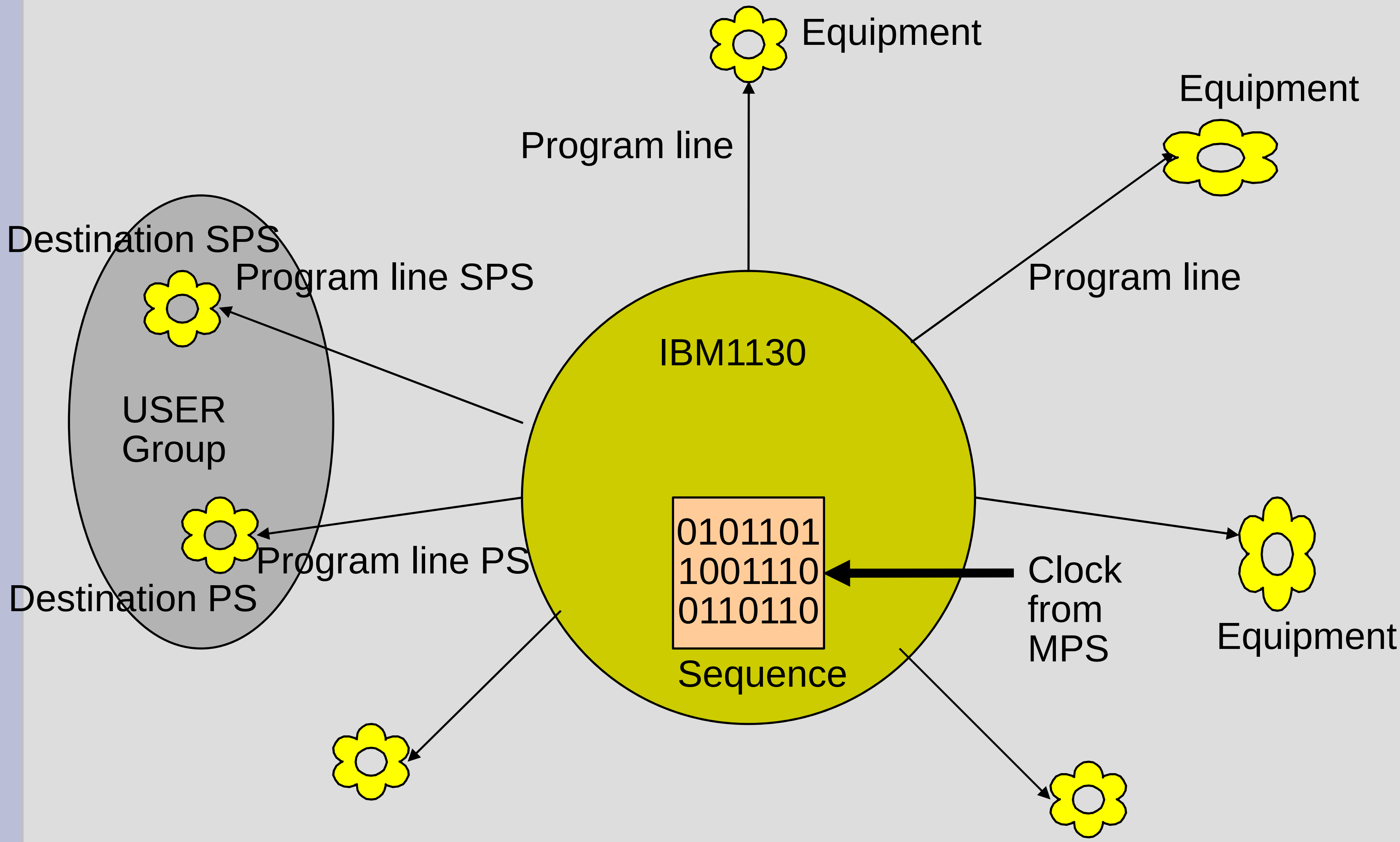
Timing Review

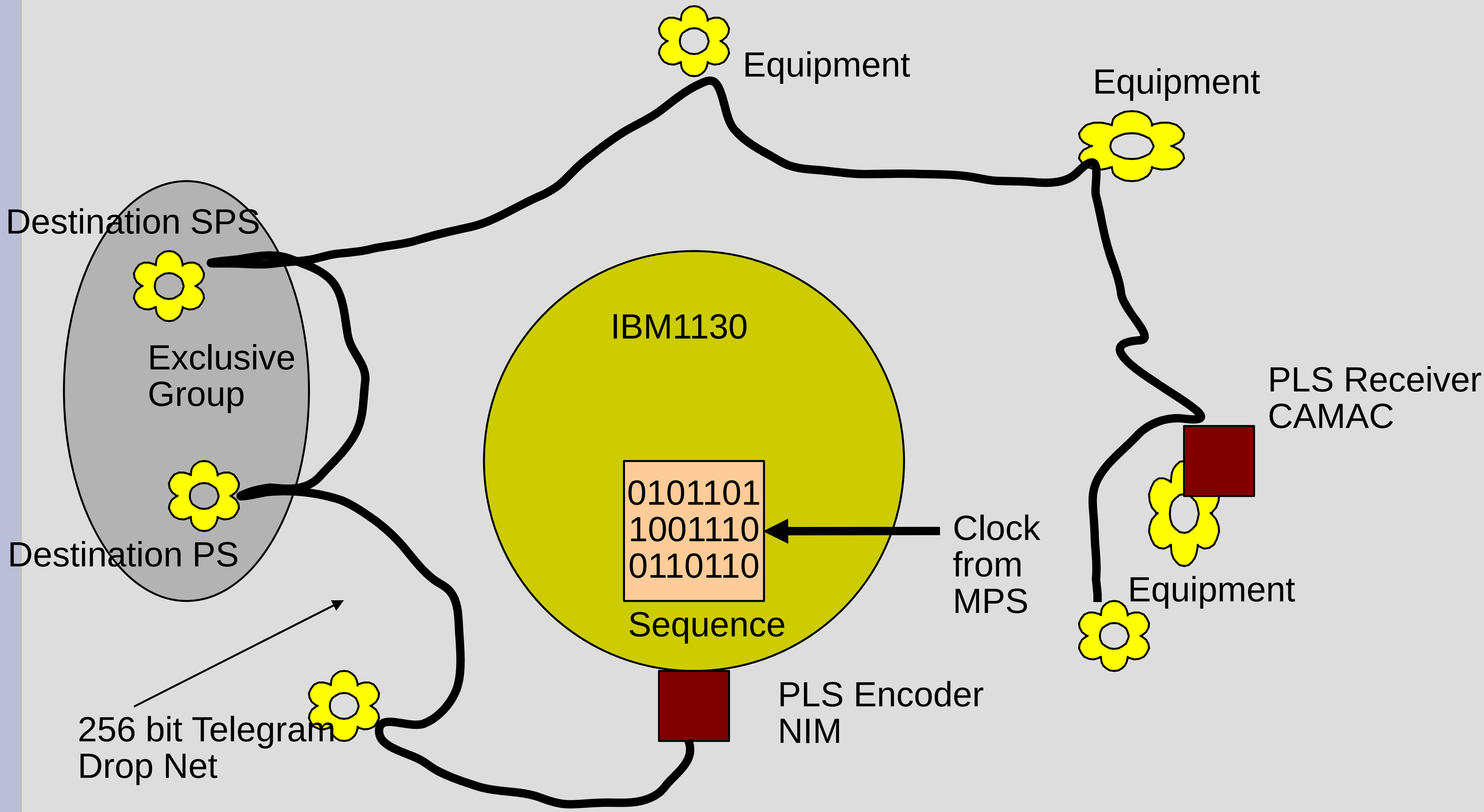
- Evolution of the central and distributed timing
- How we got where we are and why
- What are its strengths and weaknesses

A long time ago in a far away galaxy ...

The Program Line Sequencer 30 Years ago !!



The Program Line Sequencer 28 Years ago !!



How to archive cycle settings

- Back in those days we had a beam synthesizer
  - Telegram described the beam to be made
  - A beam synthesizer is difficult to archive because cycles share equipment settings.
    - Cycle A: when the destination is X the power is Y
    - Cycle B: when the destination is X the power is Z
- A Virtual-Accelerator (VA) is simpler, it's just set up to do one thing.
  - A snap shot in time of actual control system settings.
  - Big advantage of a VA is the ease with which it can be archived. (No coupling)

Archives, users, and VAs

- Because we needed to archive the settings for an accelerator cycle, we moved towards the VA idea
- An accelerator is time-sliced to manufacture beams, in each slice settings are instantiated on its equipment.
- We can think of each slice as a virtual accelerator.
- The settings in each slice are independent from each other.
  - So they can be archived independently.

User vs VA

- A USER is NOT equal to a VA.
  - For OP a USER is a cycle with run time variations.
    - E.g. with many possible destinations.
- A Cycle Instance is NOT equal to VA
  - The same VA can occur more than once in a super cycle.
- A VA IS a unique and complete set of control values for an accelerator to run a given cycle.

What's a USER anyway ?

- Instances of the same USER are different. Example SPS Fixed Target cycles could have an SPS or DUMP destination.
  - In a VA scheme this should have different IDs.
  - VA inherits from/is a subclass of/ USER.
- Multiplexing on USER leads to ..
  - Complications in CBCM for on the fly telegram calculation (FIDO)
  - Complications in timing to deal with on the fly telegrams that differ for the same USER (REGA)
  - Double and Triple PPM.

What's DTM ?

- Distributed Table Manager
  - Basically DTM is a software implementation of reflective memory (Via UDP).
  - This was needed to distribute timing configuration data (telegram description) to the front ends and to servers and work stations in a platform independent way.
  - DTM is very “real time” because the configuration data can change any time.
  - Later we used it to send telegrams out over the network to synchronize application programs.
  - This is the only job it is used for today.

