SLS Timing System

Timo Korhonen

1. Introduction
2. Concepts, components and technology
3. System structure
4. Applications
5. Conclusions

Requirements and beyond

Fundamental task : Injection control

-reference generation, RF & AC line synchronization, distribution of fiducial signa

But: a timing system can be more:

-a means to synchronize operations in several IOCs

-operation sequencing tool

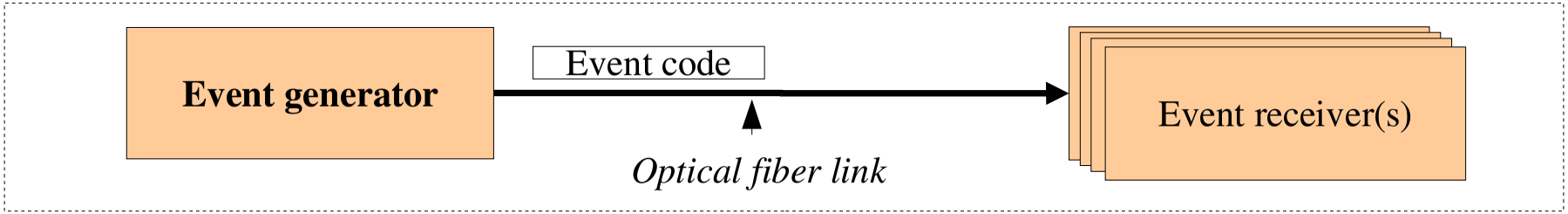
-global timebase (timestamp support)

-a means to bind a distributed system into a single coherent entity

ideal: to have "single cable" delivering all the timing to (practically)

all the IOCs

Concept of the Event System



-bare data frames are transmitted at the hardware frame rate to event receivers
-external stimuli are encoded to 8-bit event codes; when no stimulus is present, null frames are sent out

The **stimulus** to send an event can be:

-**hardware input pulse**

-**software event** (write to a register)

-an entry in an **event playback RAM** .

Upon receival of an event code the receiver can:

-output a pulse, of specified delay and width

-process an EPICS record

Each event receiver's response to an event can be independently programmed.

SLS Event system Technical Features

Technology: Gigabit Ethernet (physical layer)
short wavelength (860 nm) fiberoptic transceivers ,multimode (50/125 um) fiber up to 550 meter range
(1300 nm tranceivers could be used for longer range – up to 10 km)

50 MEvents/second (50 MHz event rate), 20 ns resolution.
Synchronized to the main RF oscillator through direct clock input

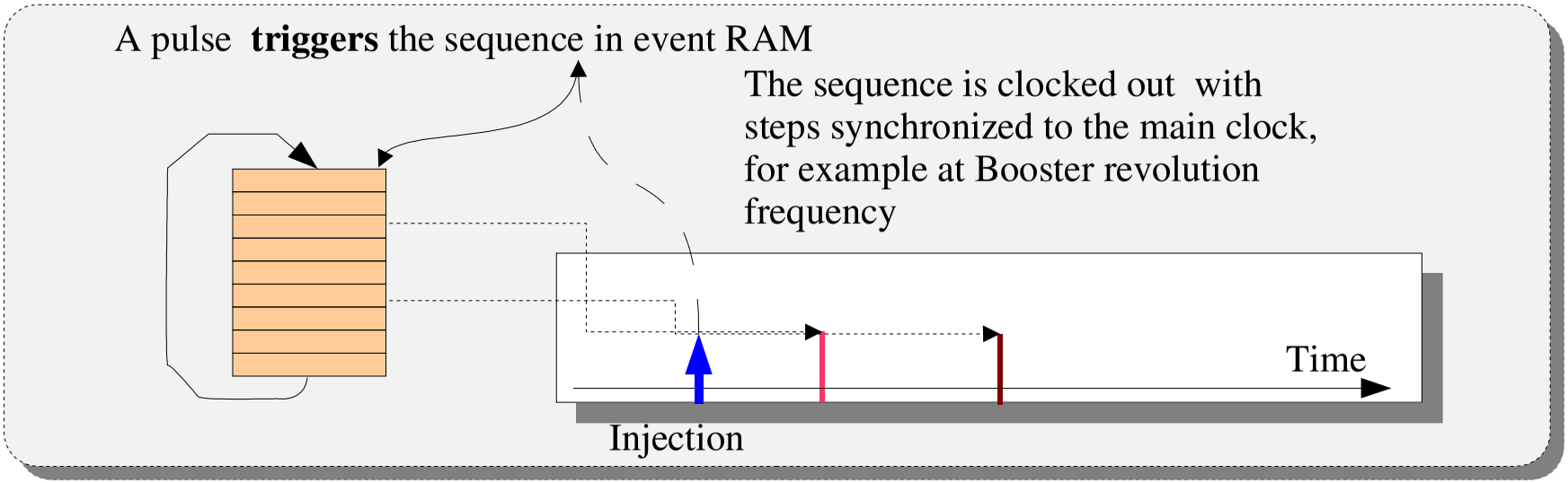
8 bits for events, independent 8 bits for a “distributed bus” (clock & other signal distribution)

