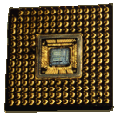
A strategy for the prediction of the error rate occuring in digital architectures as the consequence of radiations

R. Velazco[1], R. Ecoffet[2]

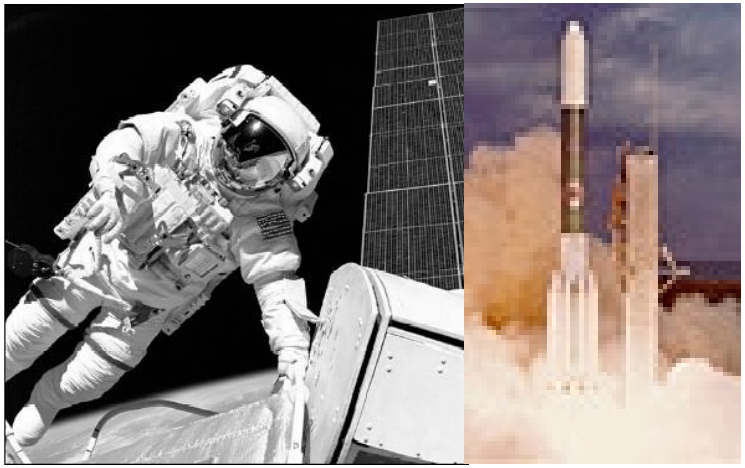
[1] TIMA Laboratory
«Circuit Qualification» Research group
Grenoble - France
http://tima.imag.fr

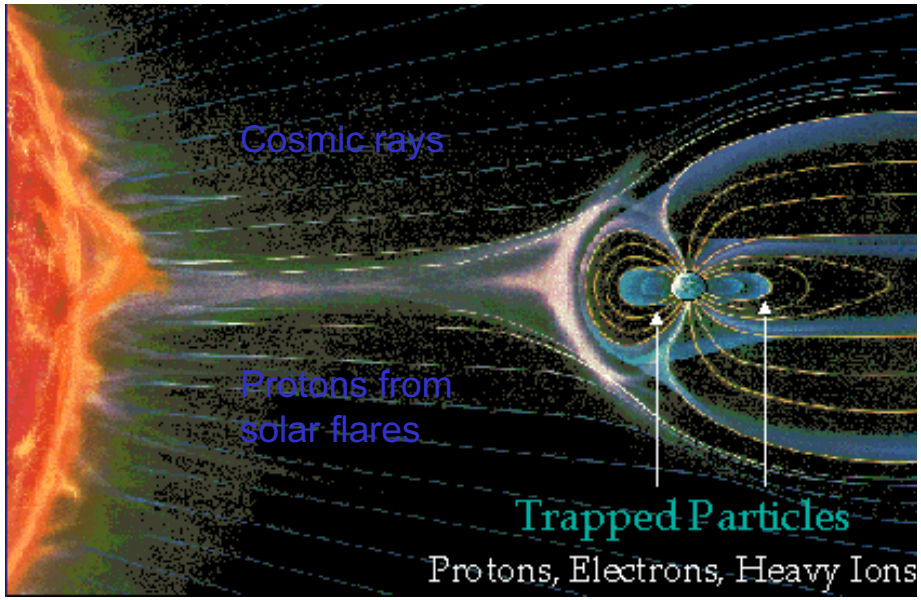
[2] CNES
Space Env. & Radiation Effects Dept.
Toulouse – France



Motivations

• Aerospace electronic systems operate in a radiation environment

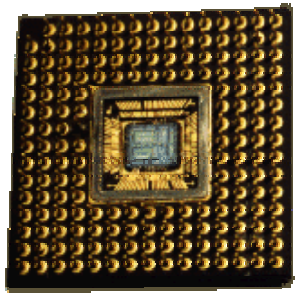


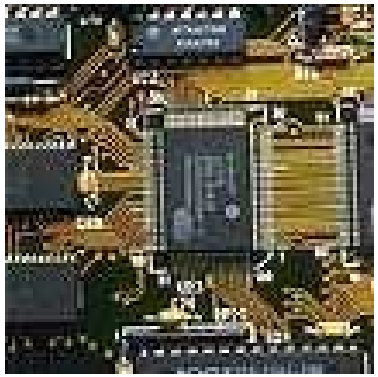


• Charged particles come from three main sources: Van Allen Belts, Cosmic Rays & Solar Flares

Motivation

- The microelectronic technology is constantly changing:
  - higher density,
  - faster devices,
  - lower power.



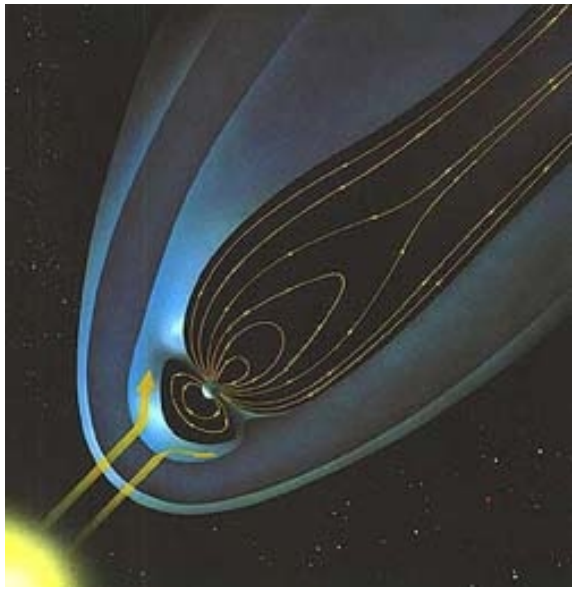


- These increase the devices' vulnerability to the effects of radiation (nuclear and space environments).
- Space Agencies favor the use of COTS technologies.
- Future submicronic technologies are potentially sensitive to the effects of atmospheric neutrons.

