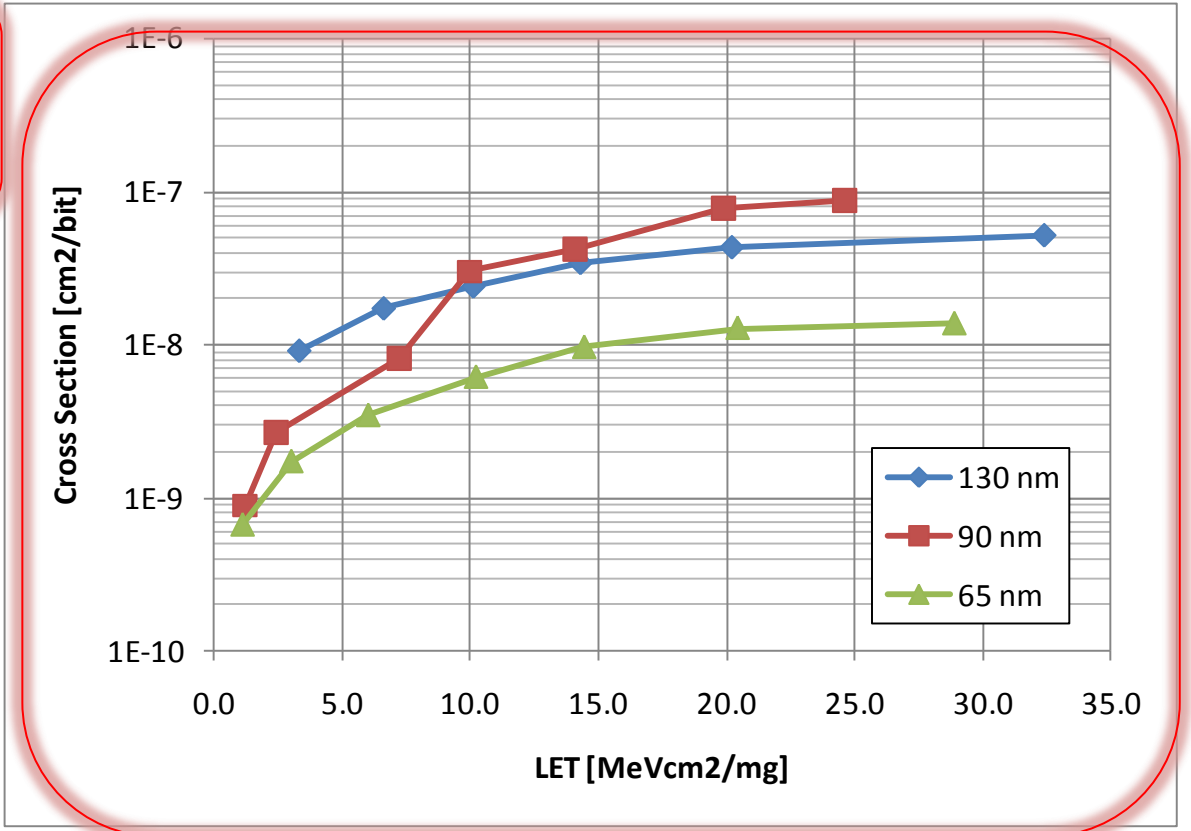
SEU WK summary

Technology comparison

- 65nm seems to saturate at a cross-section 3.4× less than 130nm
  - About proportional to 4× area reduction
- 90nm registers were custom-made (not standard cells)
  - Higher saturation cross-section though area is ½ of cell in 130 nm
- LET thresholds are less than 1.1 MeVcm2/mg for all technologies

  (all plots @1.2V supply)

- Note: SEU-robust cells are well below 10^{-10} cm^2/bit



- For more info on 65 nm (TID, ...)
  - See talk: "*Characterization of a commercial 65nm CMOS technology for SLHC applications*"