XILINX Virtex FPGA for the logic, loaded from Flash ROM
(in-system reprogrammable)

FPGA code about 4000 lines of VHDL for EVR & EVG

Event Generator

3 possible event sources:
hardware inputs, register write, event RAMs



The RAMs are used for operation (injection) sequencing of SLS
-the step clock can be set (to multiples of revolution period)
-alternate mode: one RAM runs while the second is loaded.
-any sequence can be programmed through EPICS records.
A single reference signal input triggers the sequence. All other fiducials are then sent from the event RAM.

Event Receiver

-programmable pulse and set/reset "flip-flop" outputs.
-delay & width generators down to 20 ns resolution

-trigger outputs are phase locked to RF through the local Gigabit transceiver PLL. Output pulse jitter relative to RF 15 ps RMS.

-VME interrupt facility for software synchronization
(also a hardware-delayed interrupt for fine-tuned software delays)

-timestamp counter can count received frames or "tick" events. High resolution timestamp and event number are latched into a FIFO when received

Timestamp & IRQ facilities enable synchronous operations & data acquisition:
EPICS records can be processed when an event is received. The processed records can be timestamped with the time the event occurred (hardware time)
Needs some organization, but is powerful and very precise

Faces

Event Generator (EVG)

-address (A16) switches
-status LEDs
FPGA configured
card enabled
link (locked to stream)
event
frame error (latched)

AC line input (EVG Plus only)

