HW-SW Interface for RAS in RISC-V Architectures

Daniele Rossi*, Nicasio Canino, Stefano Di Matteo, Sergio Saponara

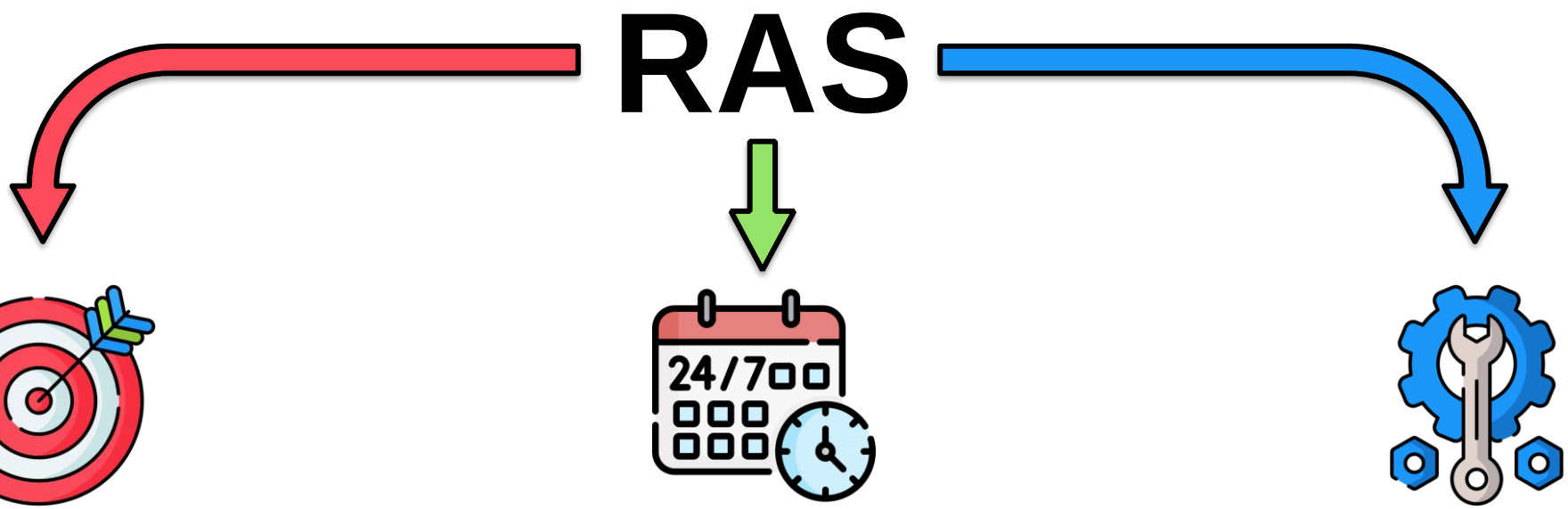
Department of information Engineering, University of Pisa, Italy

*Contact author - daniele.rossi1@unipi.it

Brief Outline

- Background
- Overview of the Designed Architecture
- Verification and Synthesis Results
- Test Environment on FPGA
- Conclusions

Reliability, Availability, Serviceability (RAS)



Reliability

Probability that the system produces correct outputs

Availability

Ability of the system to be available at any time

Serviceability

Ability of the system to provide information about the system failure occurred

How can the System RAS be improved?