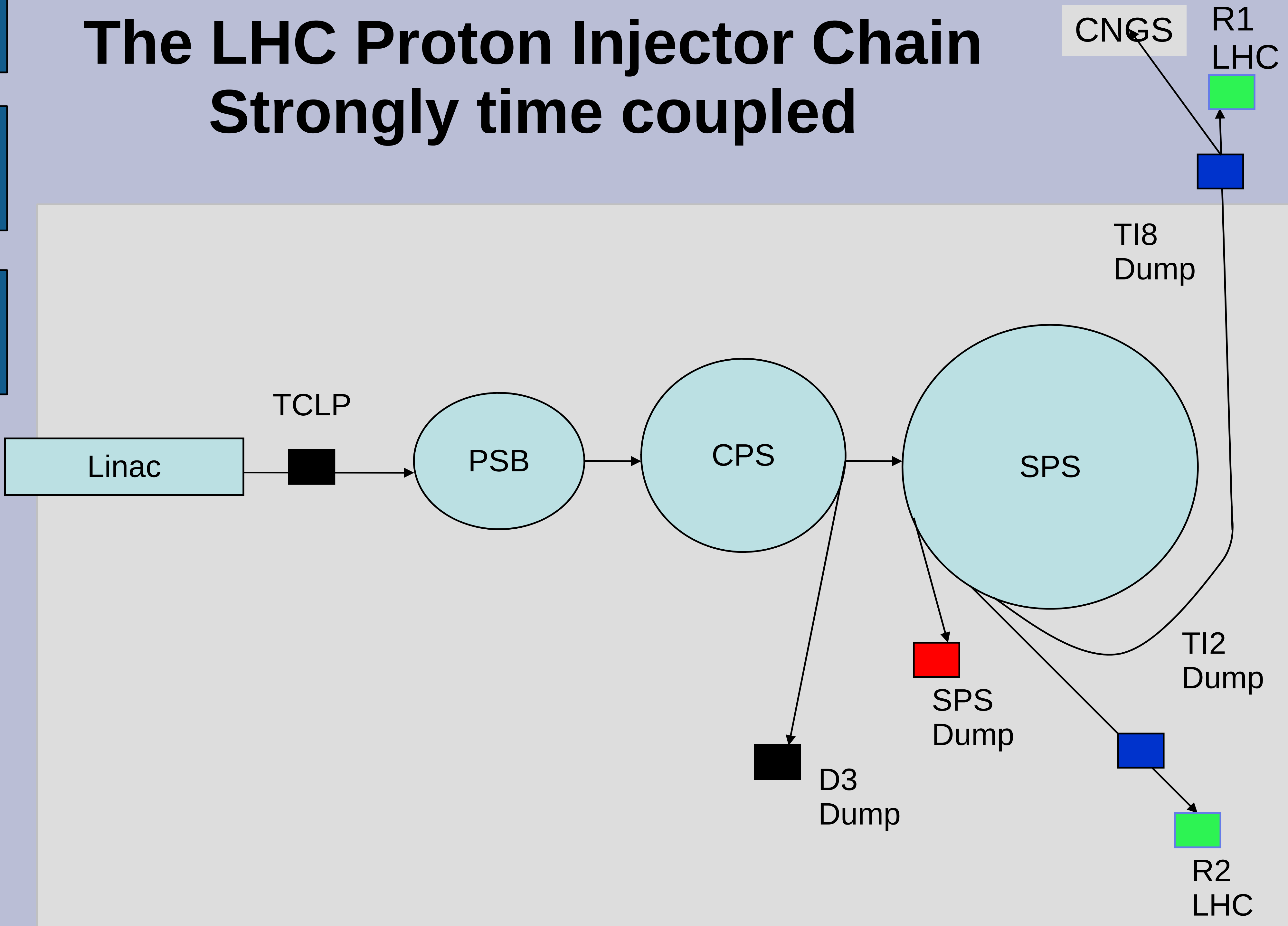
Arrival of the TG8

- The TG8 gave us the possibility of real event codes with payloads instead of just pulses.
- However we never exploited the event payloads and stayed with telegrams.
- This led to a lot of unnecessary complications such as dead zones, telegram handling libraries, and TG8 firmware ...
- We had a lot of legacy (7 accelerators), and I guess we just failed to notice, or feel the need to change.
- SPS was using payloads, so no dead zone, however used cycle instance not VA.