Motivation (cont'd)

Space radiation

- Light particles
- Heavy ions



Effects of radiation on ICs :

- Total dose (permanent effects)
- Single Events Effects (SEE):

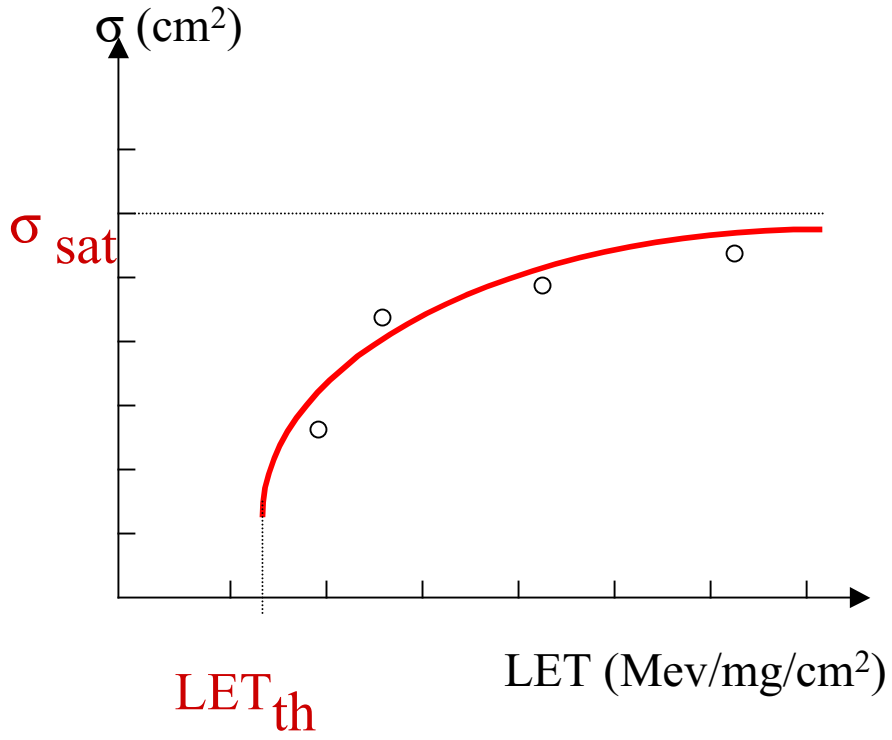
  SELs: latchups (shorts between Gnd and Power)

  SEUs: upsets (bit flips)

Motivation (cont'd)

S.E.E. sensitivity estimation:

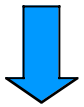
- The studied circuit is exposed to a suitable particle beam
- Events (latchups and upsets) are detected on-line
- A cross-section is derived:
  $\sigma(LET)$ = Nb of events/ Fluence



The application error rate is derived from both the calculated cross-section and the features of the final environment

Motivation (cont'd)

- Radiation testing of digital processors is usually performed with "static strategies" (register test) or with simple applications.



Which is the representativity of derived error rates with respect to those of the final application?

- **<u>Objective of our work</u>: Investigating the efficiency of a strategy for the prediction of SEU error rates based on upset-like error injection on microprocessor-based architectures**

Outline

1. **A two-step error rate prediction methodology**
   - Techniques for upset fault injection
   - The CEU approach
   - Software based upset injection strategy

2. **A case study: the DSP32C running a CMA equalizer**
   - Results of upset injection sessions
   - Radiation ground testing
   - Error rate prediction

3. **Conclusions and future work**

1. A two-step error rate prediction methodology

- **Step 1**: Radiation ground test in a particle accelerator facility: *static SEU cross-section*

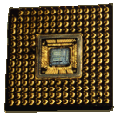
$$\sigma_{SEU} = \#upsets / \#particles \quad (usually\ given\ in\ cm^2)$$

→ How many particles to provoke an upset ?

- **Step 2**: Off-beam upset simulation:

$$\tau_{inj} = \#errors / \#upsets$$

→ How many upsets to provoke an *error in the final application*?



A two-step error rate prediction methodology (cnt'd)

- Error rate estimation:

$$\tau_{SEU} = \sigma_{SEU} * \tau_{inj} \quad \text{[errors/particle]}$$

- Error rate in flight

τ_{SEU} ***Expected particle fluency** **[errors/time unit]**

A two-step error rate prediction methodology (cnt'd)

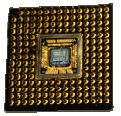
Main benefits

Radiation ground testing performed only once for a given processor but not for each application

Test cost and time drastically decrease

The application upset error rate can be evaluated concurrently with software developments.

Key point: How to perform upset simulations on the chosen application?



1.a) Techniques for upset injection in processors

Hardware based:

- Using particular processor execution modes (Trace, debugging, ...)
- Using asynchronous signals (DMA, Exceptions, Interrupts, ...)

Software based:

- Using a simulator of the processor
- Using a HDL model of the processor

1.b) An example of HW based upset simulation: the CEU approach

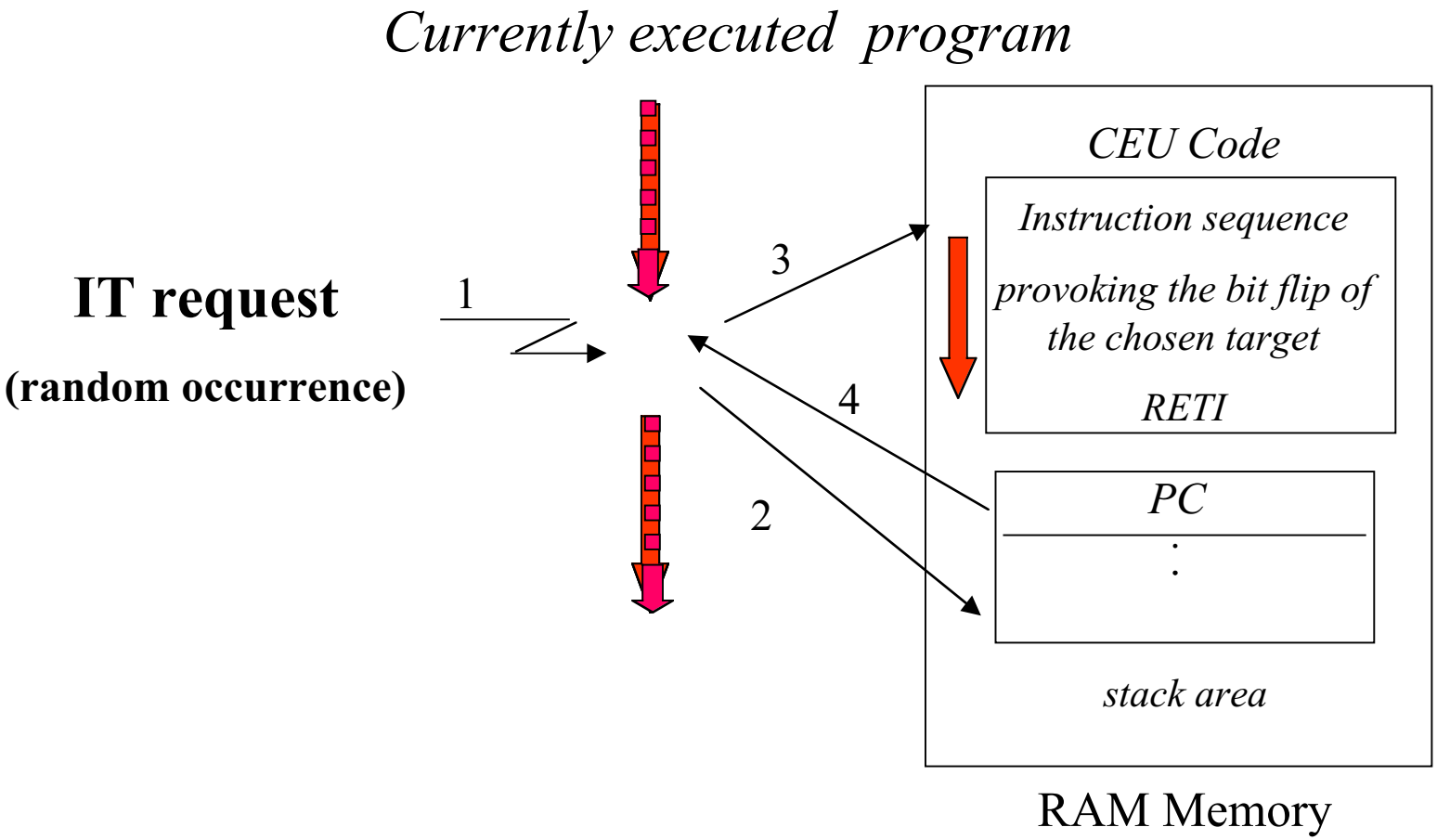
R. Velazco, S. Rezgui, R. Ecoffet., *Predicting error rate for microprocessor-based digital architectures by C.E.U. Injection*, IEEE Trans. on Nuclear Science, Vol. 47, N° 6, Dec. 2000, pp. 2405-2411.

