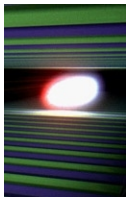
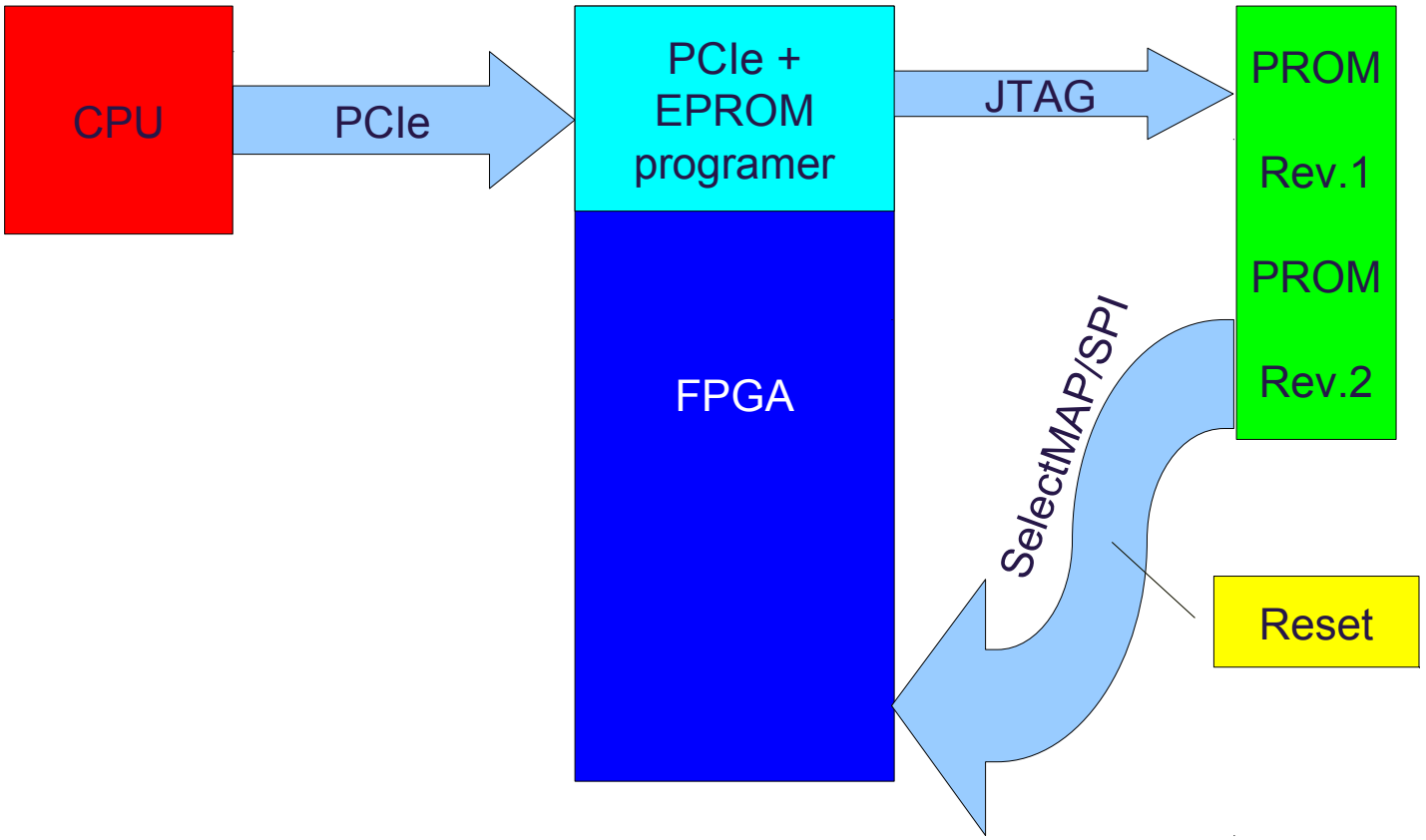
European XFEL