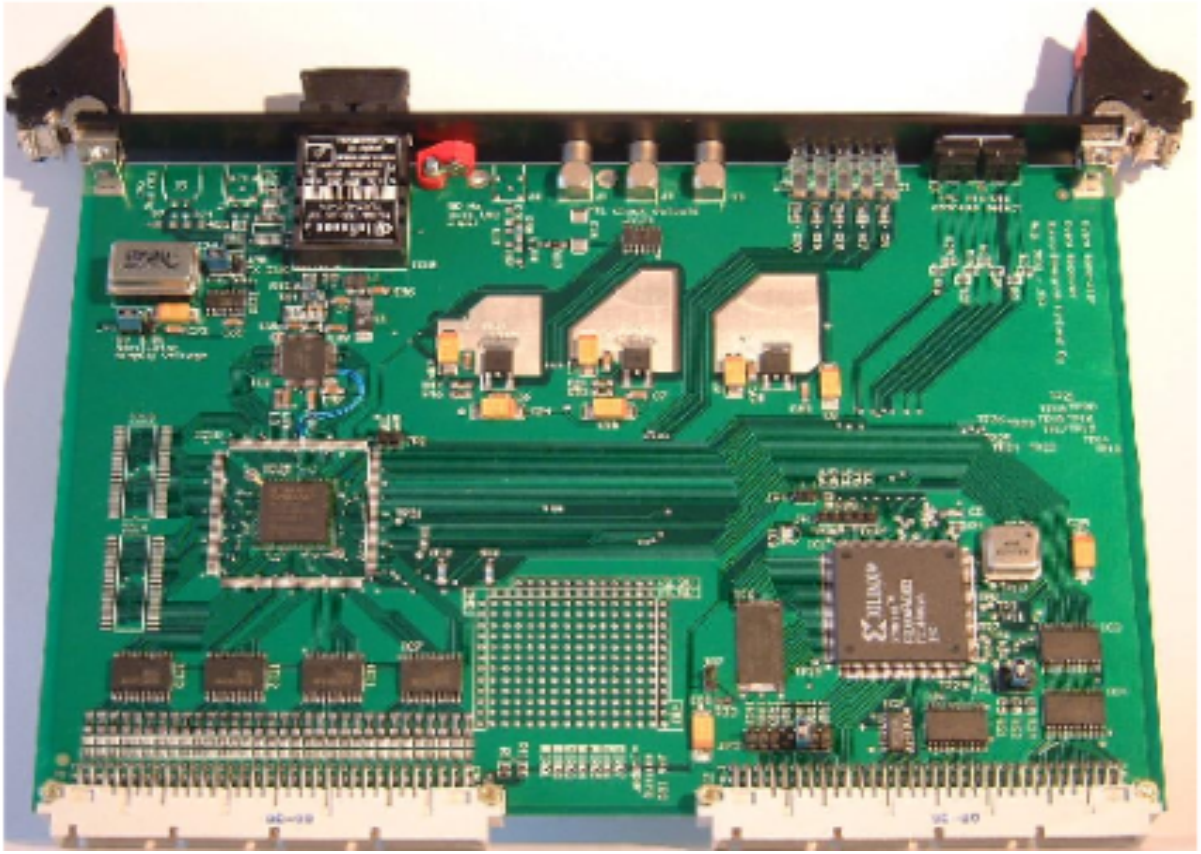
Fiberoptic connector
(SC), Transmit & receive

Reference clock input

Event Receiver (EVR)