- **Basic idea:**

  Use **Interrupt signals** to inject bit flips in processors

- **Main Steps:**
  - Selection of the CEU target (randomly or exhaustively)
  - Storage of CEU code for upset emulation in a memory zone
  - Execution of the CEU code (interruption signal assertion)
  - Comparison of obtained and expected results

An example of HW based upset simulation: the CEU approach (cnt'd)



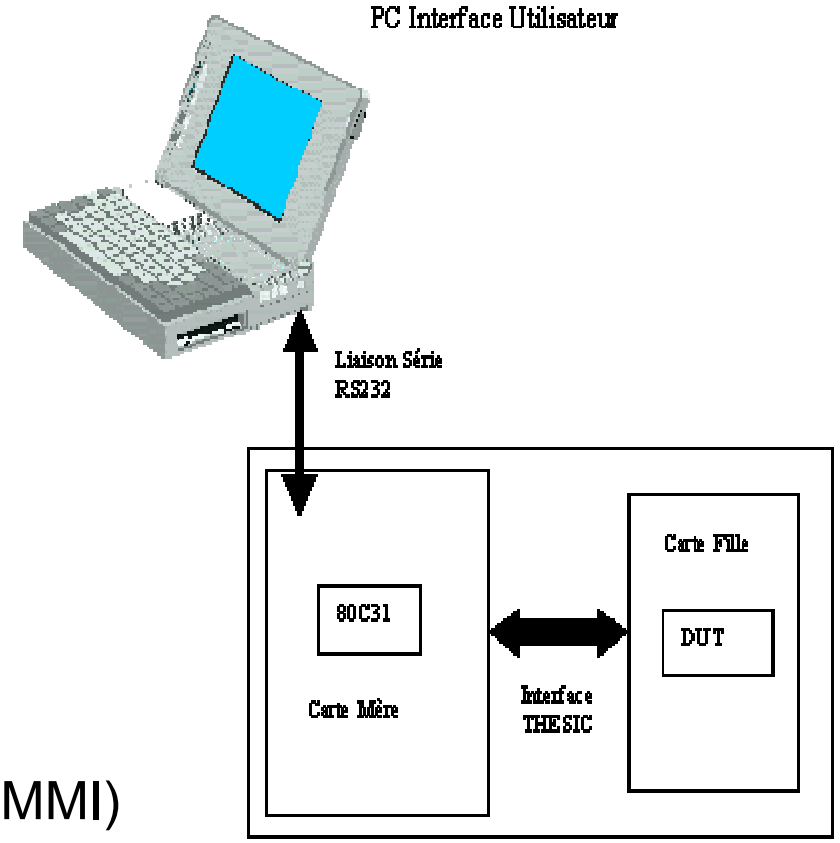
An example of HW based upset simulation (cnt'd): Hardware set-up

THESIC tester features:

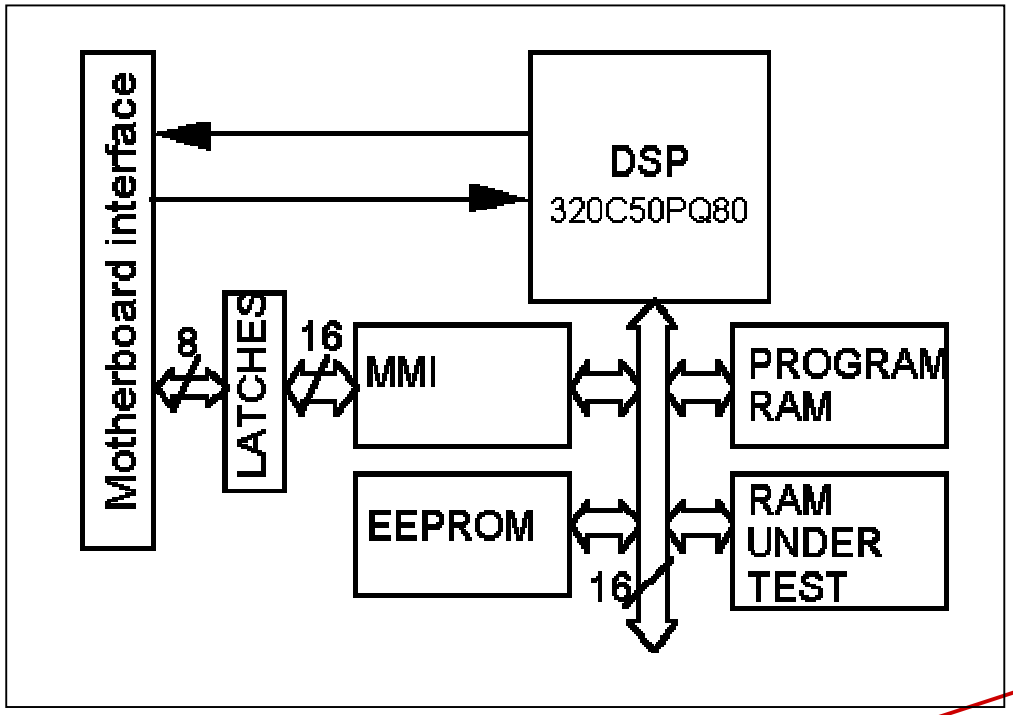
Mother board
- user interface functions
- controlling the test evolution

Daughter board
- "natural" DUT environment
- adapt to mother board protocols.

