

Timing system training for ICS integrators

Javier Cereiño Garcia

Integrated Control System Division
ESS, Sweden

<https://www.europeanspallationsource.se>
August 21, 2018

- ▶ Timing introduction
- ▶ Data buffer
- ▶ Timestamping
- ▶ Supercycle
- ▶ Delay compensation
- ▶ SAT and SIT
- ▶ Deployment

Principles and requirements

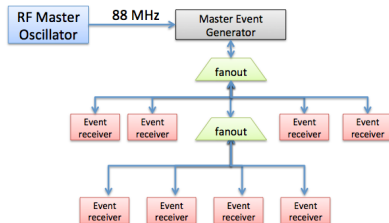
- ▶ Synchronize the operation of the facility
 - ▶ Accelerator, target, neutron instruments, CF (small extent)
- ▶ Provide the base frequency of 14 Hz
 - ▶ Also lower (beam) repetition rates
- ▶ Provide timestamping to the facility
 - ▶ Data correlation, EPICS archiving
- ▶ Provide trigger and clock signals
 - ▶ With a defined position in an operation sequence
- ▶ Support post-mortem analysis
 - ▶ What was the cause of a beam trip?

Implementation principles

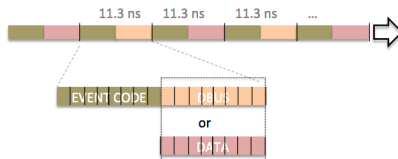
- ▶ A single master unit controls the timing distribution
 - ▶ Control from one central point
- ▶ Client units (receivers) act on masters orders
 - ▶ No dependency on e.g. computer network traffic
 - ▶ Actions pre-programmed at the receivers
- ▶ Synchronization to RF
 - ▶ All timing clocks phase-synchronous to RF (subharmonic)
- ▶ Distributes "wall-clock" time from the master
- ▶ Timestamp support in hardware
 - ▶ Used for data reconstruction and correlation

Structure of the system

- ▶ Master Event Generator (EVG) generates a continuous data stream
- ▶ The (accelerator) synchronization frequency (of 88 MHz) will be fed into the master event generator and used as the system clock. Event receivers synchronize to that clock.
- ▶ The 14 Hz operating frequency will be generated by down conversion from RF in the event generator (divide 88.0525 MHz by 6289464 to get 14.00000064 Hz)
- ▶ All timing sequences, events, etc., generated in hardware
- ▶ Links work upstream, too



On-the-wire protocol

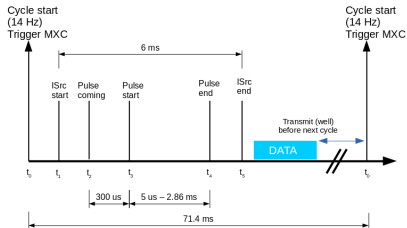


- ▶ 16-bit "frame" repeating at 88.0525 MHz (11.3 ns), i.e. 352.2 MHz/4
- ▶ Each frame contains an 8-bit event code and alternating distributed bus/data byte (8 bits).
- ▶ Distributed bus (DBus) broadcasts eight binary signals which can be output from each Event Receiver. Clocks, status bits, etc.
- ▶ Event Generator can broadcast arbitrary data (up to 2kB blocks)
- ▶ Data buffer sent one byte at a time and collected at the receiver(s)
- ▶ Trigger events and DBUS/Data are independent of each other

Events, clocks and data

- ▶ Meaning of "event" in this context: signal at a precise time
 - ▶ Hardware outputs with programmable delay, width, polarity
 - ▶ Event can also trigger software (EPICS) to do actions
 - ▶ "Collect post-mortem data now", "start counting", etc.
 - ▶ Events can be (synchronously) time-stamped
 - ▶ "Upstream" events: from EVR to EVG (and back)
- ▶ Clock: a repetitive sequence of pulses
 - ▶ E.g., a 1 kHz synchronization clock
- ▶ Data transmission
 - ▶ Limited amount of data delivered synchronously
 - ▶ For configuration and validation purposes
 - ▶ Still under development (some space for requests)

Accelerator cycle



- ▶ Pre-programmed sequences repeat at 14 Hz
 - ▶ Hardware-based (FPGA) sequencer with a RF-synchronous 88 MHz clock
 - ▶ "Programming" can happen in less than a millisecond
- ▶ "Data", i.e., beam mode, etc., transmitted in parallel
 - ▶ Contains "plan" for the next cycle: envelope mode, beam/no beam, pulse counter, rep rate, ...
 - ▶ Mode changes require a handshake with BIS
 - ▶ When mode change is transmitted, timing does not send beam before MPS/BIS gives OK