Utility frequency outputs
1,5, and 10/100 MHz LVTTL

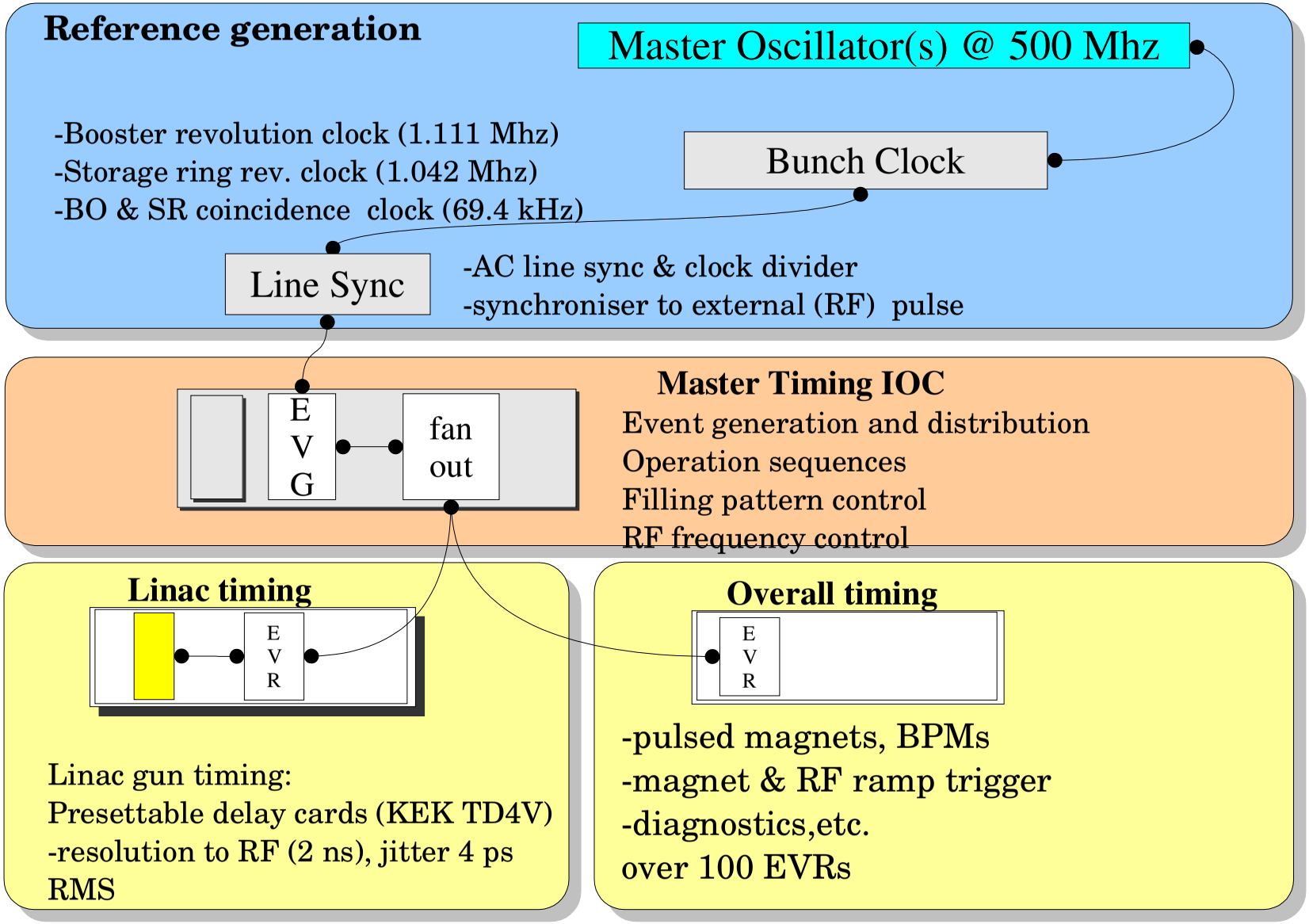
Event Receiver



Number of components
is quite small:
-High-density FPGA
(Xilinx Virtex, now Spartan)
handles the EVR functionalit
-Gigabit transceiver handles
the data traffic, framing and
phase locking
-VME logic in a CPLD

Outputs through backplane I/O & transition modules
(TTL, NIM, Optical, others...)