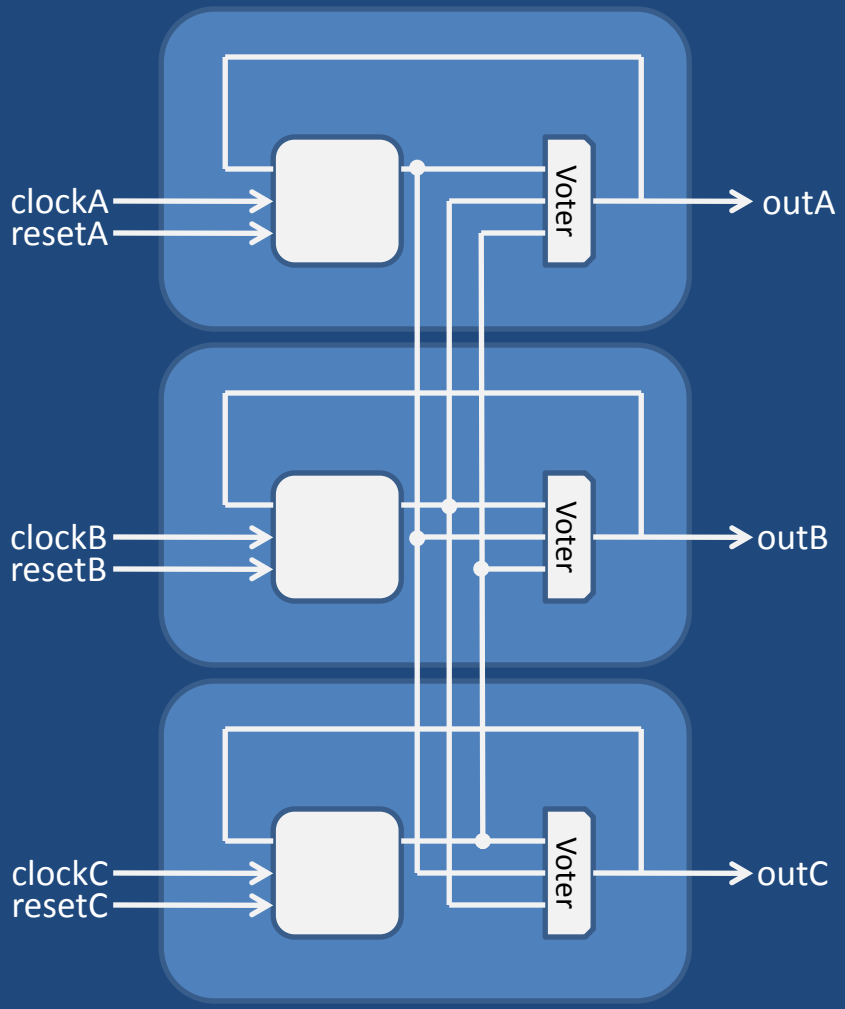
Tech.	Cell size	Area [um2]
130 nm	7.6×3.6	27.36
90 nm	3.8×1.8	10.80
65 nm	5.4×2.0	6.84

Difference from power supply voltage, triple well, register/SRAM, Dynamic/static below x2

Triple-voting in standard cell logic

Three independent clock trees

Three data paths with voters after every register stage



Full-custom/High Speed, Config Regs

PLL VCOs: enlarge transistor sizes & bias with larger currents
=> reduce sensitivity to transients (& hence induced jitter)

Traditional voting => high propagation delays (4.8 GHz not achievable)
=> Use 'transistor-voting' inside custom flip-flops

Configuration Registers

Use design from GBLD chip (no clock)

Voting gates generate a clock pulse to re-latch correct value

Reference:
O. Cobanoglu, 'A radiation tolerant 4.8Gb/s serialiser for the GBT', TWEPP 2009