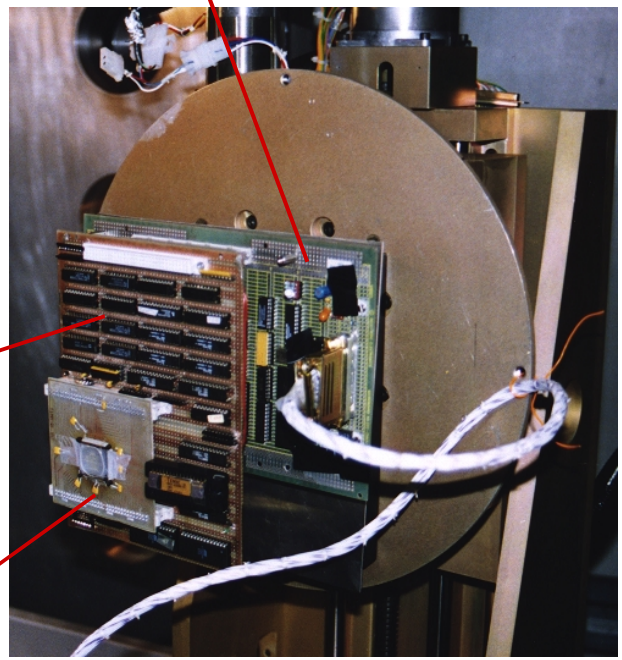
Asynchronous communication
through a memory mapped interface (MMI)



An example of HW based upset simulation (cnt'd) : Hardware set-up



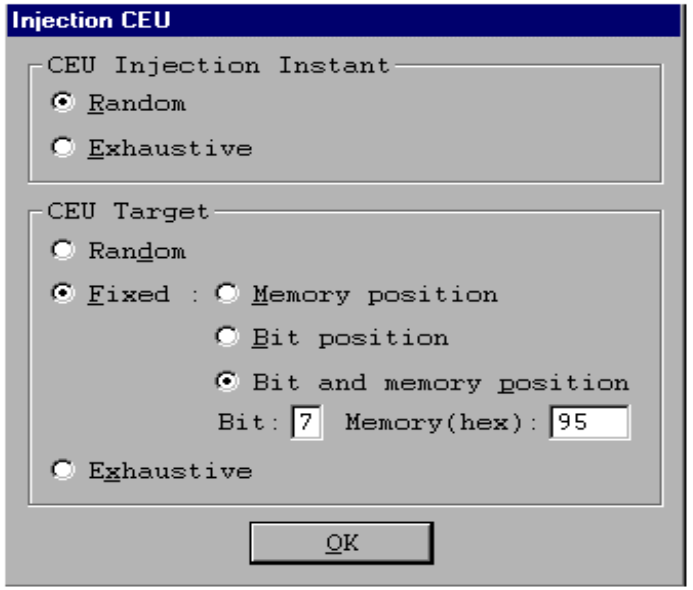
Mother board



Daughter board

DUT

An example of HW based upset simulation (cnt'd) : user interface



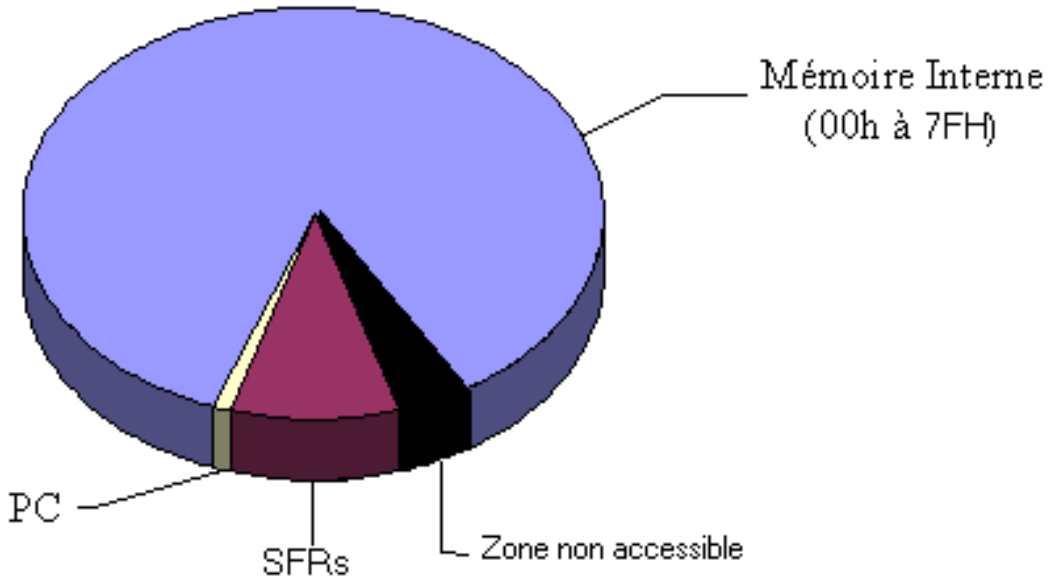
CEU-injection steps:

- CEU target selection
- CEU code preparation
- Interrupt activation
- Result collection

THESIC user's interface for CEU injection

An example of HW based upset simulation (cnt'd) a case study: the 80C51 microcontroller