SLS Timing System Structure



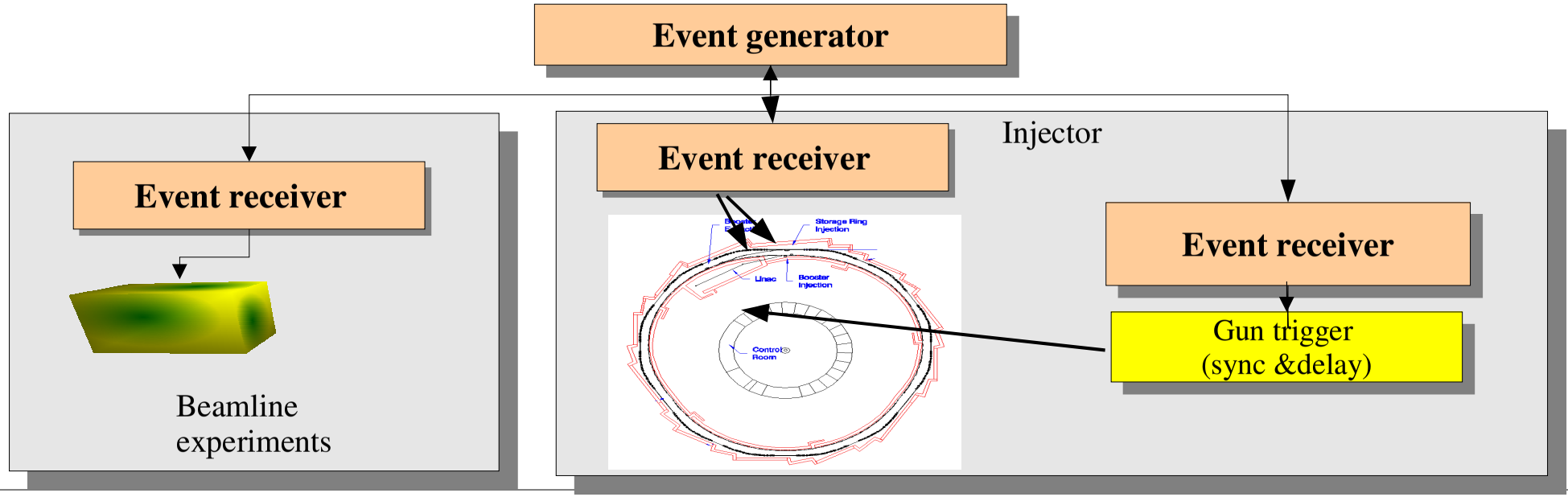
Synchronous applications

Often it is needed to do things synchronously in several IOCs.
Method:
-preset operation instructions to IOCs (into EPICS records,
possibly through channel access)
-synchronize with events: when event X comes, do actions defined for that event. Timestamp results with the event time.

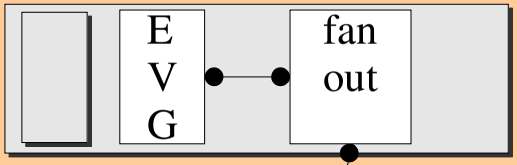
In other words, separate data & clock. Clock (=events) carries 'colour' information telling what is requested, in addition to the time.

Synchronous application example: Top-up Injection

The basic logic for top-up:
-inject every N:th cycle when the current is below a defined threshold
operations: e-gun trigger, SR injection kickers trigger enabled
Warn beamlines, send a “topup-on” event before, “topup-off” after injection
-method to feedback bucket charges in preparation: read charges, sort, send bucket numbers with lowest charges to relevant IOCs to be processed.
The processing (injections) are synchronized with events.

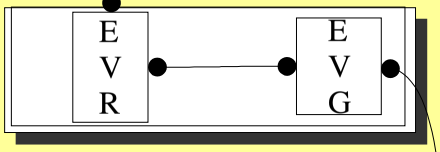


Beamline Timing



Master Timing IOC

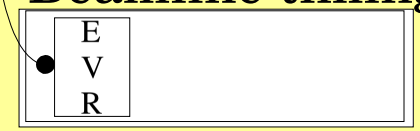
Event generation and distribution



Beamline master

Event receivers and
generators can be cascaded
-> add a sub-branch for a beamline

Beamline timing



Beamline can have its own timing events multiplexed with machine timing, but not interfering with other branches
Needs coordination of event allocation
Development for high-resolution EVR (RF regeneration) is in progress

Summary

-Existing EPICS support was a big advantage: the basic functionality was there from the beginning and we could incrementally add the features that we needed. (Substantial changes: at SLS, the event system is used for all timing, including injection control)
However, the support does not well fit into the general model of EPICS: the recod types are hardware-specific. Could be useful to re-think this.

System is now very reliable: after February* of this year only one reboot (during a shutdown) for firmware upgrade, otherwise continuous 24x7 operation

Number of applications is large and still growing

Synchronization requirements for beamline applications are emerging; extensions are in progress

(*in February, some software fixes were done and an error in RF input cabling was corrected.)