

# Cheby – Memmap description

FDF 2025

**Tristan Gingold, Bartosz Bielawski**

**CERN**

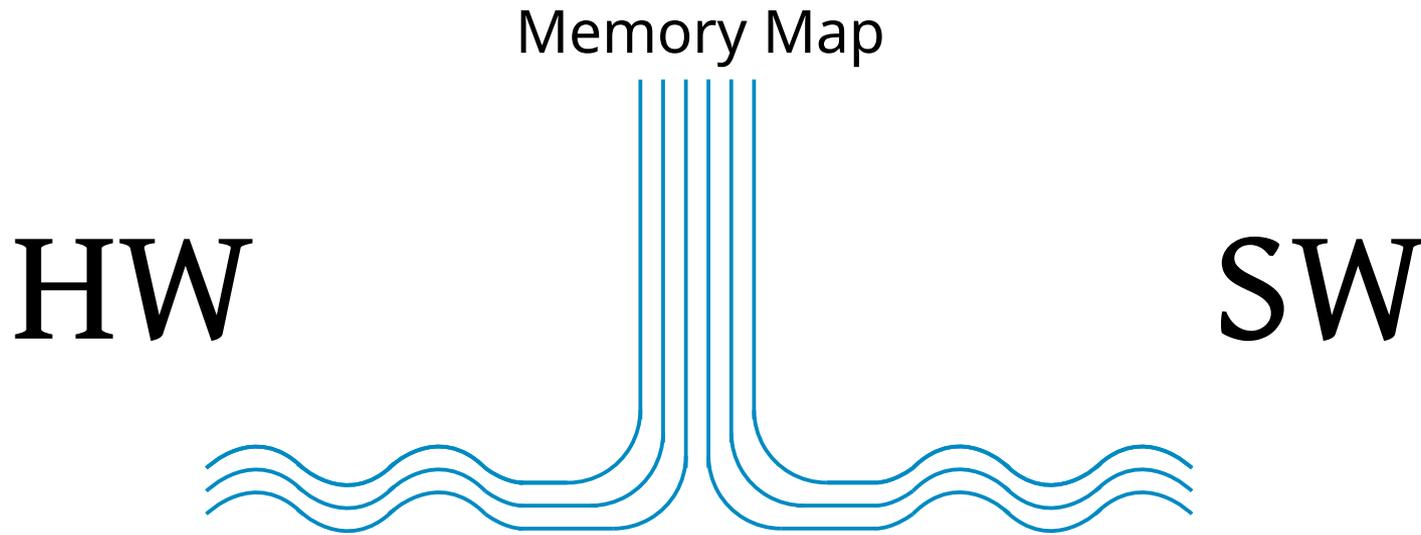
2025-05-22

# What is Cheby?

Cheby generates HDL and software files from a description of a memory map.

HDL:	VHDL, Verilog, SystemVerilog
Software:	C headers, python files
HDL testbenches:	constant files
Doc:	HTML, markdown
(and more)	

# Why a memory map generator?



The memory map is the interface  
The views must be consistent  
Writing this code is tedious

# Why Cheby?

There are other memory map generators.  
There are even many standards for that.

But:

- We heavily use WishBone bus
- We need to support legacy files
- We need CERN specific outputs for some tools

# Main Challenges

Features

Performance

Genericity

- Register behaviours

Extensibility

- User specific generator?

# Strong points (1)

## Diversity:

- Registers (one or more words)
- Fields: bit or bit range in a register
- Memories (single or dual ports)
- Blocks: group of anything
- Repetition: repeat a group of registers
- Sub map
  - instantiation of another memory map

# Strong points (2)

## Buses:

- AXI 4 Lite
- Wishbone
- Avalon
- Simple (strobe+ack)

Width

Endianness

# Strong points (3)

## Genericity

- Access to raw signals (strobe and ack)
- You can implement any feature
- Trade-off in provided features

## Extensible (x-\* attributes...)

- Users can add their own attributes
- Ignored by the generator

# Strong points (4)

## Performance

- Address decoder is carefully generated
- Fine control of pipelining
- Large or deep maps supported

Lots of tests (also useful as examples)

User documentation (in markdown)

# Strong points (5)

Open-Source: <https://cern.ch/cheby-repo>

Widely used at CERN and outside

- With contributions

Python 3.x

# Weak points (1/2)

## No using a standard format

- But systemRDL specification is 100+ pages...
- Extensions ?
- (But still YAML based)

## Yet another tool ?

- We had to deal with our legacy

# Weak points (2/2)

YAML + pyYAML make errors report difficult

- No access to line+column in the YAML file

Not for high performance interconnect

- Do not use the memories for main processor memories
- No cross-bars, no simultaneous accesses, no burst
- For peripherals