Error Detection & Correction

Reliability → ***Correcting*** *the error*

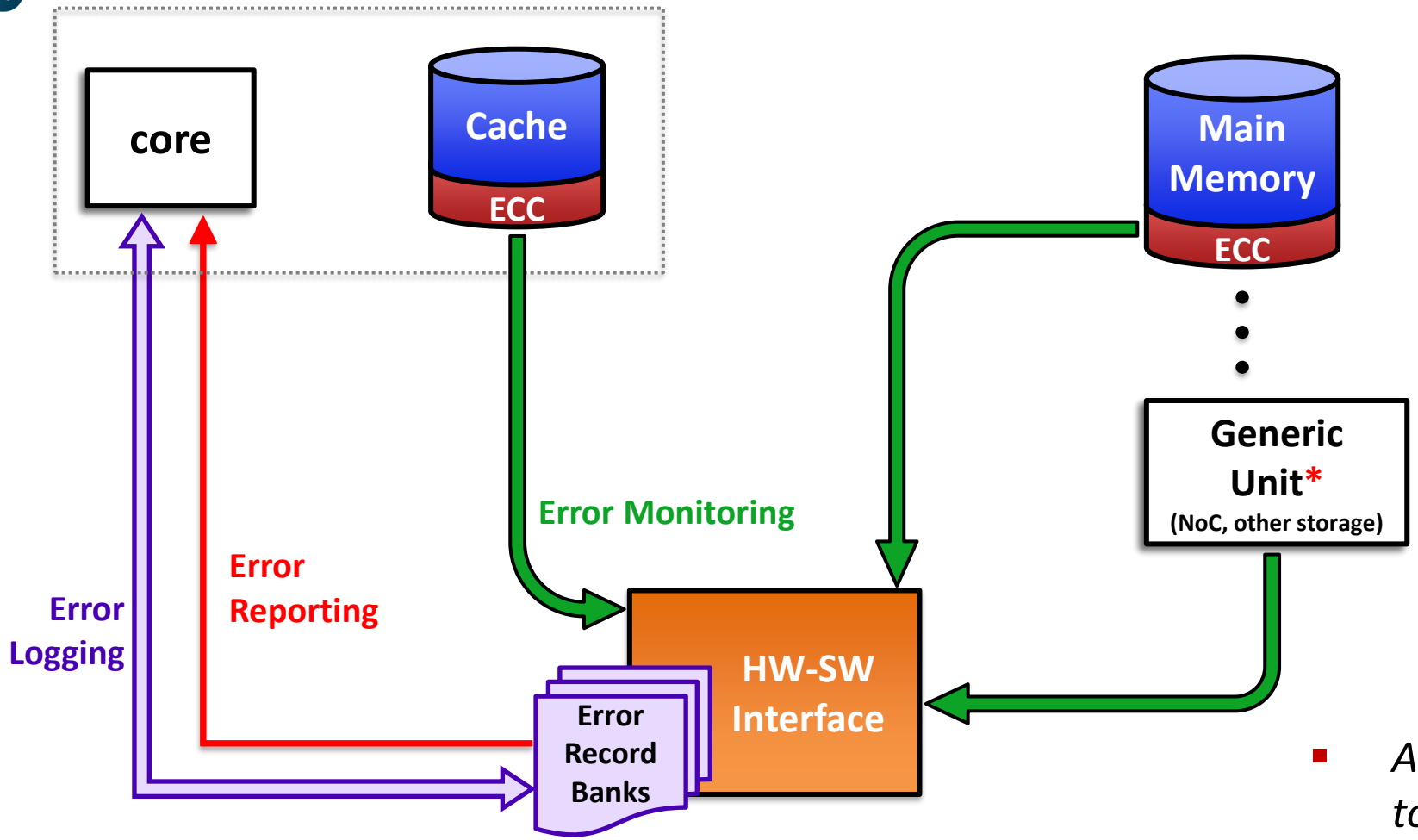
Availability → ***Preventing*** *the system to undergo a failure*