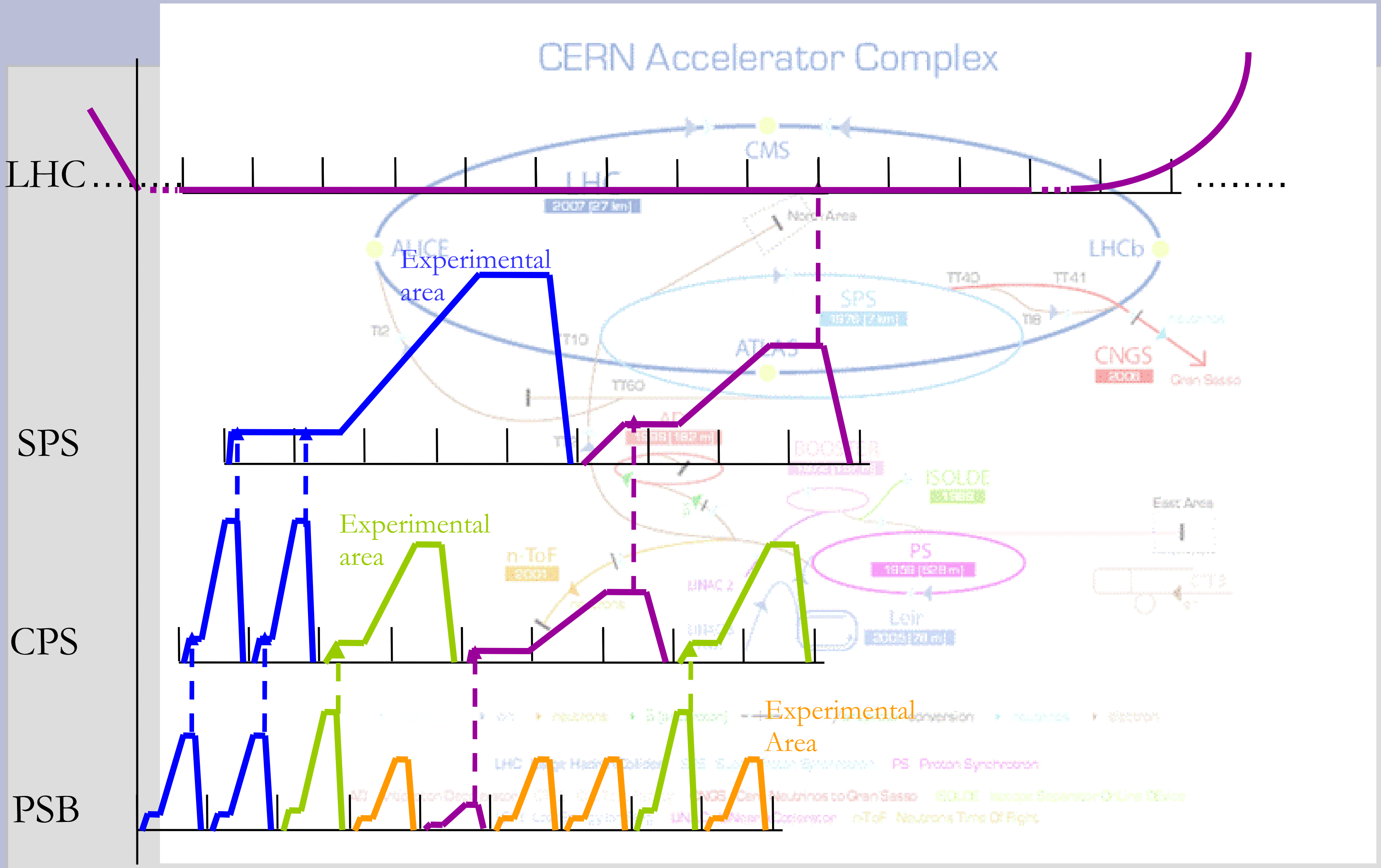
History

- When AB division was formed, the PS central timing was extended to include the SPS as a strongly coupled machine.
- Given the LHC start up date at that time, we had to implement rapid coordinated super cycle changes for LHC filling rapidly.
- This involved a lot of work to replace the Faraday cage timing, changing SPS event codes, and implementing an SPS telegram.
- Today the SPS is fully integrated into the CBCM and uses USER in the payloads to drive multiplexing.

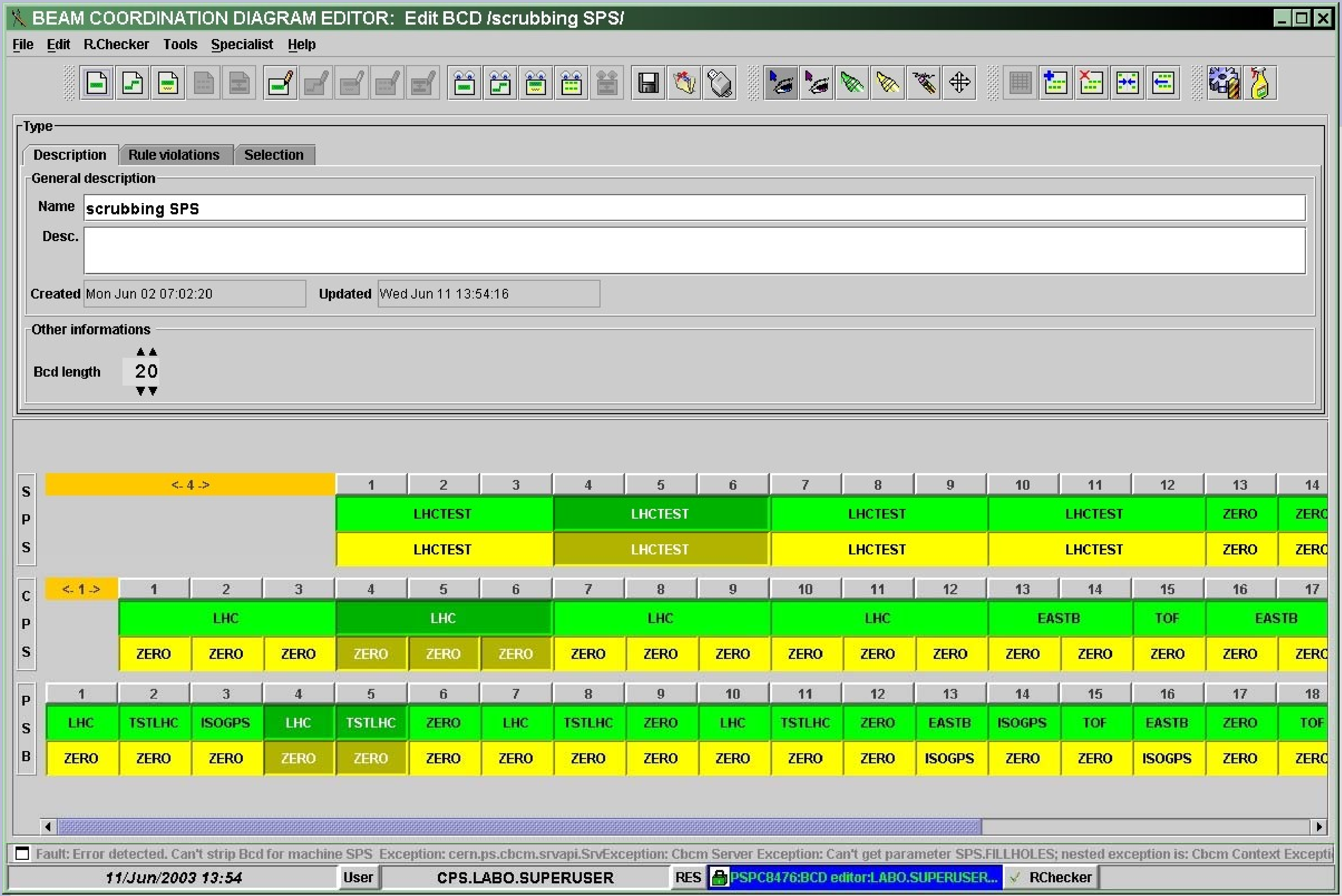
The LHC Proton Injector Chain
Strongly time coupled