Timing adjustment policy

- ▶ Local delay settings are used for local fine-tuning; adjusting for propagation, processing, etc., delays during commissioning
- ▶ Pulse length, etc. information in beam data buffer is not for setting the pulse timing locally but for:
 - ▶ Pulse verification, feed-back/forward, etc.
- ▶ Beam off will be a dedicated event. A gate signal will be set on with a beam on event and set off with a beam off event
 - ▶ Central handling of the beam duration, no software activity needed at the (hundreds of) local receivers (to set the delays)
 - ▶ Actions can be tied to the beam off event, like beam-sync data collection and broadcast
 - ▶ Pulse may be cut short by the timing system. Separate event will allow clean handling of the early pulse abort

Documentation

- ▶ ESS Timing System System Architecture Description - ESS-0088633
- ▶ Data Model Specification for the ESS Timing System - ESS-0225673
- ▶ Event System with Delay Compensation Technical Reference - <http://mrf.fi/>
- ▶ EVR Usage Guide - <http://epics.sourceforge.net/mrfioc2/evr-usage.pdf> or <https://github.com/epics-modules/mrfioc2/blob/master/documentation/evr-usage.lyx> (lyx version)
- ▶ Manuals in CHESS

Data buffer

<https://confluence.esss.lu.se/display/HAR/Data+Buffer> <https://confluence.esss.lu.se/display/HAR/Event+and+Trigger+Reference>

- ▶ Protocol number
- ▶ Protocol version
- ▶ Linac cycle counter/ID Heartbeat
- ▶ Proton beam status
- ▶ Proton beam destination (last point)
- ▶ Proton beam mode
- ▶ Proton beam present
- ▶ Proton beam length
- ▶ Proton beam energy
- ▶ Proton beam current
- ▶ Raster pattern
- ▶ Target segment

https://github.com/icshwi/esstex/tree/master/document/ics_eng_docs_timestamping

```
6690f8a.icslab-ipc01.14335 > generalTimeReport  
Backwards time errors prevented 0 times.
```

Current Time Providers:

"EVR", priority = 50

Current Time is 2018-08-13 11:18:40.234652.

"OS Clock", priority = 999

Current Time is 2018-08-13 11:20:05.683007.

Event Time Providers:

"EVR", priority = 50

Supercycle

- ▶ Supercycle: a predefined sequence of machine cycles:
 - ▶ Timing pattern
 - ▶ Beam parameter (data)
- ▶ Cycles are pre-configured (with a control-room application)
- ▶ E.g., a supercycle for:
 - ▶ 1 Hz operation must have 1 cycle with beam, 13 cycles without beam
 - ▶ 0.1 Hz: 1 cycle with beam, 139 cycles without beam
- ▶ Editor application allows to define
 - ▶ pulse length (timing events in HW sequencer)
 - ▶ beam current (iris change, probably not between cycles)
 - ▶ pulse delay (within 2.86 time window, in sequencer)
 - ▶ beam destination (in data buffer)
 - ▶ beam mode (in data buffer)
 - ▶ etc.

Example

- ▶ Example from <https://gitlab.esss.lu.se/icshwi/reftabs/-/blob/master/supercycles/example-noCalib-2Hz-14c.csv>.

```
Id , IonMagSt , IonMagEnd , RFSt , LEBTCSt , LEBTCEnd , MEBTCSt , MEBTCEnd , BPulseSt , BPulseEnd ,  
  BLen , BEn , BCurr  
0 , 10000 , 10050 , 10010 , 10100 , 10150 , 10200 , 10250 , 13300 , 13350 , 50 , 100 , 100000  
1 , , , 10011 , 10100 , 10151 , 10200 , 10251 , 13300 , 13350 , 51 , 101 , 100001  
2 , , , 10012 , 10100 , 10152 , 10200 , 10252 , 13300 , 13350 , 52 , 102 , 100002  
3 , , , 10013 , 10100 , 10153 , 10200 , 10253 , 13300 , 13350 , 53 , 103 , 100003  
4 , , , 10014 , 10100 , 10154 , 10200 , 10254 , 13300 , 13350 , 54 , 104 , 100004  
5 , , , 10015 , 10100 , 10155 , 10200 , 10255 , 13300 , 13350 , 55 , 105 , 100005  
6 , , , 10016 , 10100 , 10156 , 10200 , 10256 , 13300 , 13350 , 56 , 106 , 100006  
7 , 10000 , 10060 , 10017 , 10100 , 10157 , 10200 , 10257 , 13300 , 13350 , 57 , 107 , 100007  
8 , , , 10018 , 10100 , 10158 , 10200 , 10258 , 13300 , 13350 , 58 , 108 , 100008  
9 , , , 10019 , 10100 , 10159 , 10200 , 10259 , 13300 , 13350 , 59 , 109 , 100009  
10 , , , 10020 , 10100 , 10160 , 10200 , 10260 , 13300 , 13350 , 60 , 110 , 100010  
11 , , , 10021 , 10100 , 10161 , 10200 , 10261 , 13300 , 13350 , 61 , 111 , 100011  
12 , , , 10022 , 10100 , 10162 , 10200 , 10262 , 13300 , 13350 , 62 , 112 , 100012  
13 , , , 10023 , 10100 , 10163 , 10200 , 10263 , 13300 , 13350 , 63 , 113 , 100013
```