Partial reconfiguration

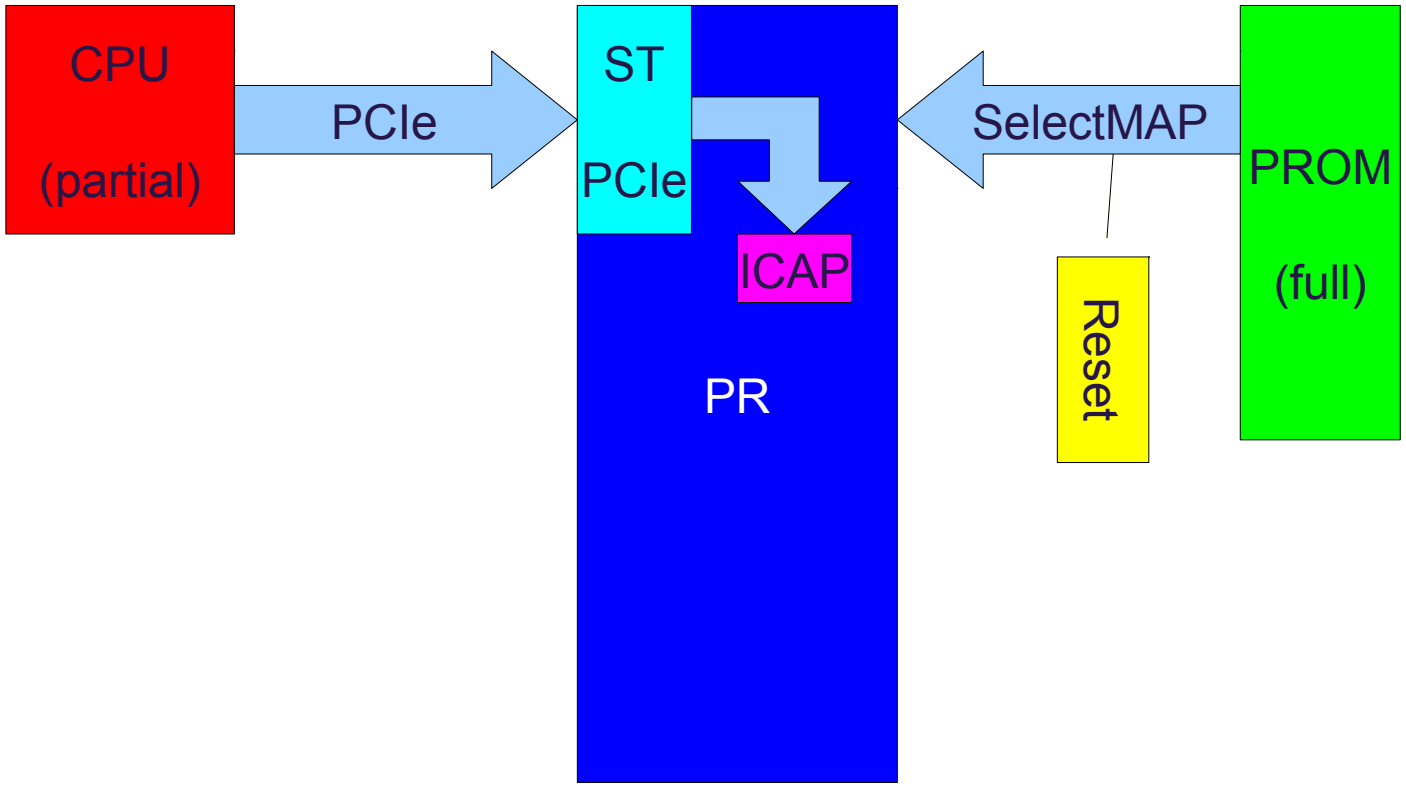
M.Grecki

Remote booting of FPGA (current solution)

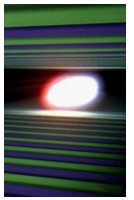


It is rather “remote EPROM programing” than remote FPGA booting! (15 min.)

Remote booting of FPGA (first proposed solution)