CERN accelerator network sequenced by central timing generator



CBCM Sequence Manager



Name: scrubbing SPS

Desc.:

Created: Mon Jun 02 07:02:20

Updated: Wed Jun 11 13:54:16

Bcd length: 20

SPS	<- 4 ->	1	2	3	4	5	6	7	8	9	10	11	12	13	14
		LHCTEST			LHCTEST			LHCTEST			LHCTEST			ZERO	ZERO
		LHCTEST			LHCTEST			LHCTEST			LHCTEST			ZERO	ZERO

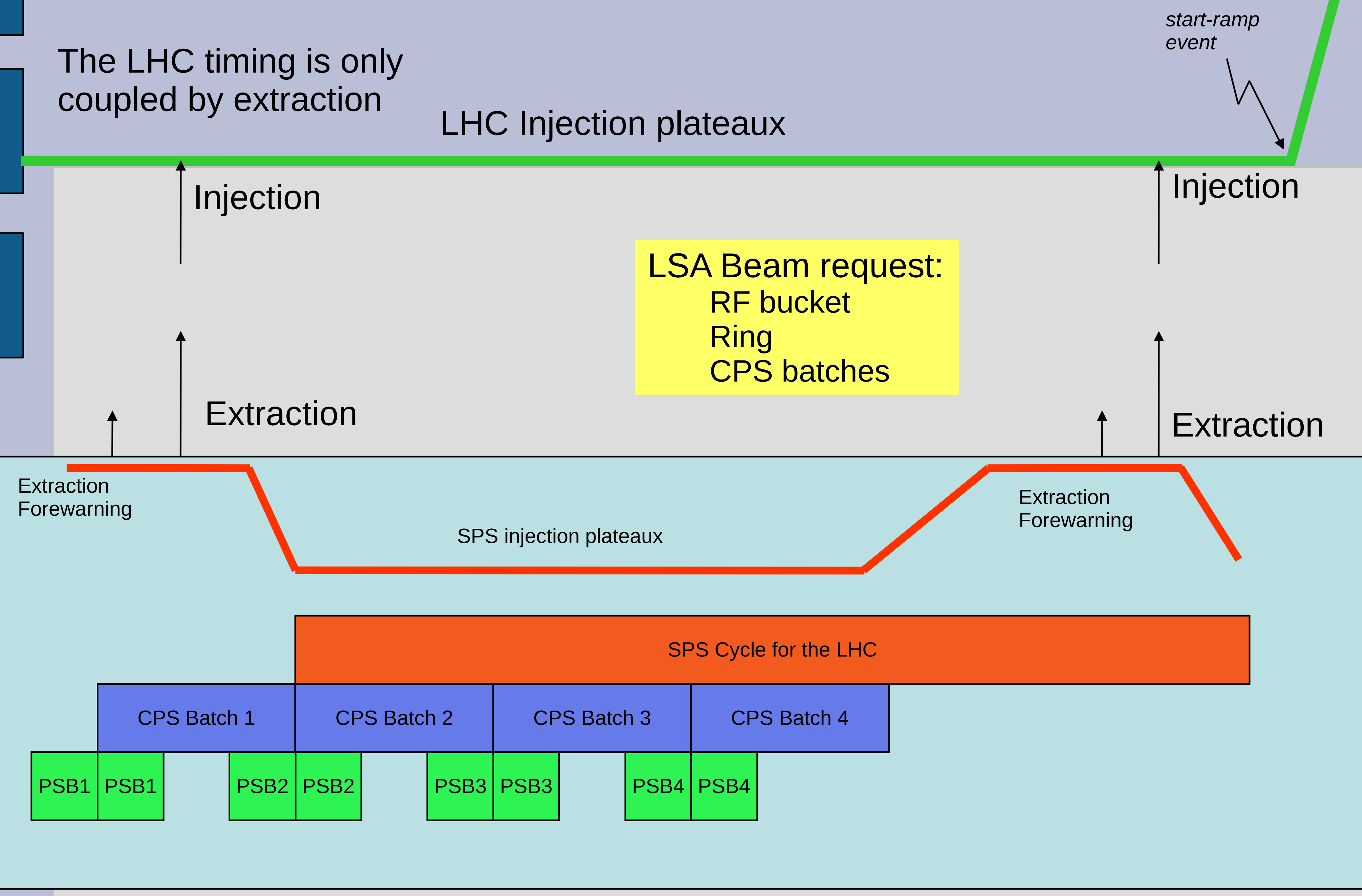
CPS	<- 1 ->	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
		LHC			LHC			LHC			LHC			EASTB		TOF	EASTB	
		ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO

PSB	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
	LHC	TSTLHC	ISOGPS	LHC	TSTLHC	ZERO	LHC	TSTLHC	ZERO	LHC	TSTLHC	ZERO	EASTB	ISOGPS	TOF	EASTB	ZERO	TOF
	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ZERO	ISOGPS	ZERO	ZERO	ISOGPS	ZERO	ZERO

```
Fault: Error detected. Can't strip Bcd for machine SPS  Exception: cern.ps.cbcm.srvapi.SrvException: Cbcm Server Exception: Can't get parameter SPS.FILLHOLES; nested exception is: Cbcm Context Excepti
```

11/Jun/2003 13:54 | User | CPS.LABO.SUPERUSER | RES | PSPC8476:BCD editor:LABO.SUPERUSER... | ✓ RChecker

The LHC Beam



The LHC timing is only coupled by extraction

LHC Injection plateaux

start-ramp event

Injection

Injection

LSA Beam request:
RF bucket
Ring
CPS batches

Extraction

Extraction

Extraction Forewarning

Extraction Forewarning

SPS injection plateaux

SPS Cycle for the LHC

CPS Batch 1

CPS Batch 2

CPS Batch 3

CPS Batch 4

PSB1 PSB1

PSB2 PSB2

PSB3 PSB3

PSB4 PSB4

So here we are today

- Is the central timing able to orchestrate the Injector chain to fill the LHC ?
  - Yes ! The LHC Fill Use case has been implemented.
- Is the LIC central timing open and easy to understand ?
  - NO ! Its closed and very complex, even the timing experts have problems.
- Can we export general timing system say, to another lab like GSI ?
  - NO ! Too many dependencies on CERN concepts (Especially telegrams)

Strengths and weaknesses

- Is timing reliable and efficient ?
  - YES and NO. Relies on experts. Uses redundant concepts. But it seems to work OK.
- Is it easy to maintain ?
  - NO ! It evolved over a very long time, and is not based on the new controls standards.
- Is it flexible ?
  - YES and NO. FIDO** is a user hostile language, but very flexible.
- Is it simple ?
  - NO ! It is far too complex, legacy

** FIDO is a programming language used to control the central timing

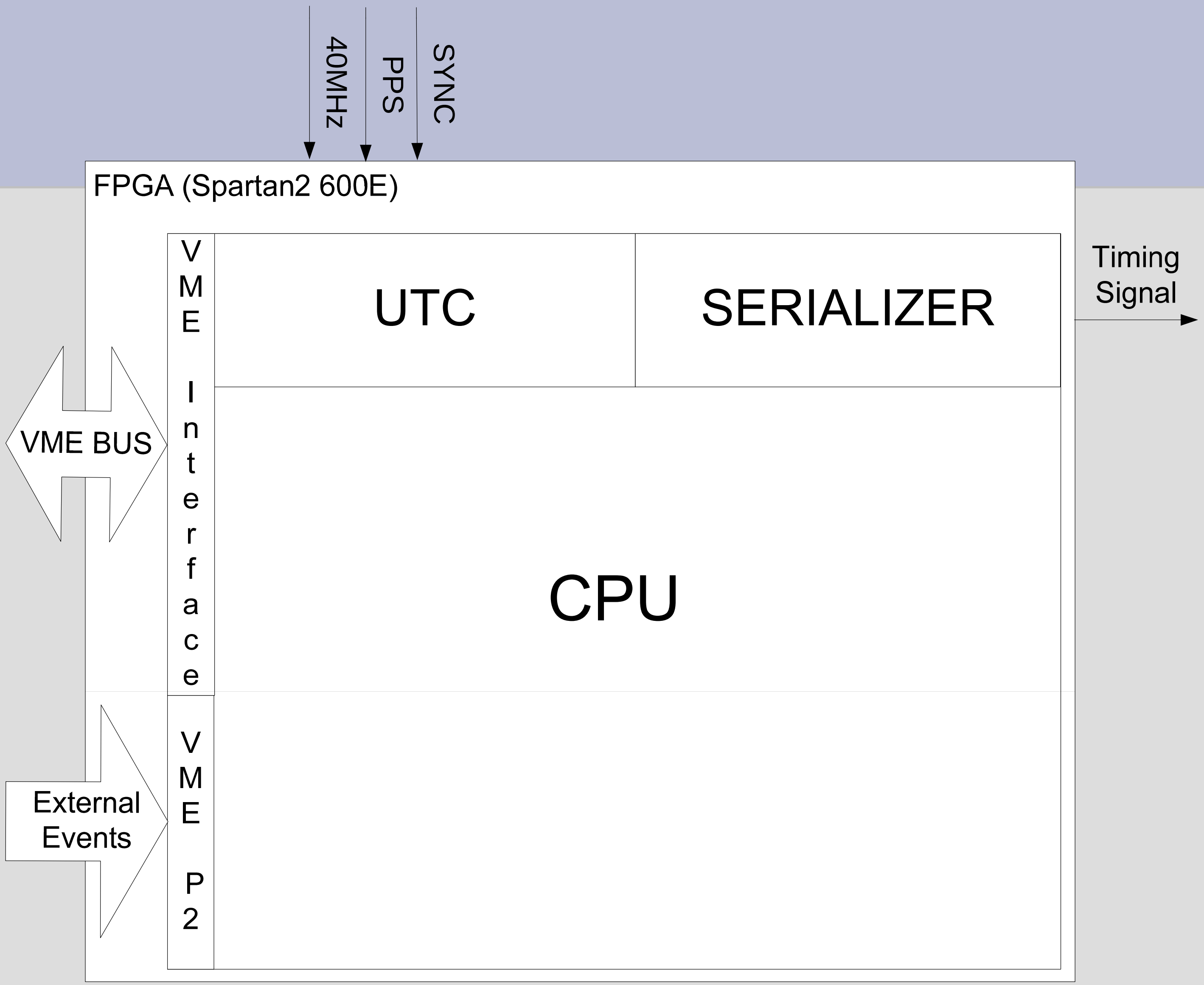
And then came the LHC machine timing requirements