DC

Delay compensation is achieved in measuring the propagation delay of events from the delay compensation master EVM through the distribution network up to the Event Receivers. At the last stage the EVR is aware of the delay through the network and adjusts an internal FIFO depth to match a programmed target delay value.

SAT and SIT

- ▶ When EVR is installed in FEB/G02: SAT and SIT.
- ▶ Jerzy does it.
- ▶ <https://confluence.esss.lu.se/display/HAR/Acceptance>
- ▶ SAT: check that the HW is OK, MCH correctly configured, fibres in place, etc.
- ▶ SIT: check that the EVR is working fine, receives events from the EVG, correctly creates trigger and gate signals, etc.

Deployment

```
require "autosave" "5.10.2"  
require "mrfioc2" "2.3.1-beta.3.3"  
epicsEnvSet "PEVR" "SYS-SUB:Ctrl-EVR-101"  
epicsEnvSet "PCI_SLOT" "08:00.0"  
epicsEnvSet "OUTARGS" "WIDTH0=1000, EVT0T0=14, BP0SRC=0, WIDTH1=1000, EVT1T0=15,  
    BP1SRC=1, WIDTH2=10, EVT2T0=12, WIDTH3=10, EVT3T0=13, BP2SRC=49, WIDTH4  
    =1000, EVT4T0=125, BP3SRC=4, WIDTH5=1000, EVT5T0=40, BP4SRC=5, WIDTH7  
    =1000, EVT7T0=42, BP6SRC=7,"  
epicsEnvSet "INARGS" "BPIN5=, BP5EVT=41, BPIN7=, BP7EVT=43,"
```

Questions?

```
### Load the mrfioc2 module ###
require mrfioc2,2.2.0-rc2

### Define several needed macros ###
epicsEnvSet("IOC", "TRAINING")
epicsEnvSet("DEV1", "EVR1")
epicsEnvSet("MainEvtCODE" "14")
epicsEnvSet("HeartBeatEvtCODE" "122")
epicsEnvSet("ESSEvtClockRate" "88.0525")

### Register the EVR with the IOC and load the database ###
mrmEvrSetupPCI("${DEV1}", "0a:00.0")
dbLoadRecords("evr-mtca-300u-ess.db","EVR=${DEV1}, SYS=${IOC}, D=${DEV1}, FEVT=
(ESSEvtClockRate)")

### Needed with software timestamp source w/o RT thread scheduling ###
var evrMrmTimeNSOverflowThreshold 100000

iocInit()

### Set delay compensation to 70 ns, needed to avoid timesptamp issue ###
dbpf $(IOC)-$(DEV1):DC-Tgt-SP 70
```

```
[iocuser@icslab-ts03 cmds]$ iocsh.bash TRAINING-EVR1.cmd
```

```
caget TRAINING-EVR1:Link-Sts
caget TRAINING-EVR1:Link-Clk-I
caget TRAINING-EVR1:Cnt-LinkTimo-I
caput TRAINING-EVR1:Link-Clk-SP 100
caget TRAINING-EVR1:Link-Clk-I
caget TRAINING-EVR1:Link-Sts
caget TRAINING-EVR1:Cnt-LinkTimo-I
caget TRAINING-EVR1:Cnt-LinkTimo-I
caput TRAINING-EVR1:Link-Clk-SP 88.0525
caget TRAINING-EVR1:Link-Clk-I
caget TRAINING-EVR1:Link-Sts

caget TRAINING-EVR1:Time-Valid-Sts

caget TRAINING-EVR1:FwVer-I
```

```
camonitor TRAINING-EVR1:EvtECnt-I
```

```
caput TRAINING-EVR1:DlyGen0-Evt-Trig0-SP 14
```

```
caput TRAINING-EVR1:DlyGen0-Width-SP 1000
```

```
caput TRAINING-EVR1:OutFP0-Src-SP 0
```

```
/epics/base-3.15.5/require/3.0.0/siteMods/mrfioc2/2.2.0-rc2/db/evr-mtca-300u-ess  
.db
```

```
TRAINING-EVR1:OutFP0-Src-Pulse-SP
```

```
TRAINING-EVR1:OutFP0-Src-DBus-SP
```

```
TRAINING-EVR1:OutFP0-Src-Scale-SP
```