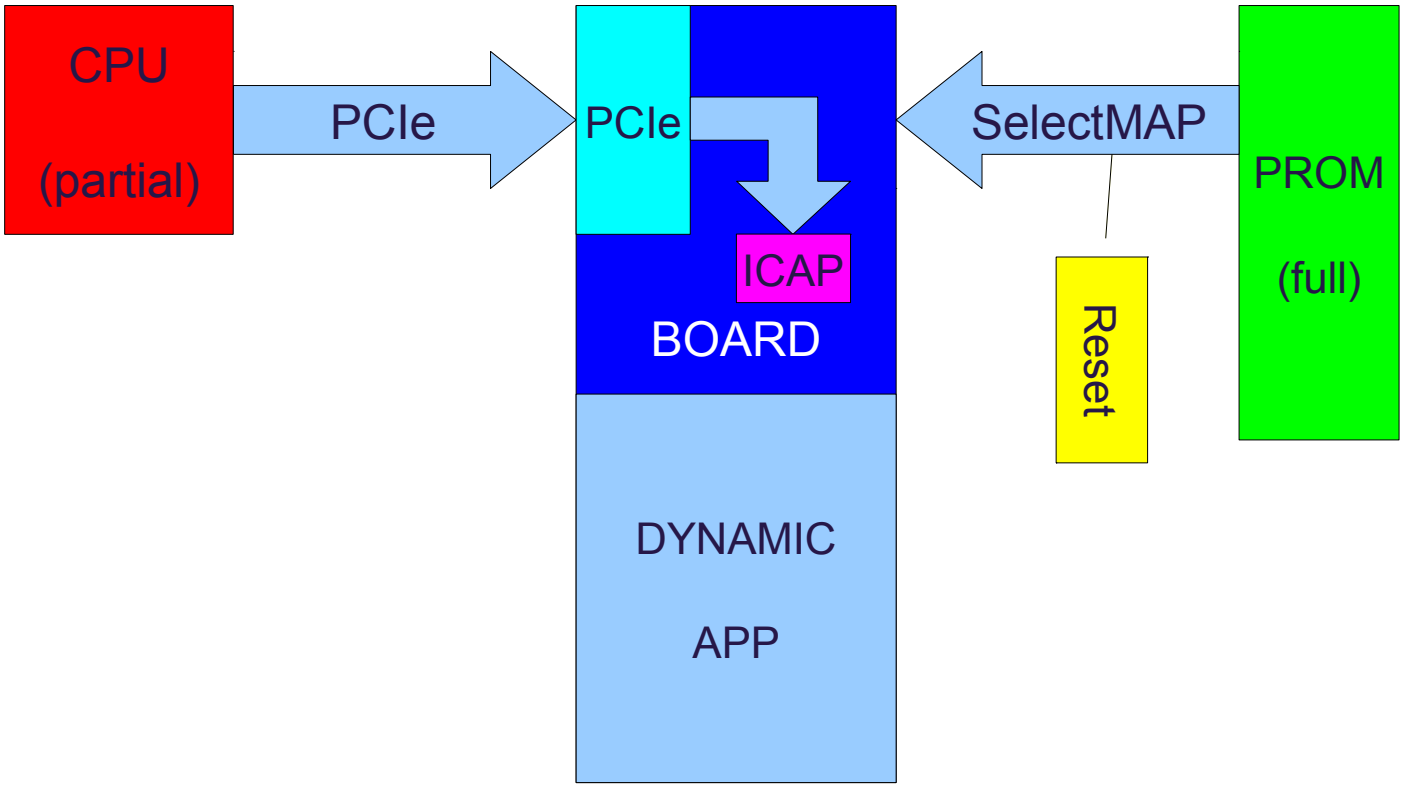


Design Considerations

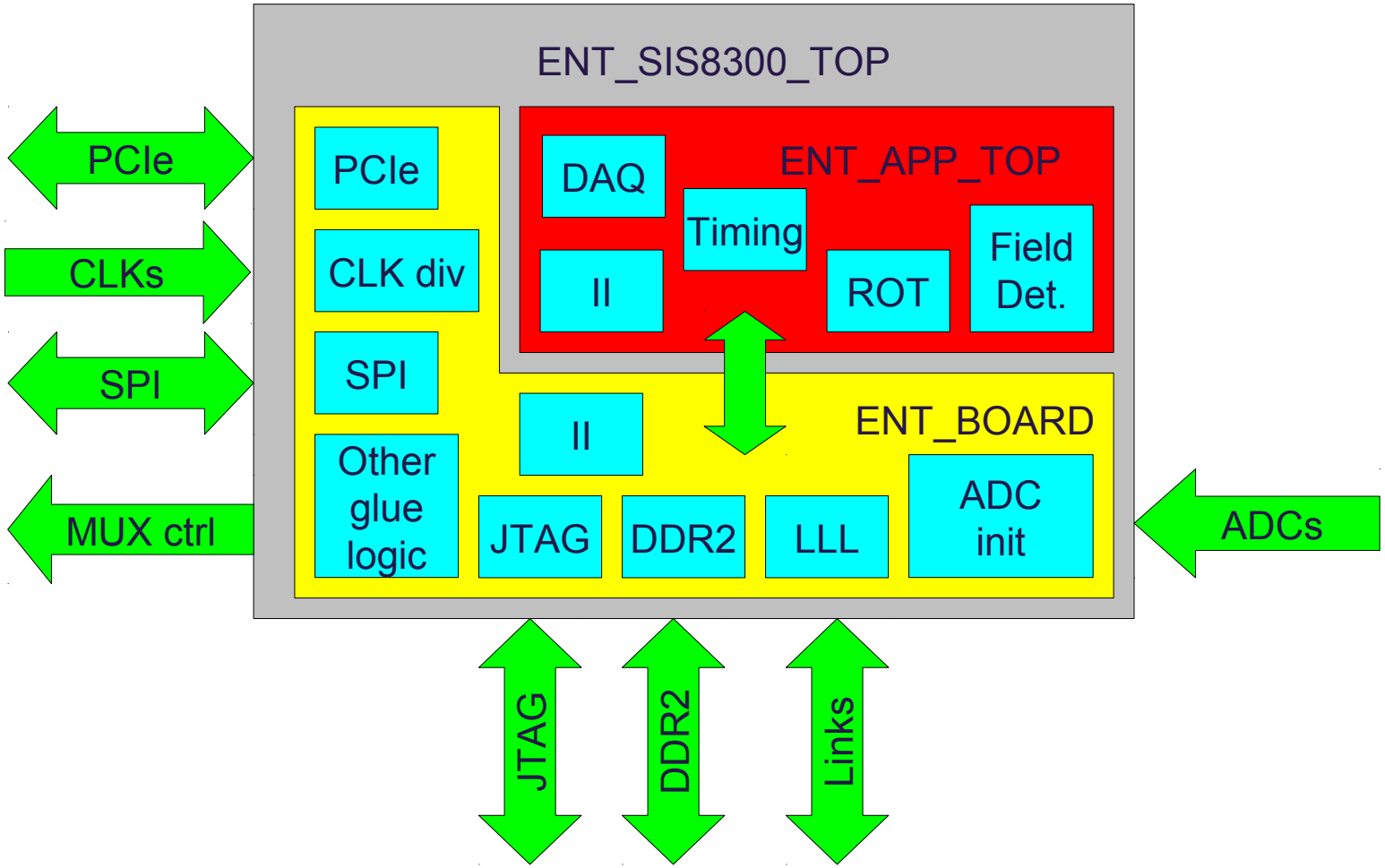


- Some component types can be reconfigured and some cannot.
  - Clocks and Clock Modifying Logic must reside in the static region.
    - Includes BUFG, BUFR, MMCM, PLL, DCM, and similar
  - The following components must reside in the static region:
    - I/O and I/O related components
    - MGT (Multi-Gigabit Transceivers) and related components
    - Individual architecture feature components (such as BSCAN, STARTUP, etc.) must remain in the static region of the design
- Global clocking resources to Reconfigurable Partitions are limited, depending on the device and on the clock regions occupied by these Reconfigurable Partitions. See Global Clocking Rules in Chapter 7 for more information.
  - IP restrictions may occur due to components used to implement the IP. Examples include:
  - ChipScope ICON (BUFG)
  - EDK blocks with global buffers
  - MIG controller (MMCM)
- No bidirectional interfaces are permitted between static and reconfigurable regions, except in the case where there is a dedicated route. For example, a bidirectional I/O buffer (such as IOBUF) in the reconfigurable region routed to a top level I/O pad in the static logic can cross between the reconfigurable region and static logic via a bidirectional interface.