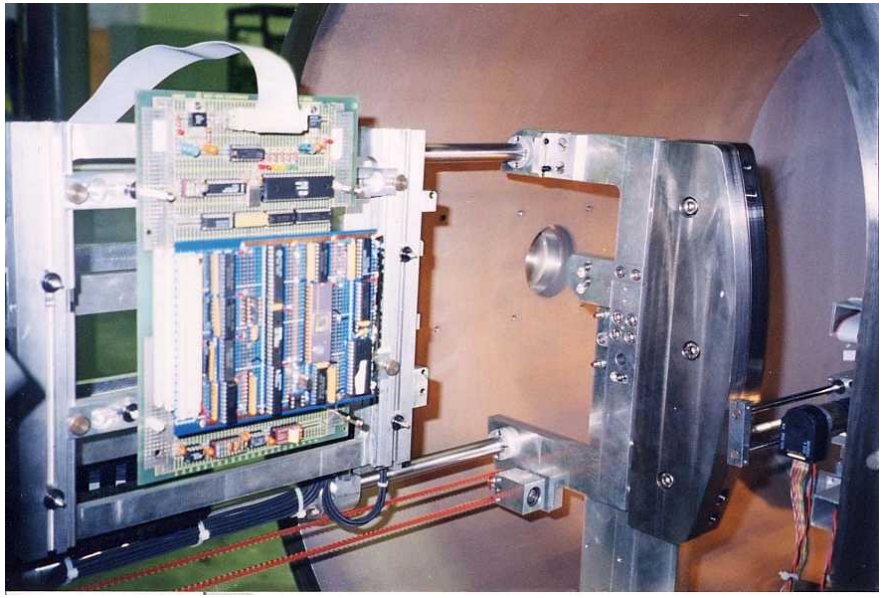
CEU targets for the 8051: accessibility 93%



An example of HW based upset simulation: a case study (cnt'd)

- The 8051 THESIC daughterboard was exposed to heavy ion beams while running a matrix multiplication program.
- The "Cyclone" cyclotron facility of Louvain-la-Neuve was used.
- Main Goals:
  - measure the underlying 8051 SEU cross-section
  - assessing the methodology of error rate prediction

An example of HW based upset simulation (cnt'd): a case study - Radiation testing



The 8051 Thesic daugtherboard at the vacuum chamber of Cyclone

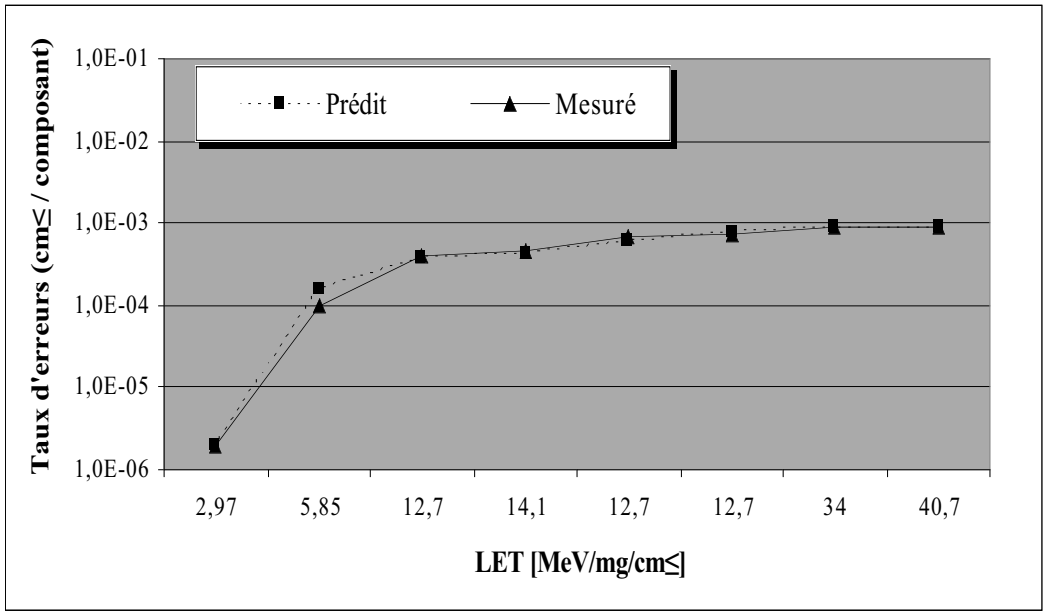
M/Q=5	ENERGY [MEV]	LET [MeV/mg/cm?]
$^{40}Ar^{8+}$	150	14.1
$^{20}Ne^{4+}$	78	5.85
$^{15}N^{3+}$	62	2.97
$^{10}B^{2+}$	41	1.7
$^{84}Kr^{17+}$	316	34

- **M** atomic mass
- **Q** ion charge state

Available beams

An example of HW based upset simulation(cnt'd): Radiation testing measures vs. Predicted results

Particle beam	Effectif LET [MeV/mg/cm?]	Error rate : [cm?/ composant]	
		Measured	Predicted
Nitrogene (N)	2,97	$2{,}00\ 10^{-6}$	$2{,}00\ 10^{-6}$
Neon (Ne)	5,85	$1{,}02\ 10^{-4}$	$1{,}55\ 10^{-4}$
Chlorine (Cl)	12,7	$3{,}96\ 10^{-4}$	$3{,}78\ 10^{-4}$
Argon (Ar)	14,1	$4{,}50\ 10^{-4}$	$4{,}33\ 10^{-4}$
Cl (at 48°)	19,5	$6{,}63\ 10^{-4}$	$6{,}00\ 10^{-4}$
Cl (at 60°)	25,4	$7{,}13\ 10^{-4}$	$7{,}55\ 10^{-4}$
Krypton (Kr)	34	$9{,}12\ 10^{-4}$	$8{,}86\ 10^{-4}$
Bromine (Br)	40,7	$8{,}85\ 10^{-4}$	$9{,}00\ 10^{-4}$



Exposed Program: a 6x6 Matrix Multiplication

An example of HW based upset simulation (cnt'd): Benefits and drawbacks

☺ • Automated injection

☺ • Good experimental results

☺ • Short duration of injection sessions (i.e. 1000 upsets / 3 min.)

☺ • Generic approach

☹ • Not all targets are accessible through the instruction set

☹ • Need to build a hardware prototype incerases developing time and cost

1.c) Software based upset injection strategy

SEU injection may be achieved by means of a SW simulator

- Easier & cheaper implementation
- The targets include a wide set of processor's memory elements
- Suitable to study the effects of SEUs on critical memory zones

Software based upset injection strategy (cnt'd)

GENERIC MODEL for an Upset

1. Program Execution on Processor
2. Stop Execution when time = INSTANT
3. Flip the TARGET bit among all processor bits
4. Resume execution

(*) INSTANT & TARGET are pseudo-randomly chosen

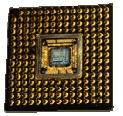
Software based upset injection strategy (cnt'd)

- Using the processor simulator to inject bit flips while executing the studied program