https://github.com/icshwi/esstex/tree/master/document/ics_eng_docs_timestamping

```
6690f8a.icslab-ipc01.14335 > generalTimeReport  
Backwards time errors prevented 1464 times.
```

Current Time Providers:

"EVR", priority = 50

Current Time is 2018-08-13 11:18:40.234652.

"OS Clock", priority = 999

Current Time is 2018-08-13 11:20:05.683007.

Event Time Providers:

"EVR", priority = 50

```
dbLoadRecords("./counter.db")
```

```
camonitor counter
```

```
caput TRAINING-EVR1:Time-I.EVNT 14
```

```
caput TRAINING-EVR1:Time-I.INP "@OBJ=EVR1, Code=14"
```

```
caput counter.TSEL TRAINING-EVR1:Time-I.TIME
```

EVR: standalone mode (I...)

```
### Load the mrfioc2 module ###
require mrfioc2,2.2.0-rc2

### Define several needed macros ###
epicsEnvSet("IOC", "TRAINING")
epicsEnvSet("DEV1", "EVR1")
epicsEnvSet("MainEvtCODE" "14")
epicsEnvSet("HeartBeatEvtCODE" "122")
epicsEnvSet("ESSEvtClockRate" "88.0525")

### Register the EVR with the IOC and load the database ###
mrmEvrSetupPCI("${DEV1}", "0a:00.0")
dbLoadRecords("evr-mtca-300u-ess.db","EVR=${DEV1}, SYS=${IOC}, D=${DEV1}, FEVT=$
(ESSEvtClockRate)")

### Needed with software timestamp source w/o RT thread scheduling ###
var evrMrmTimeNSOverflowThreshold 100000
```

```
ioclnit()
```

```
#### Set delay compensation to 70 ns, needed to avoid timestamp issue ####  
dbpf $(IOC)-$(DEV1):DC-Tgt-SP 70
```

```
#### Get current time from system clock ####  
dbpf $(IOC)-$(DEV1):TimeSrc-Sel "Sys. Clock"
```

```
#### Set up the prescaler that will trigger the sequencer at 14 Hz ####  
dbpf $(IOC)-$(DEV1):PS0-Div-SP 6289424 # in standalone mode because real freq  
    from synthsiezer is 88.05194802 MHz, otherwise 6289464
```

```
#### Set up the sequencer ####  
dbpf $(IOC)-$(DEV1):SoftSeq0-RunMode-Sel "Normal"  
dbpf $(IOC)-$(DEV1):SoftSeq0-TrigSrc-2-Sel "Prescaler 0"  
dbpf $(IOC)-$(DEV1):SoftSeq0-TsResolution-Sel "uSec"  
dbpf $(IOC)-$(DEV1):SoftSeq0-Load-Cmd 1  
dbpf $(IOC)-$(DEV1):SoftSeq0-Enable-Cmd 1
```

```
#### Run the script that configures the events and timestamp of the sequence ####  
system("/bin/sh ./configure_sequencer_14Hz.sh $(IOC) $(DEV1)")
```



```
configure_sequencer_14Hz.sh:  
# Bash script to configure the EVG/EVR sequencer  
# All values in us  
  
# Event code 14 (14 Hz), 127 is the end of sequence  
caput -a $1-$2:SoftSeq0-EvtCode-SP 2 14 127  
  
# Defining time at which the event codes are sent in us  
caput -a $1-$2:SoftSeq0-Timestamp-SP 2 0 1  
  
# Commit the sequence to HW  
caput $1-$2:SoftSeq0-Commit-Cmd 1
```

```
# Set up FPIN0. Generate Code 10 locally on rising edge.  
#dbpf $(IOC)-$(DEV1):OutRB00-Src-SP 61 # PCIe-EVR-300DC  
dbpf $(IOC)-$(DEV1):In0-Trig-Ext-Sel "Edge"  
dbpf $(IOC)-$(DEV1):In0-Code-Ext-SP 10
```

```
TRAINING-EVR1:In0-Lvl-Sel  
TRAINING-EVR1:In0-Edge-Sel  
TRAINING-EVR1:In0-Trig-Ext-Sel  
TRAINING-EVR1:In0-Code-Ext-SP
```