Results

Continuous BERT during irradiation
Read back config registers every 2 secs

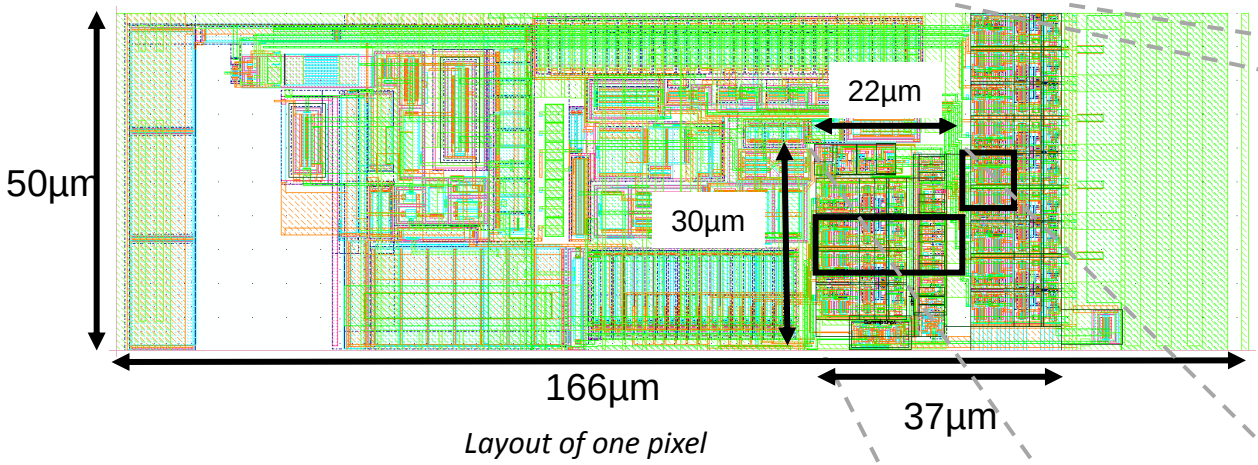
- No latch up
- No configuration upsets
- No errors in standard-cell logic (loopback test without SERDES)

TX-mode

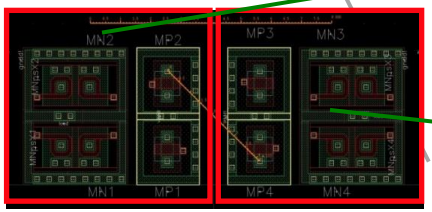
- LET threshold ~ 15 MeVcm2/mg..... prediction of low upset rate from Federico
- Some upsets cause large number of errors in a frame..... To be understood

RX-mode.... To be tested

Latches pixel implementation 2/2



Layout of one pixel



A1 A2

DICE latch

A1	C2
B1	D2
C1	A2
D1	B2

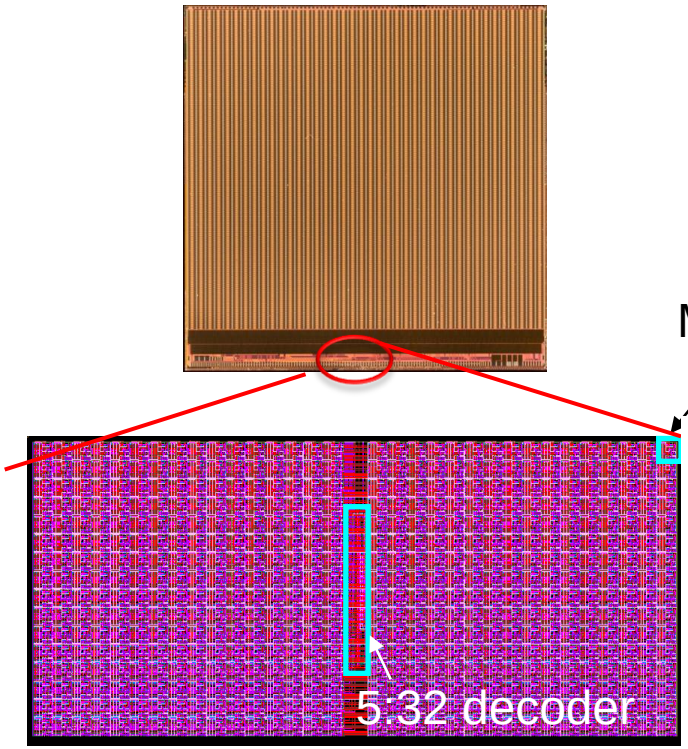
A1	C2
B1	D2
C1	A2
D1	B2
E1	G2
F1	H2
G1	I2
H1	E2
I1	F2