- Command file modeling a bit flip: using simulator breakpoints

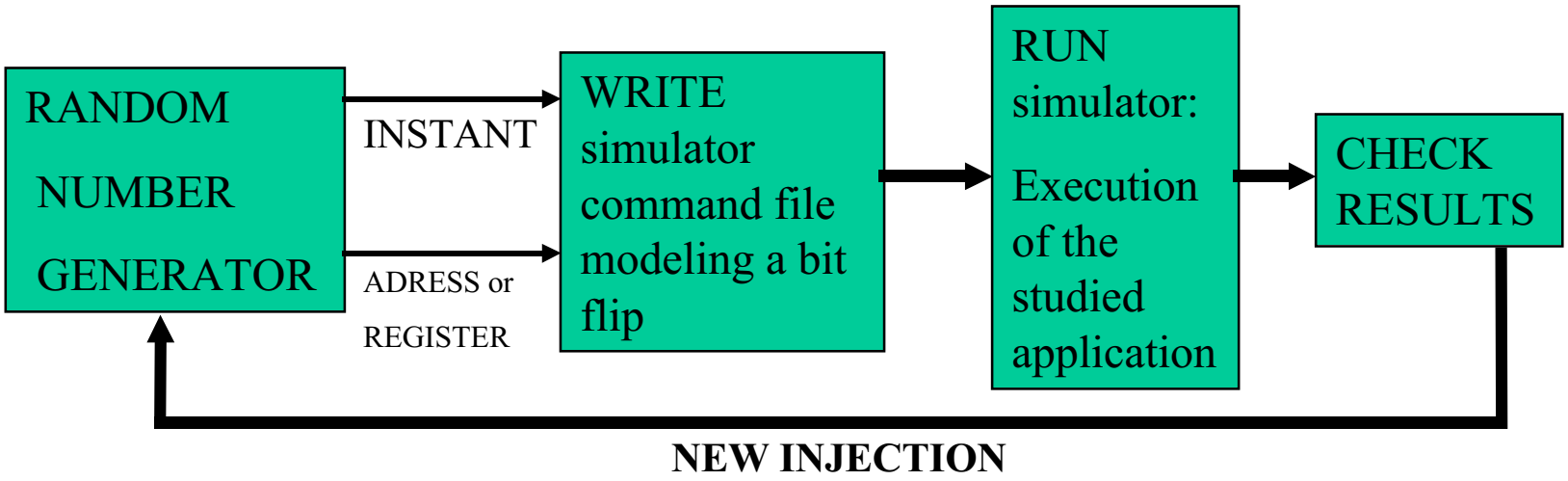
```
when t>=TOTAL_TIME {"Lost Seq"; quit}

when t>=INSTANT {TARGET=TARGET^XOR_VALUE; cont}

b end_cma {Print progarm output results; quit}

run
```

- the values of INSTANT and TARGET are instanciated by a Testbench



Software based upset injection strategy (cnt'd)

- Writing command files to inject a bit flip
- C-language Testbench:



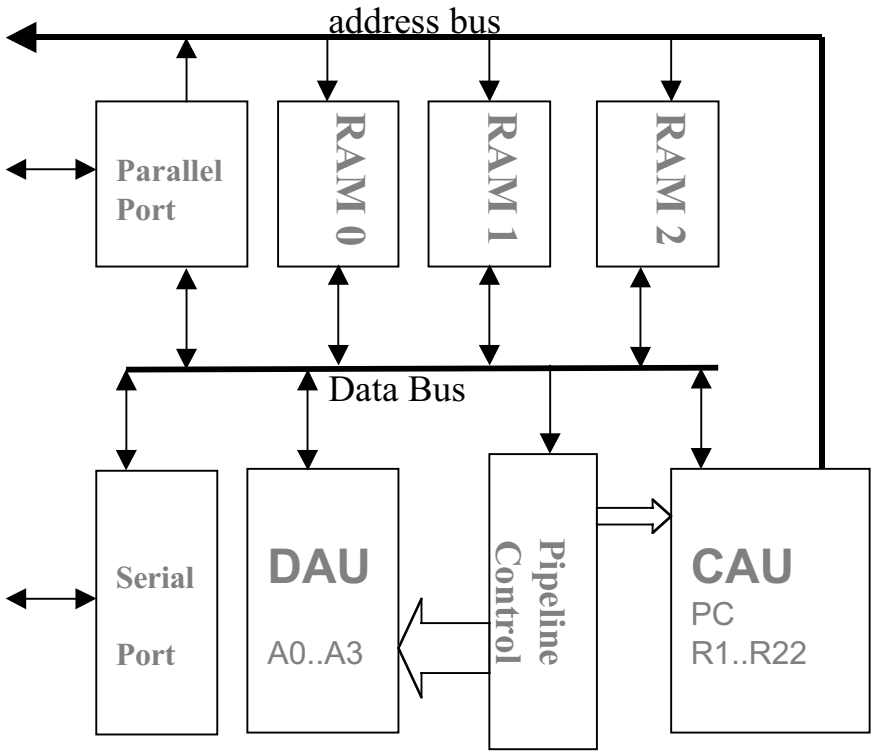
Software based upset injection strategy (cnt'd)

Main drawbacks

- Long simulation time
- Partial injection in particular processor blocks (i.e. pipeline)
- "False" tolerance of hardware control registers

2. A case study: the DSP32C

- RAMs: 2KB
- Address bus: 3 Bytes
- Data bus: 4 Bytes
- CAU: Arithmetic, Logic operations & program flow control
- DAU: Floating point operations. Four stages pipelined
- ~50.000 bits in DSP



A case study: the CMA equalizer

- **Constant Modulus Equalizer** (1280 bytes of code, 1381598 clock cycles, 133 float inputs)

- Upset target area:
- RAM0 (Output array)
- RAM1 (Global Variables, Input array)
- RAM2 (Stack)
- Registers (Rx, PC, Ax, …)

Non perturbed area:

- Code in the External Memory

A case study: Faulty behavior classification

- Fault injections sessions were performed using “d3sim”

