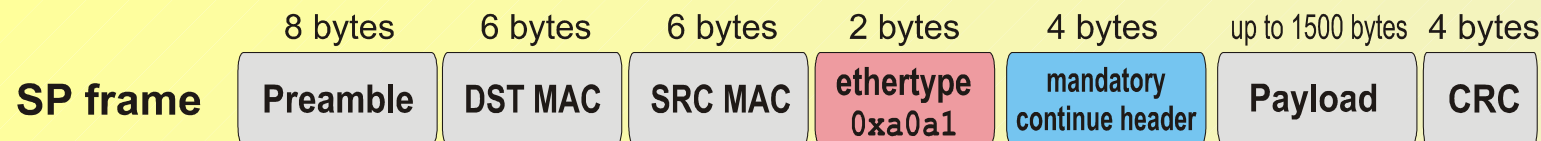
WRP Ethernet frames

- **SP (Standard Priority) frames** – frames having *ethertype* 0x0a0a1, used for exchange of WRP messages. They are handled like normal Ethernet traffic (e.g. queued and buffered), so their routing is not deterministic.
- **HP (High Priority) frames** – frames of *ethertype* 0xa0a0, which have absolute priority over any other traffic, forcing fragmentation of other frames if it's necessary. They are routed deterministically, with constant routing delay introduced by switches.
- **Non-WR traffic** – everything else in the network (TCP/IP, etc.), handled like SP frames.

SP and non-WR frames

- SP frames are used for exchange of WRP messages and to support frame fragmentation feature
- When HP frame arrives and in the same time SP or non-WR frame is being transmitted, it is immediately broken and HP frame is routed
- When HP frame transmission is done, switch or node sends rest of broken frame in special SP frame
- Fragmentation and reconstruction is transparent for higher-level protocols and devices incompatible with WR

SP frame structure

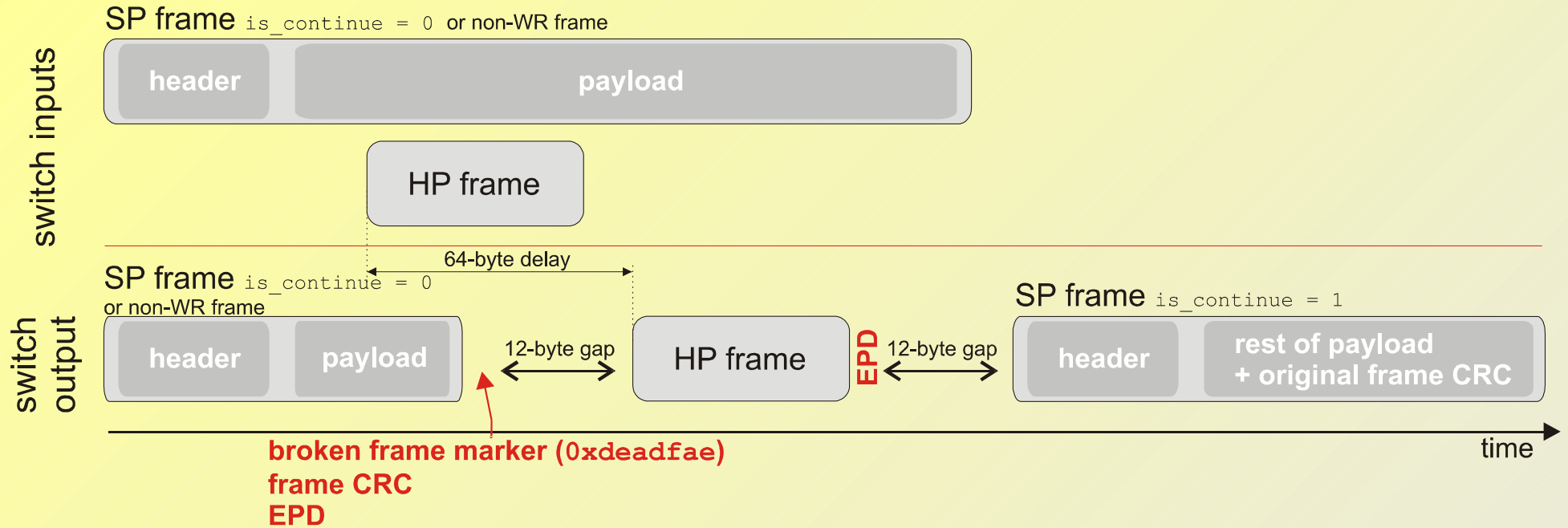


- **Special `continue` header:**

```
struct SP_mandatory_header { // mandatory SP header
    uint16_t is_continue;
    uint16_t continue_offset;
};
```