https://github.com/icshwi/esstex/tree/master/document/ics_eng_docs_mtca_backplane_clocks

```
# Set TCLKB to low, enable it and power it up
dbpf $(IOC)-$(DEV1):OutTCLKB-Src-SP 63
dbpf $(IOC)-$(DEV1):OutTCLKB-Ena-Sel 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pwr-Sel 1
# TCLKB is 40-bit pattern, set the starting 20 bits to 1 (and the rest to 0 -
  default)
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.BF 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.BE 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.BD 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.BC 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.BB 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.BA 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.B9 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.B8 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.B7 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.B6 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.B5 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.B4 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.B3 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.B2 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.B1 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low00_15-SP.B0 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low16_31-SP.BF 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low16_31-SP.BE 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low16_31-SP.BD 1
dbpf $(IOC)-$(DEV1):OutTCLKB-Pat-Low16_31-SP.BC 1
```

```
require mrfioc2 , 2.2.0-rc2

epicsEnvSet("IOC", "TRAINING")
epicsEnvSet("DEV1", "EVG1")

epicsEnvSet("ESSEvtClockRate" "88.0525")

mrmEvgSetupPCI($(DEV1), "16:0e.0")
dbLoadRecords("cpci-evg230-ess.db", "SYS=$(IOC), D=$(DEV1), EVG=$(DEV1), FEVT=$
    (ESSEvtClockRate), FRF=$(ESSEvtClockRate), FDIV=1")

iocInit()
```

```
# Event frequency:
caput TRAINING-EVG1:EvtClk-FracSynFreq-SP 100

# Software event:
caput TRAINING-EVG1:SoftEvt-EvtCode-SP 10

# Timestamping:
dbpf $(IOC)-$(DEV1):1 ppsInp-Sel "Sys Clk"
...
epicsThreadSleep 5
dbpf $(IOC)-$(DEV1):SyncTimestamp-Cmd 1

# RF input:
dbpf $(IOC)-$(DEV1):EvtClk-Source-Sel "RF"
dbpf $(IOC)-$(DEV1):EvtClk-RFFreq-SP 88.0525
dbpf $(IOC)-$(DEV1):EvtClk-RFDiv-SP 1

# Configure front panel for event 125 1 Hz
dbpf $(IOC)-$(DEV1):TrigEvt1-EvtCode-SP 125
dbpf $(IOC)-$(DEV1):TrigEvt1-TrigSrc-Sel "Univ0"
dbpf $(IOC)-$(DEV1):1 ppsInp-Sel "Univ0"
dbpf $(IOC)-$(DEV1):1 ppsInp-MbbiDir_.TPRO 1
```

```
# Heart Beat 1 Hz:
dbpf $(IOC)-$(DEV1):Mxc7-Prescaler-SP 88052500
dbpf $(IOC)-$(DEV1):TrigEvt7-EvtCode-SP $(HeartBeatEvtCODE)
dbpf $(IOC)-$(DEV1):TrigEvt7-TrigSrc-Sel "Mxc7"

# Setup of sequencer:
dbpf $(IOC)-$(DEV1):Mxc0-Prescaler-SP 6289464
dbpf $(IOC)-$(DEV1):SoftSeq0-RunMode-Sel 0 # normal mode
dbpf $(IOC)-$(DEV1):SoftSeq0-TrigSrc-Sel "Mxc0"
dbpf $(IOC)-$(DEV1):SoftSeq0-TsResolution-Sel 2 # us
dbpf $(IOC)-$(DEV1):SoftSeq0-Load-Cmd 1
dbpf $(IOC)-$(DEV1):SoftSeq0-Enable-Cmd 1
system("/bin/sh ./configure_sequence.sh $(IOC) $(DEV1)")
```

Questions?