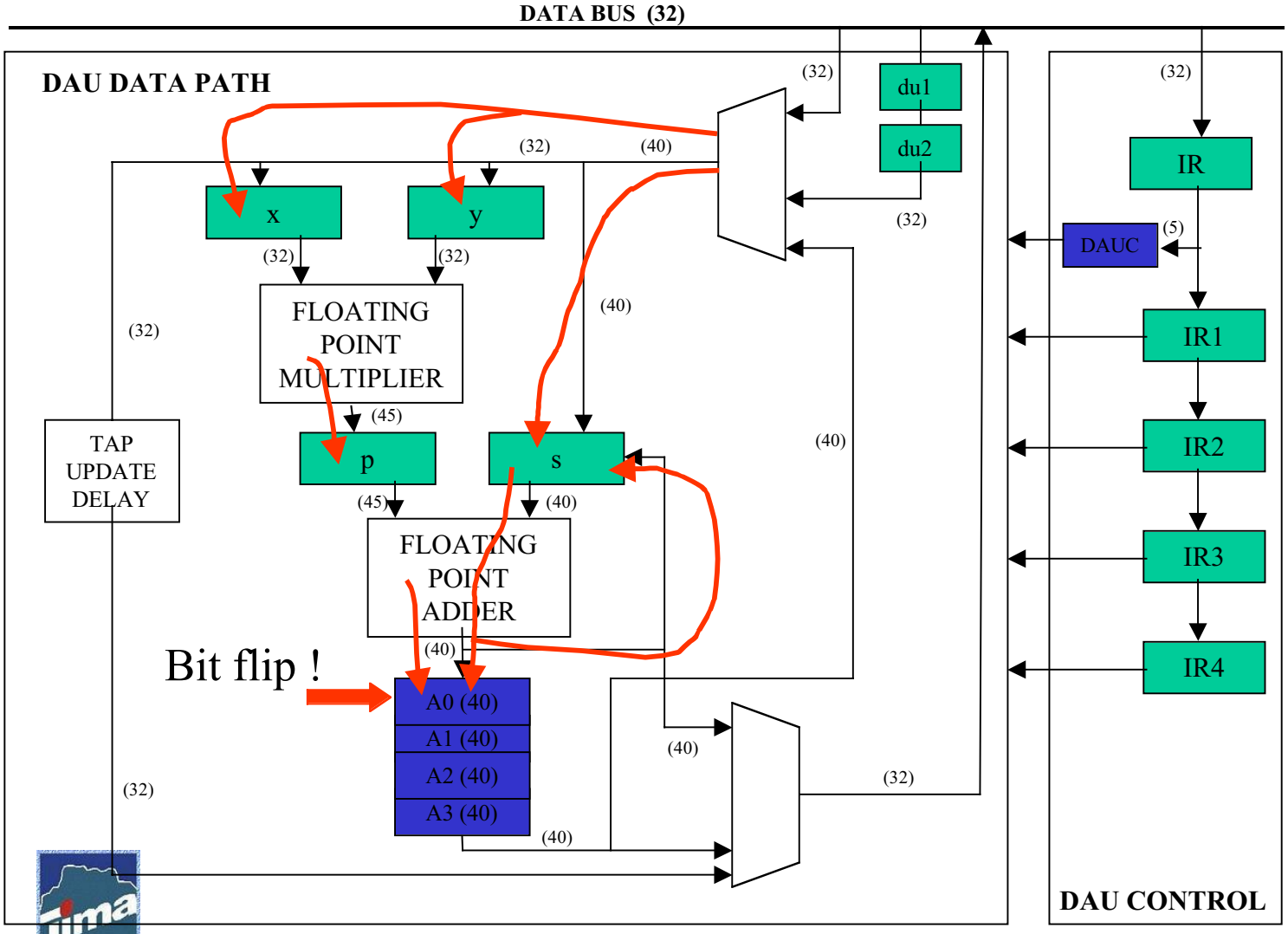
- Faulty behavior are classified as:
  - CMA error: final result different that the expected one
  - LS error: execution has not finished after expected cycles
  - Halted: processor mechanism to indicate some exception

2.a) Upset injection results

	#injected	CMA error	LS error	Halted	Total errors
RAM0	16253	1571	0	0	1571
RAM1	16262	3144	0	0	3144
RAM2	16617	85	8	0	93
Ax registers	120	!!!! 0	0	0	0
Rx	572	37	32	0	69
PC	15	6	4	0	10
Other registers	249	0	0	1	1
TOTAL	50088	4843	44	1	4888

$$\tau_{in} = 0.097 \quad \text{errors / upset}$$

Upset injection results (cnt'd)



Targeted registers

Non-targeted registers

Example

```
a0 = b
c = a0 + d*e
```

Effectless

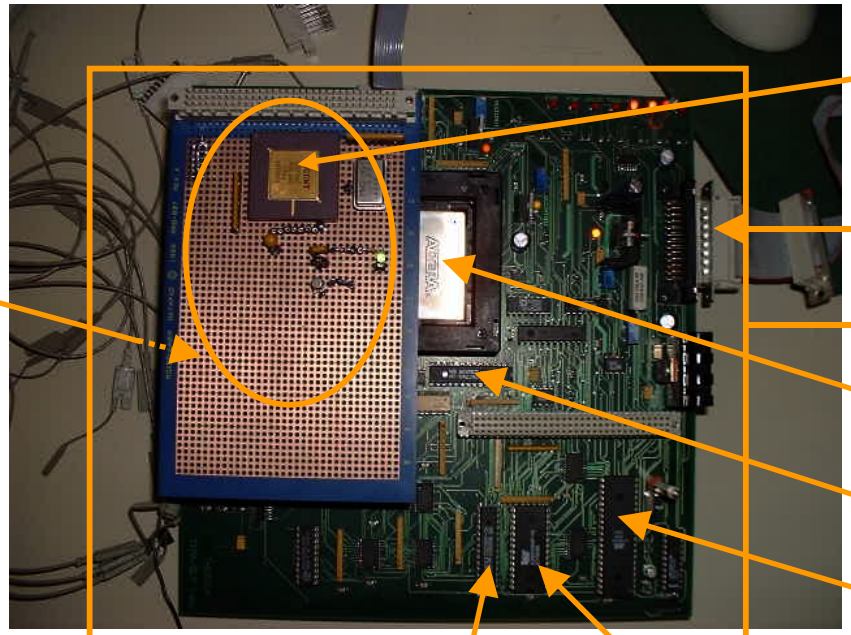
2.b) Radiation ground test

- Static test to obtain the underlying SEU cross-section of the DSP32C

- Dynamic test: run the CMA equalizer while exposing the DSP32C under radiation

Radiation ground test experimental set up: the THESIC+ system

- Using THESIC+ tester



daughterboard
with DSP32C & clock

Connection to PC

motherboard

FPGA

MMI (shared memory)