- This required a complete rethink ...
- No basic-periods..
- No cycles..
- LSA..

Multitask Timing Generator MTT

- Implements 16 parallel virtual processors
- Each processor can be assigned a task to run from program memory.
- Program memory may contain many more tasks than available processors.
- Event table tasks synchronize with the millisecond clock and send out events.

MTT hardware module

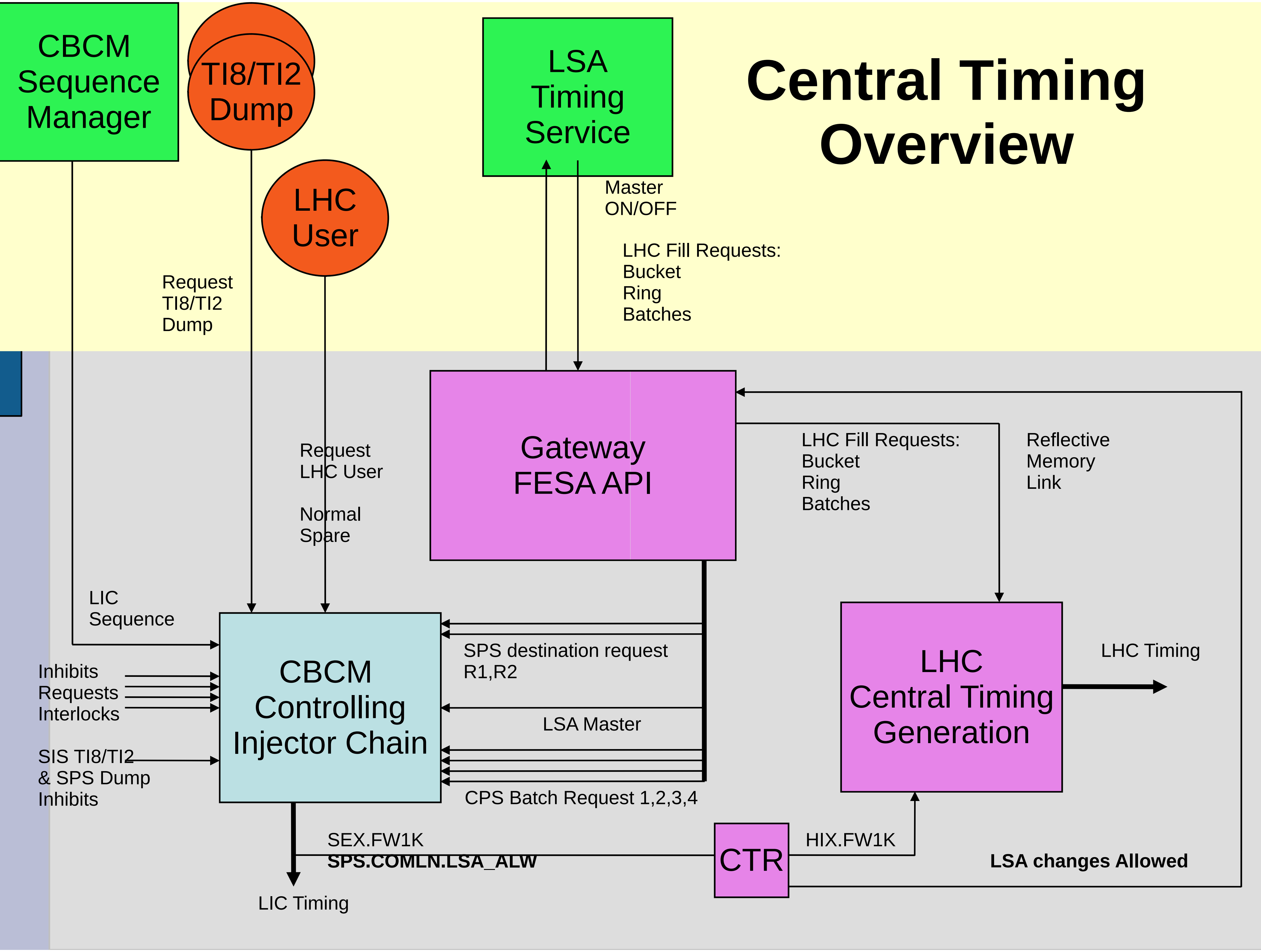


See: The LHC central timing hardware implementation
P. Alvarez, J. Lewis, J. Serrano CERN, Geneva, Switzerland
This conference