Remote booting of FPGA (implemented solution)

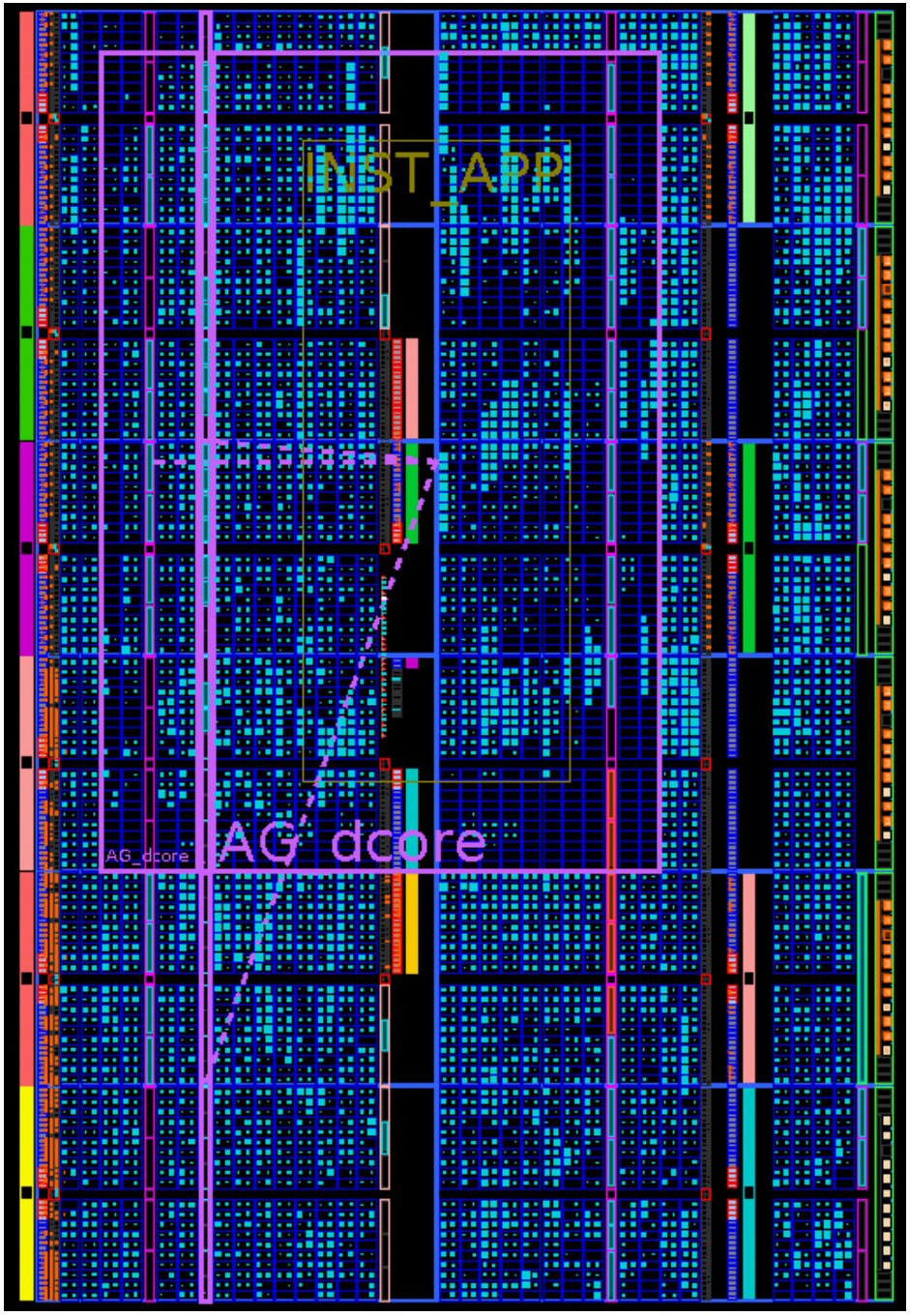


Architecture of SIS8300 firmware

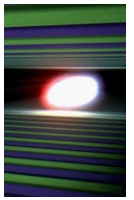


Device Utilization Summary:

Number of BSCANs	1	out of	4	25%
Number of BUFDSs	2	out of	6	33%
Number of BUFGs	17	out of	32	53%
Number of BUFIOs	13	out of	56	23%
Number of BUFRs	5	out of	24	20%
Number of DCM_ADVs	1	out of	12	8%
Number of DSP48Es	**20**	**out of**	**48**	**41%**
Number of FIFO36_72_EXPs	2	out of	60	3%
Number of FIFO36_EXPs	1	out of	60	1%
Number of GTP_DUALs	**5**	**out of**	**6**	**83%**
Number of ICAPs	1	out of	2	50%
Number of IDELAYCTRLs	8	out of	16	50%
Number of LOCed IDELAYCTRLs	5	out of	8	62%
Number of ILOGICs	152	out of	560	27%
Number of LOCed ILOGICs	8	out of	152	5%
Number of External IOBs	329	out of	480	68%
Number of LOCed IOBs	243	out of	329	73%
Number of IODELAYs	85	out of	560	15%
Number of LOCed IODELAYs	8	out of	85	9%
Number of External IPADs	24	out of	518	4%
Number of LOCed IPADs	12	out of	24	50%
Number of OLOGICs	112	out of	560	20%
Number of External OPADs	20	out of	24	83%
Number of LOCed OPADs	8	out of	20	40%
Number of PCIEs	**1**	**out of**	**1**	**100%**
Number of PLL_ADVs	3	out of	6	50%
Number of RAMB18X2SDPs	1	out of	60	1%
Number of RAMB36SDP_EXPs	10	out of	60	16%
Number of LOCed RAMB36SDP_EXPs	1	out of	10	10%
Number of RAMB36_EXPs	14	out of	60	23%
Number of LOCed RAMB36_EXPs	4	out of	14	28%
Number of RAMBFIFO18s	10	out of	60	16%
Number of RAMBFIFO18_36s	2	out of	60	3%
Number of Slices	**5981**	**out of**	**7200**	**83%**
Number of Slice Registers	14515	out of	28800	50%
Number used as Flip Flops	14410			
Number used as Latches	105			
Number of Slice LUTS	14195	out of	28800	49%
Number of Slice LUT-Flip Flop pairs	19113	out of	28800	66%



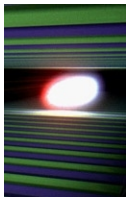
Resource consumption for NonPR/PR project



Device Utilization Summary:	No PR		PR		Increase
...					
Number of BUFGs	13	->	17	/ 32	12.5%
...					
Number of OLOGICs	129	->	112	/ 560	-3%
...					
Number of Slices	5815	->	5981	/ 7200	2.3%
Number of Slice Registers	14406	->	4515	/ 28800	0.4%
Number of Slice LUTS	13060	->	14195	/ 28800	4%
Number of Slice LUT-Flip Flop pairs	18406	->	19113	/ 28800	4%

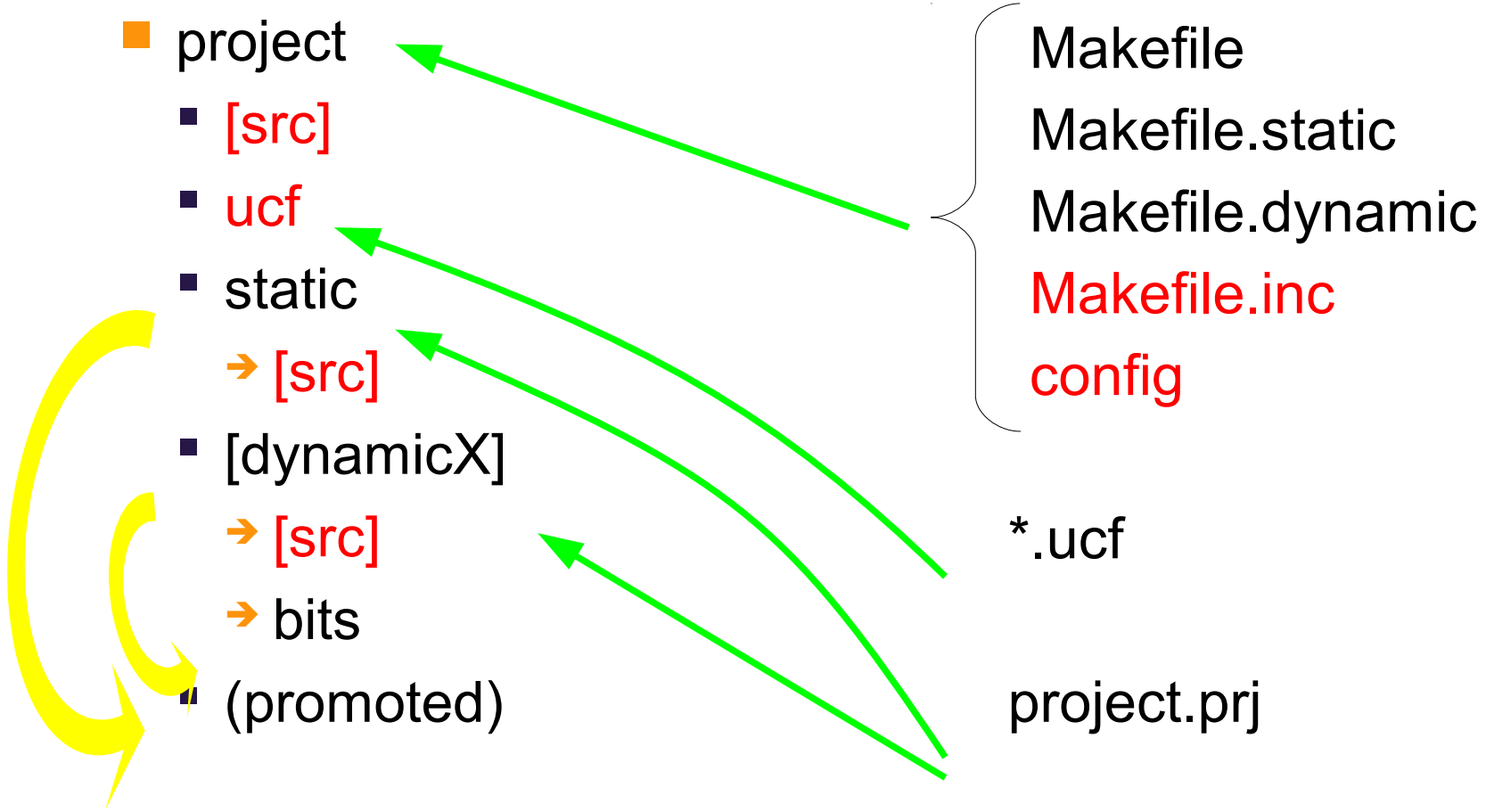
- compilation time:
  - NonPR project – 18 min.
  - PR project
    - whole project (static & dynamic part, implementation) – 19 min.
    - only update of dynamic part and implementation – 13 min.