Interleaved structure version B

1 elementary cell=2 inverters

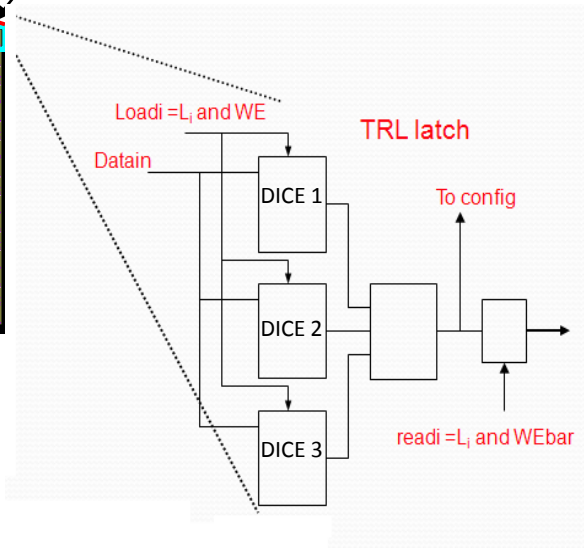
- The DICE latch consists of 2 elementary memory cells to form the 13 bits configuration memory of the pixel.
- In order to separate sensitive pair nodes and improve the SEU tolerance, we used interleaved layout for each latch in the pixel configuration bloc (as shown next)
- Delicate operation for several reasons:
  - It increases the dedicate area (+25%)
  - It complicates the interconnection between elementary cells

Memory for global configuration

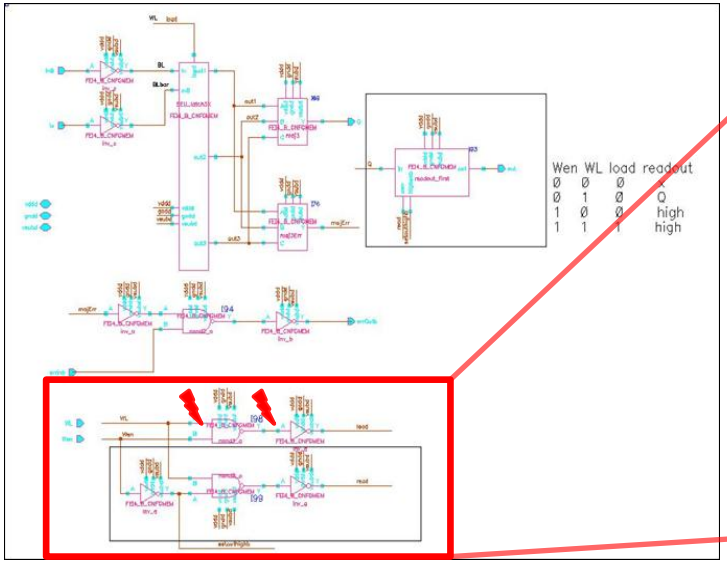


Layout of the global memory (900µmx360µm)

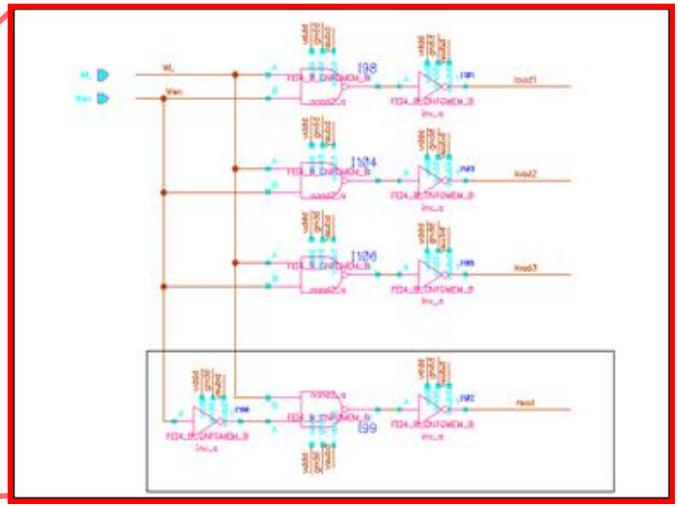
- 512 bits are stored in a Triple Redundant DICE Latches (TRL)
  - TRL Cell area : 27µm × 19µm
  - Bloc area : 900 µm × 360µm
- Errors are determined with comparing the Read -back data to the loaded data



SEU sensitivity improvement



Memory cell unit schematic



New load path

- The load input signal is common to the 3 latches of the cell memory:
  - A glitch in the internal NAND or inverter causes a glitch on the load signal.
  - In this case the current value on the data bus is copied in the memory and can create an error.
- In order to reduce the sensitivity to glitches, we triplicate the load path
  - This cell has been modified in the new submitted version of FEI4 chip (FEI4-B),
  - we hope to reduce considerably this sensitivity from 0.016 to 0.002 errors/spill where 0.002 a value that comes from previous measurements.