MTT External Events Task

```
strp:     % Start program and interrupt survey task
          movv    TskStsRUNNING      LRegTASK_STATUS                          % Say we are running
          movv    ConsNOT_SET        LRegParRUN_COUNT                         % Run forever
          int     2                                                           % Notify survey we are running

cont:     % Wait for the VME P2 bits and send out events accordingly
          worv    ConsVMEP2 BITS     VMEP2                                    % Wait for VME P2 bits
          movr    VMEP2              RegVmeP2                                 % Copy reg and clear bits

tdmp1:    % Test for dump ring 1
          andv    ConsHX_DMPD1_BIT   RegVmeP2           LRegTEMP              % Test dump 1 bit
          beq     tdmp2                                                       % Go check for dump 2 bit
          movv    ConsHX DMPD1       EVOUT                                    % Send dump 1
          movv    ConsHX ENBPM1      EVOUT                                    % Re-enable after a dump

tdmp2:    % Test for dump ring 2
          andv    ConsHX_DMPD2_BIT   RegVmeP2           LRegTEMP              % Test dump 2 bit
          beq     tinj                                                        % Go injection warning bit
          movv    ConsHX DMPD2       EVOUT                                    % Send dump 2
          movv    ConsHX ENBPM2      EVOUT                                    % Re-enable after a dump

tinj:     % Test for LHC injection
          andv    ConsHIX_FW_BIT    RegVmeP2            LRegTEMP              % Test inject bit
          beq     tpm1                                                        % Go check for PM ring 1
          movv    ConsHIX_FW        EVOUT                                     % Send injection forewarning

tpm1:     % Test for post mortem bit 1
          andv    ConsHX_PM1_BIT    RegVmeP2            LRegTEMP              % Test PM ring 1 bit
          beq     tpm2                                                        % Go check for PM ring 2
          movv    ConsHX PM1        EVOUT                                     % Send PM-1 trigger

tpm2:     % Test for post mortem bit 2
          andv    ConsHX_PM2_BIT    RegVmeP2            LRegTEMP              % Test PM ring 2 bit
          beq     cont                                                        % Go check for PM ring 2
          movv    ConsHX PM1        EVOUT                                     % Send PM-1 trigger (not PM-2)

          jmp     cont                                                        % Go wait for next P2 interrupt
```

Central Timing Overview



CBCM Sequence Manager

TI8/TI2 Dump

LHC User

LSA Timing Service

Master ON/OFF

LHC Fill Requests:
Bucket
Ring
Batches

Request TI8/TI2 Dump

Gateway FESA API

Request LHC User

Normal
Spare

LHC Fill Requests:
Bucket
Ring
Batches

Reflective Memory Link

LIC Sequence

CBCM Controlling Injector Chain

SPS destination request R1,R2

LSA Master

CPS Batch Request 1,2,3,4

Inhibits
Requests
Interlocks

SIS TI8/TI2
& SPS Dump
Inhibits

LHC Central Timing Generation

LHC Timing

SEX.FW1K
SPS.COMLN.LSA_ALW

CTR

HIX.FW1K

LSA changes Allowed

LIC Timing

Where are we with the LHC ?

- On schedule, timing works
- Can we drive the LHC ?
  - YES ! The performance during the dry runs was correct.

LHC Timing

- Is it simple, exportable, open, flexible and minimal.
  - YES !
- Does it follow controls standards ?
  - YES !
- Is it easy to maintain ?
  - YES !
- Strange but in the LHC telegrams perform a useful function. (Snap shot). They are NOT used to control the machine.

What are the users complaining about ?

- Controls people
  - It's too complicated
  - Difficult to diagnose
  - Non standard
- Operations
  - Seem to be mostly happy
  - Would like Fast economy mode
  - To break strong coupling during MDs
  - More control over the central timing events

Controls complaints

- We use our own middleware DTM
  - It was running before Java was even invented !
- We use our own data base accessed via RMI
  - We used to use Oracle dynamic SQL Pro-C from an X-motif C application. Every year we had to rewrite half of it to keep up with the Oracle updates. In those days the data base was slow and unreliable.
- In other words the concepts and code are old and out of date.

What's the problem for HT ?

- Does the ADE belong with the injector chain?
- We can't duplicate each other.
- We need help from other sections, we are too stretched to do it all on our own.
- Two differing approaches to LHC and LIC timing.
- We are maintaining a lot of stuff that could be dropped.

What we should do ?

- Use controls standards, hence exploit the work of other sections and get more people working on areas that we should not be responsible for. (DTM, Data-Base....)
- Basically learn from what we did in the LHC (Build one system not two).
- Use the injector chain renovation project to tidy up the mess.
- Take into account user requirements ...