- Nonzero value of `is_continue` flag means that this SP frame contains next segment of previously received frame
- `continue_offset` field specifies at which offset previous frame has been truncated

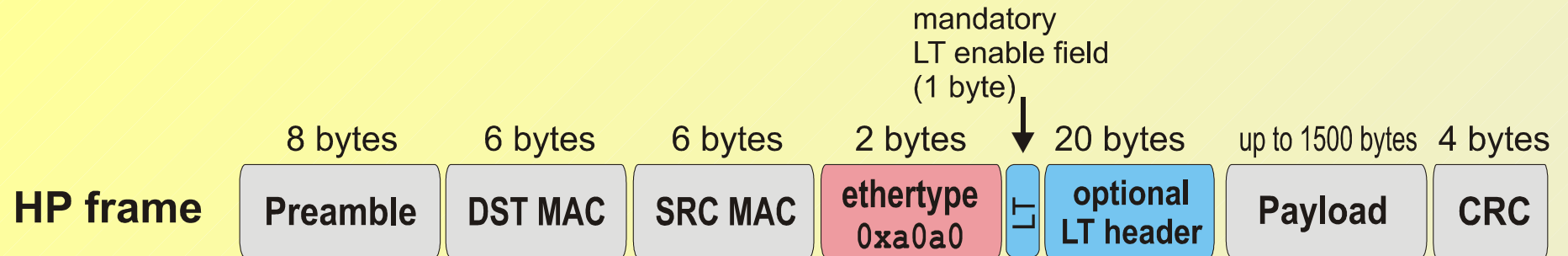
Frame fragmentation



HP frames

- Recognized **only** by unique *ethertype* value
- Used for high priority, crucial timing and control messages
- Erasure-proof LT encoding for non-handshaked protocols
- Physically, they are always broadcast to prevent overflowing FIFOs in switches, although they are addressed like normal Ethernet frames.

HP frame structure



Because most of time-critical messages cannot be acknowledged, we **must be sure** that they will always reach the destination even when link introduces some errors.

Solution: LT coding, protecting both from single-symbol errors and loss of full frames

LT encoding

LT stands for Luby Transform codes, a class of fountain codes. The idea of LT encoding is very simple:

1. Original message is split into N blocks, called

$$A_1 \dots A_N$$

2. Randomly chosen blocks are XORed to produce $m >$

$$N \text{ equations } X_1 \dots X_m :$$

$$X_1 = A_1 \text{ xor } A_3 \text{ xor } A_7$$

$$X_2 = A_1 \text{ xor } A_4 \text{ xor } A_5$$

3. $X_1 \dots X_m$ are sent in separate packets with equation coefficients

LT encoding (2)

We can reconstruct original message having only N correctly received frames by solving the equation system.

This method is particularly efficient for packet networks – even if full frames are lost (due to error in frame header/link signalling), we can still reconstruct original message!

WRP messages

- Simple, easily extendable NTLV (Name-Type-Length-Value) format, a bit similar to ASN.1 but much simpler
- 5 types of messages:

WRP_INVITE

WRP_INVITE_RESPONSE

WRP_ACK

WRP_REPORT_NODE

WRP_REPORT_DELAY

Compatibility

- Frame fragmentation can be implemented in software in network card driver – no special hardware is required for WR nodes
- Each WR switch runs full featured PTPv2 for timing synchronization with non-WR slaves (even nonsynchronous)
- If the device does not respond to WRP_INVITE message, the non-standard features (HP, fragmentation) are disabled by the switch for the port to which device is connected

Questions?