“abcnasic” SEU strategies

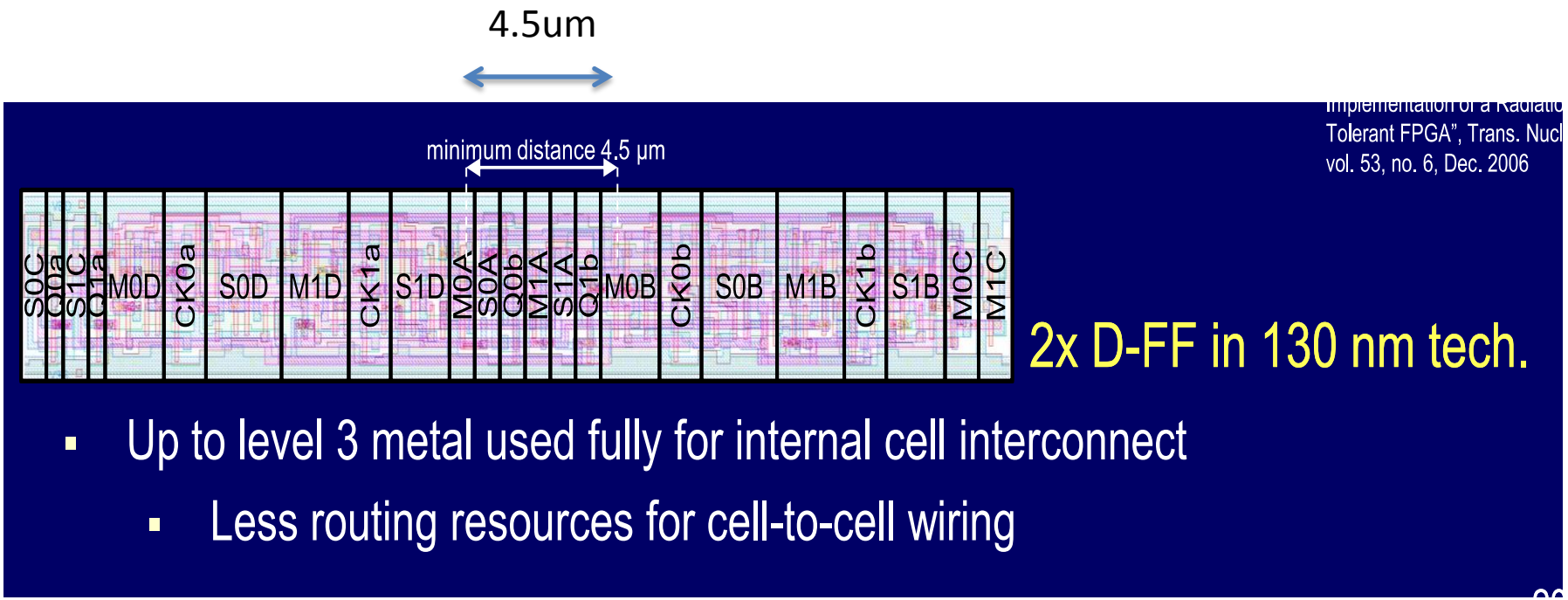
- Simple ideas and proposals
- Assumptions based on :
  - Tentative to limit the triplications where strictly needed
  - In criticality order

Criticality	Method	Example
Very critical	Block triplication	Critical Commands
Critical	ECC (Hamming) Local triplication	Sequencers
Less	specific	FIFO controls
Low	DICE cell	Baseline every where else (?)