**Reksio**

# Reksio / GUI

## Create your memory map with a graphical tool

- See possible tags and attributes
- Verify correctness of the map
- Run other tools directly
  
- Precompiled binaries available
- Open source

The screenshot displays the Reksio v2.1.2 GUI interface. The main window shows a 'Nodes tree' on the left, listing various components like 'memory-map', 'x-driver-edge', 'x-enums', 'submap', and several 'reg' and 'repeat' entries. The right pane shows the 'Attributes' for the selected 'I4TunerControlVME' node, including fields for 'bus', 'name', 'description', 'size', 'equipment-code', 'module-type', 'name', 'bus-type', 'endianness', 'dma-mode', 'driver-version', and 'driver-version-suffix'. Below this, there are two tables: 'x-enums/enum' and 'x-driver-edge/constant-values/constant-value'. The 'x-enums/enum' table has columns for 'name', 'width', and 'description', with one entry 'bpTriggers' having a width of 4. The 'x-driver-edge/constant-values/constant-value' table has columns for 'name', 'value', 'type', and 'description', listing various constants like 'CST\_HALF\_FS', 'CST\_16BLSB2DEGHS', 'CST\_VNOM', 'CST\_PNOM', 'CST\_ADCLKFREQ', 'CST\_ADCLK\_PERIOD', 'CST\_PREPRO\_RATE', 'CST\_PREPRO\_PERIOD', 'CST\_IQ\_RATE', and 'CST\_IQ\_PERIOD'. At the bottom, there is a 'reg' table with columns for 'name', 'address', and 'description', listing 'control1', 'status1', and 'fault1'. The status bar at the bottom indicates the memory map is loaded.

name	width	description
bpTriggers	4	

name	value	type	description
CST_HALF_FS	16384.0		Half full scale of 16bit signed number
CST_16BLSB2DEGHS	0.0054931640625		180/pow(2,15)
CST_VNOM	3000000.0		Nominal voltage in [V]
CST_PNOM	1000000.0		Nominal voltage in [W]
CST_ADCLKFREQ	100e6		ADC clock frequency [Hz]
CST_ADCLK_PERIOD	10		ADCLK period in [ns]
CST_PREPRO_RATE	512.0		Decimation rate of signal processing unit in FPGA
CST_PREPRO_PERIOD	5632.0		Period of preProErr and SVCAV signals [ns]
CST_IQ_RATE	32.0		Decimation rate of IQ signal decoder
CST_IQ_PERIOD	352.0		Period of preProErr and SVCAV signals [ns]

name	address	description
control1	next	default value
status1	next	default value
fault1	next	default value

# Other features

## Additional code generators

- C++ driver wrapper
  - CERN's EDGE driver generator,
  - Other HW backends (e.g. mmap...)
- FESA class skeletons

## Remote HW access:

- for development and testing using Python

```
auto tcwProxy = drv->tunCtrlRFWinGen.control.getProxy();

//allow extra enum value that triggers directly from WriteHW
auto tunerTriggerSource = pDev->sfTunerTriggerSource.get(pCtxt);
bool triggeredByWriteHW = tunerTriggerSource == rfWinGenTrigSel::write_hw;
if (triggeredByWriteHW)
    tunerTriggerSource = rfWinGenTrigSel::soft_trig;

tcwProxy.rfWinTrigSel.set(tunerTriggerSource);
tcwProxy.repeat.set(pDev->sfTunerMeasurementRepeat.get(pCtxt));
tcwProxy.clrPulse.set(true); //always reset the counter
tcwProxy.softTrigOFF.set(true); //stop the FSM
tcwProxy.write();
```

**Questions?**

# Backup: example (1)

```
memory-map:  
  bus: wb-32-be  
  name: demo_all  
  description: an example with all the features  
  children:  
    - reg:  
      name: reg0  
      description: a normal reg with some fields  
      width: 32  
      access: rw
```

# Backup: example (2)

- reg:

```
name: preProErrSel
description: pre-processing error selection channel\ndefault value
width: 32
access: rw
address: next
x-fesa:
  multiplexed: false
  persistence: true
children:
  - field:
      name: drive
      description: Drive signal error selection
      range: 2-0
      x-enums:
        name: selector
  - field:
      name: vCav
      description: Voltage cavity signal error selection
      range: 5-3
      x-enums:
        name: selector
```

# Backup: example (3)

- reg:

```
name: avgSVCav
width: 32
access: ro
description: avg square cavity Voltage\ndefault value
type: unsigned
comment: value is squared but calculation is done in the software
x-fesa:
  persistence: false
  multiplexed: false
x-conversions:
  read: sqrt(val/pow(2.0,30.0))
```