Batch mode compilation

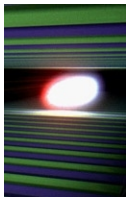


- project template prepared (sources, support files, directories structure, Makefiles) for Partial Reconfiguration projects adjusted to remote booting
- covers automatically the whole design flow from vhdl source up to bitstream generation
- area for reconfigurable core defined (but certainly require adjustment for final project)
- software for bitstream download prepared (initial ad-hoc version but working)
- automatic version generation (version number kept in config file automatically increased after successful compilation)
- all compilation reports and config kept for analysis
- script helping to optimize the area allocation

Project structure

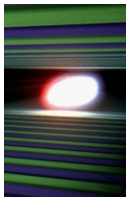


Compilation flow



- compilation of static part
- compilation of dynamic part (module 0)
- implementation of static+dynamic0 module
- partition promotion
- compilation and implementation of all other dynamic modules using previously promoted static partition
- final verification of bitstream suitability for partial verification

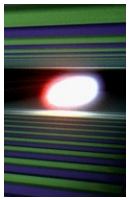
Usage



- put sources in adequate directories, create *project.prj* files for static and dynamic parts, edit *Makefile.inc* and *config* to adjust settings
- *make*
  - makes all targets (finally produces bitstream files for programming, reports, increases compilation number), can make dynamic targets in parallel (*-j n*)
- *rprog*
  - reprograming (writes bitstream to pcie bar0)
- *make clean; make cleanreps*
  - removes all previous compilation results and reports

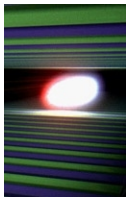
European XFEL

Benefits of proposed solution



- **MUCH** faster reprograming time (~1 second)
- Hot-plug not needed after reprograming
- No EPROM reprograming
  - saving EPROM lifetime
  - damage of EPROM contents not possible
- EPROM programmer does not need to be integrated in all designs – it can be just loaded when needed
- Compilation of dynamic part only. Static part simply imported (saves compilation time)
- Extra bonus – configuration memory readout for SEU detection

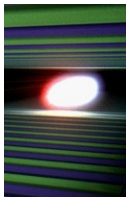
Conclusion



- easy to use tool for SIS8300 firmware development
- fast and easy firmware download (~1 sec.)
- readout of configuration memory possible

To do:

- generate the „basic” project and program all the SIS8300 board in use and in the storage (rev. 0 firmware)
- prepare the same for uTC



Thanks for you attention!