S. Bonacini CERN

Dice Cell

Double DFF with interleaved subblocks to separate redundant nodes



The DICE cell is currently ported to the current version of the IBM 130nm cell library (not Artisan) by Filipe De Sousa

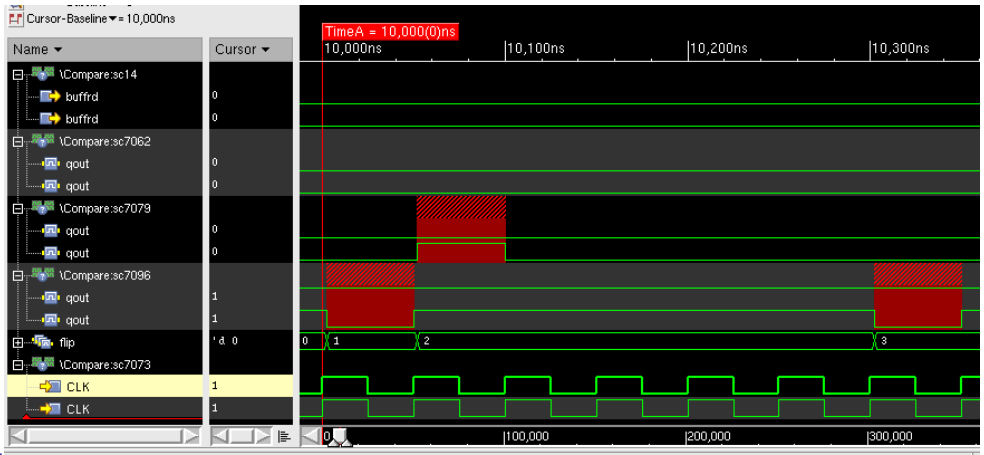
SEU protection

- Goal:
  - Investigate high level SEU mitigation techniques with low power/area overhead and high effectiveness.
- To do:
  - Validate each SEU mitigation technique by simulation.
  - Power/area cost and effectiveness will be evaluated and compared after synthesis.
- SEU mitigation techniques:
  - Triplication
  - Hamming code
  - …

SEU insertion in simulation

- The script search for every register that can be upset and present the list to the designer
- The designer choose which register [or multiple] to upset
- The designer may also specify when in the simulation the SEU should occur
- Using the same testbench the comparison between a simulation with and without SEU is very practical using the a comparison tool from the simulator program.

```
N DCL.controller.controller.odd_dcl_done_reg.i0.r1.Q
N DCL.controller.controller.\WAIT_hamming_reg[1] .i0.r1.Q
N DCL.controller.controller.datavalid_reg.i0.r1.Q
y DCL.controller.controller.\visual_WAIT_current_reg[0] .i0.r1.Q
N DCL.controller.controller.even_dcl_done_reg.i0.r1.Q
N DCL.controller.controller.even_scan_reg.i0.r1.Q
N DCL.controller.controller.scan_reg.i0.r1.Q
N DCL.controller.controller.\WAIT_hamming_reg[0] .i0.r1.Q
N DCL.controller.controller.odd_next_reg.i0.r1.Q
N DCL.controller.controller.odd_scan_reg.i0.r1.Q
N DCL.controller.controller.\ESC_WAIT_hamming_reg[0] .i0.r1.Q
N DCL.controller.controller.\OSC_WAIT_hamming_reg[1] .i0.r1.Q
N DCL.controller.controller.\visual_OSC_WAIT_current_reg[1] .i0.r1.Q
N DCL.controller.controller.odd_selected_reg.i0.r1.Q
N DCL.controller.controller.\visual_ESC_WAIT_current_reg[2] .i0.r1.Q
y DCL.controller.controller.\visual_WAIT_current_reg[2] .i0.r1.Q
N DCL.controller.controller.even_next_reg.i0.r1.Q
N DCL.controller.controller.\OSC_WAIT_hamming_reg[0] .i0.r1.Q
N DCL.controller.controller.\visual_ESC_WAIT_current_reg[1] .i0.r1.Q
N DCL.controller.controller.\visual_OSC_WAIT_current_reg[2] .i0.r1.Q
N DCL.controller.controller.\visual_ESC_WAIT_current_reg[0] .i0.r1.Q
N DCL.controller.controller.\visual_OSC_WAIT_current_reg[0] .i0.r1.Q
N DCL.controller.controller.buffrd_reg.i0.r1.Q
N DCL.controller.controller.\OSC_WAIT_hamming_reg[2] .i0.r1.Q
N DCL.controller.controller.\ESC_WAIT_hamming_reg[2] .i0.r1.Q
y DCL.controller.controller.\visual_WAIT_current_reg[1] .i0.r1.Q
```