80c51

EEPROM for 80c51 programs

External RAM for the 80c51

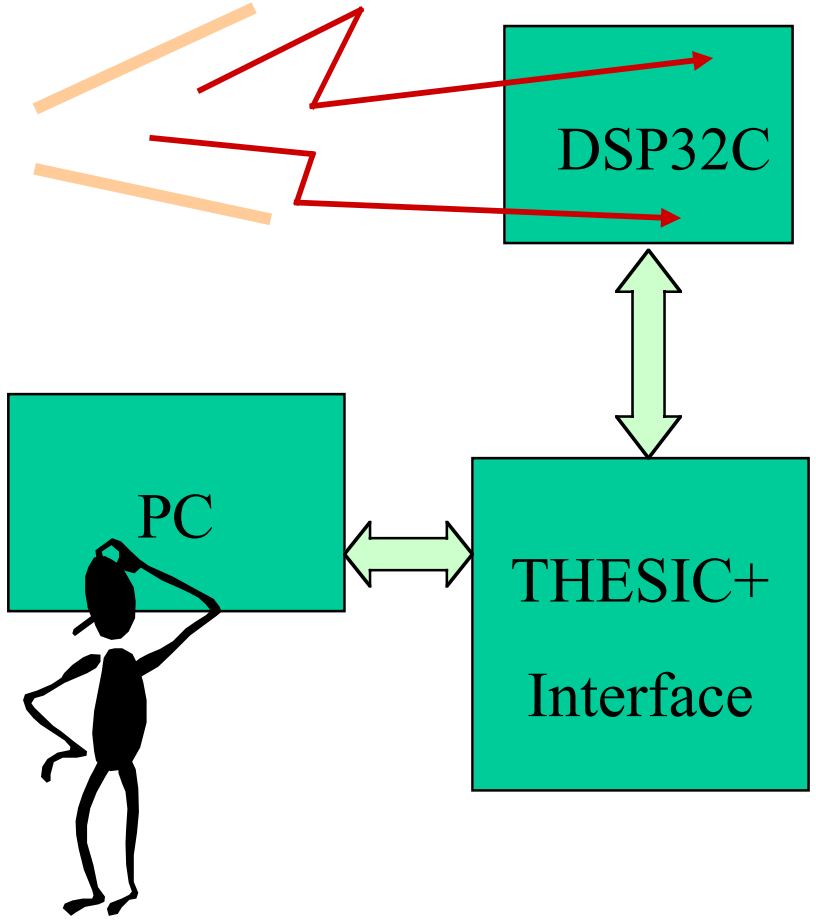
EEPROM's for DSP32C programs

Radiation ground test : radiation conditions

- Vacuum chamber
  - Californium 252
  - LET: 20 - 40 MeV
  - Flux: ~ 280 Particles/s
- PC with Terminal Interface
- THESIC+ test system
- DSP32C daughterboard

Radiation ground test: static test strategy

- Exposing the DSP to a particle beam
- Observing changes in memory areas (internal RAM 's and registers)
- Deriving the device static cross-section



Radiation ground test: static test strategy (cnt'd)

Main features:

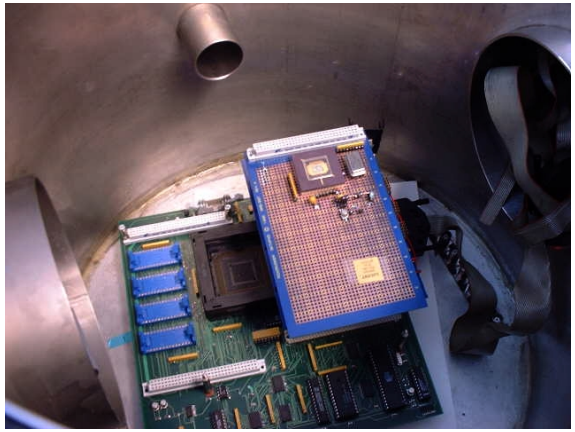
- Minimizing the operations of the processor under radiation
- Maximizing real exposure time of studied memory areas

	DSP32C	80c51
1. Write a pattern in the analyzed memory area locations	X	
2. Create a "mirror" vector		X
3. Read analyzed memory area and copy it in the MMI (shared memory) of THESIC+ board	X	
4.Compare the MMI content with the mirror vector. If differences exist, update the mirror and indicate upset occurrence parameters to the user's interface.		X

2.c) Final goal: error rate prediction

- SEU static cross-section

$\sigma_{SEU} = 2.7*10^{-3}$ upsets/particle



- Upset injection session

$\tau_{in} = 0.097$ errors/upset

Final goal: error rate prediction (cnt'd)

- **Predicted CMA error rate:**

$$\tau_{SEU} = \sigma_{SEU} * \tau_{in} = 2.7*10^{-3} \text{ x } 0.097 = \underline{2.6*10^{-4}} \text{ errors/particle}$$

- **Measured error rate for the DSP32C while running CMA under Cf252**

$$\tau_{SEU} = \underline{3.38*10^{-4}} \text{ errors/particle}$$

Experiment with a particle accelerator

Fault Injection Error Rate

$$\tau_{FI} = \frac{\#errors}{\#Injected\ \ faults}$$

Device	Error Rate (FI)
RAM 0	83.33×10^{-3}
RAM 2	125×10^{-3}
Registers	34×10^{-4}

SEU Static cross-section

$$\sigma device = \frac{\#upset}{Fluency}$$

Device	Cross-Section
RAM 0	2.6×10^{-3}
RAM 2	1.9×10^{-3}
Registers	1.3×10^{-4}

SEU Error Rate (Predicted)

$$\tau_{SEU} = \sum_i \sigma_{device}(i) \times \tau_{FI} = 4.9 \times 10^{-4}$$

Experiment with a particle accelerator (cont'd)

Radiation testing results: 111 application errors observed

- Ion Chlorine 156
- Fluency 2.54×10^5 $\left[\frac{\#\ particles}{cm?}\right]$
- Flux 8×10^2 $\left[\frac{\#\ particles}{cm? \times \sec}\right]$

SEU Error Rate (Measured)

$$\tau device = \frac{\#upset}{Fluency} = \frac{111}{2.54 \times 10^5} = 4.3 \times 10^{-4}$$

Error Rate Predicted	Error Rate Measured
4.9×10^{-4}	4.3×10^{-4}

Conclusions

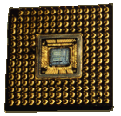
- SEU error-rates in processor-based architectures can be accurately predicted from software upset injection sessions and SEU cross-sections.

  - The robustness of different versions of a flight application can be evaluated off-beam.

- Using a software simulator to inject upsets:
  - decreases implementation cost and complexity
  - allows the study of the processor critical areas
  - allows automated upset injection sessions

But…

- needs a software simulator
- entails long simulation times (7 seconds / upset for the CMA running on the DSP32C)



Future Work

- Comparing predictions vs. error rates issued from radiation ground testing experiments using a particle accelerator facility
- Applying the approach to other complex processors and programs
- Qualification of telecommunication architectures in terms of error probability
- Improve the prediction accuracy by injecting upsets using VHDL models