□ □□□ □□□ □□ □&. □□□□□□□□□□□□□□□□□

% ! %!" & / 6 3

&" ! " 1! * + ! 1%" %+ ! / ! @3

&" " " 4 " ! " % ! " 8 " &

! * % %*% *" / ' . " !* !3(

) " + %& *" % ! * 6 " ! % - +

) %! %* 6 */ 3 6 . " 7 !- * / 3

? 6 * ? * 4

* +% !" " /) *!" + + & 3

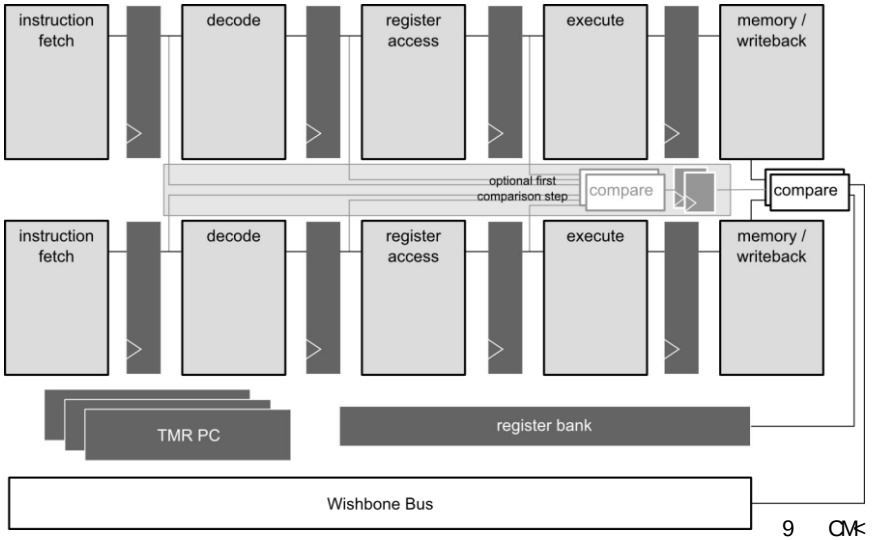
% %*, / ' . 1% & 6 3

&" ! " 1" 4 " ! " " 8 " & 1%" %+ !

" ! ! >%! % & #% *! ! *" % +" " *!

/&" " . ' % ! !- " + ! +%+ &" %!" 3

A



" * " !% " %" & *' &

+ % #%* !% "
"# " " , ' !" %+.

+% " #&#" " +" (
" "!) 0 ! *% ! %& &
%&& " % & %&%" "
%*# ** ' +%+ *%

*,) % 0 !" ! *' & /&%%- % ! " ." ! %*&3

+% " #" ' %+- & (0) " "! /& % *'
&" %&* +. 3

" 0 6 1 ! 6

." " &(? * 4 " " @& " L &") ' ") 0

SEU Summary

- There is now a much better understanding in our community of the SEU sensitive of memory elements in 130, 90 and 65 nm technologies (~scales with sensitive area)
  - Voltage, Triple well, Registers/SRAM (difference < ~ x2)
  - (do not confuse cross section from high ionizing particles (LET) with hadron cross section)
- Basic protection schemes well know
  - TMR: Single voter, Triple voter, Triple clocking
  - Hamming (and other error correction codes)
  - Special latch/registers: DICE
- Appropriate/optimal protection scheme depends on information type and system effects.
- Tools/approaches for HDL protection insertion schemes and fault injection simulation
  - FPGA: Multiple tools appearing
  - ASIC: Some home made scripts being used.
- Watch out for Multiple bit errors and common signals across redundant elements (e.g. clock, load, reset , ,).
- Do not map directly basic approaches/conclusions between ASICs and FPGAs
- We will in the future have the SEU working group as integral part of MUG.