Error Logging & Reporting

Availability → ***Signalling*** *to system SW only* ***when needed***

Serviceability → ***Storing*** *relevant* ***error-related information***

HW-SW Interface for Error Logging and Reporting



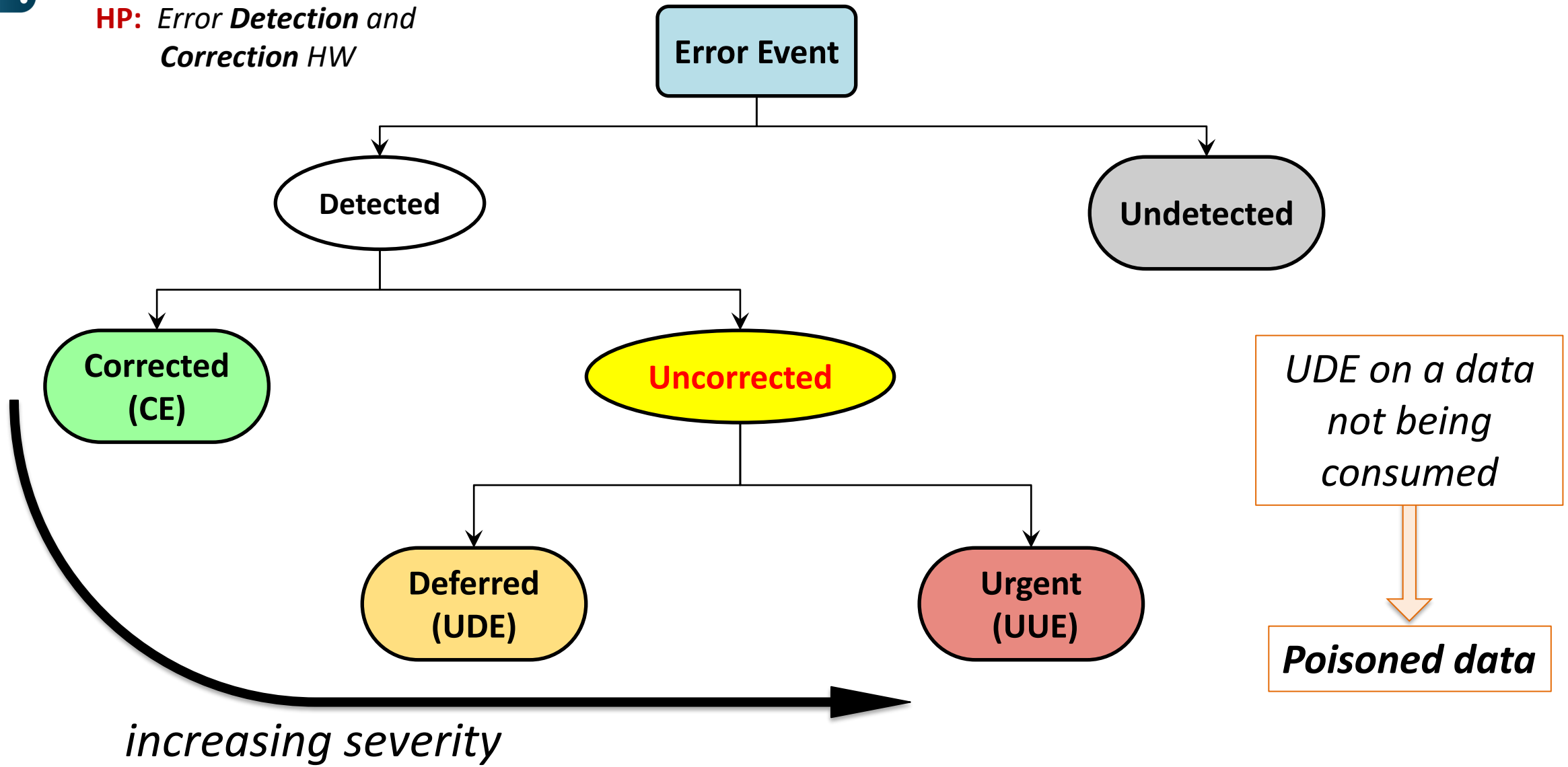
*must have Error **Detection** and **Correction** HW

- *Error-related data* ⟶ ***Error Record***

- *All **existing** ones are proprietary to **Intel**, **AMD**, or **Arm***
- ***No** consolidated solutions for the **RISC-V architecture***

Error Taxonomy

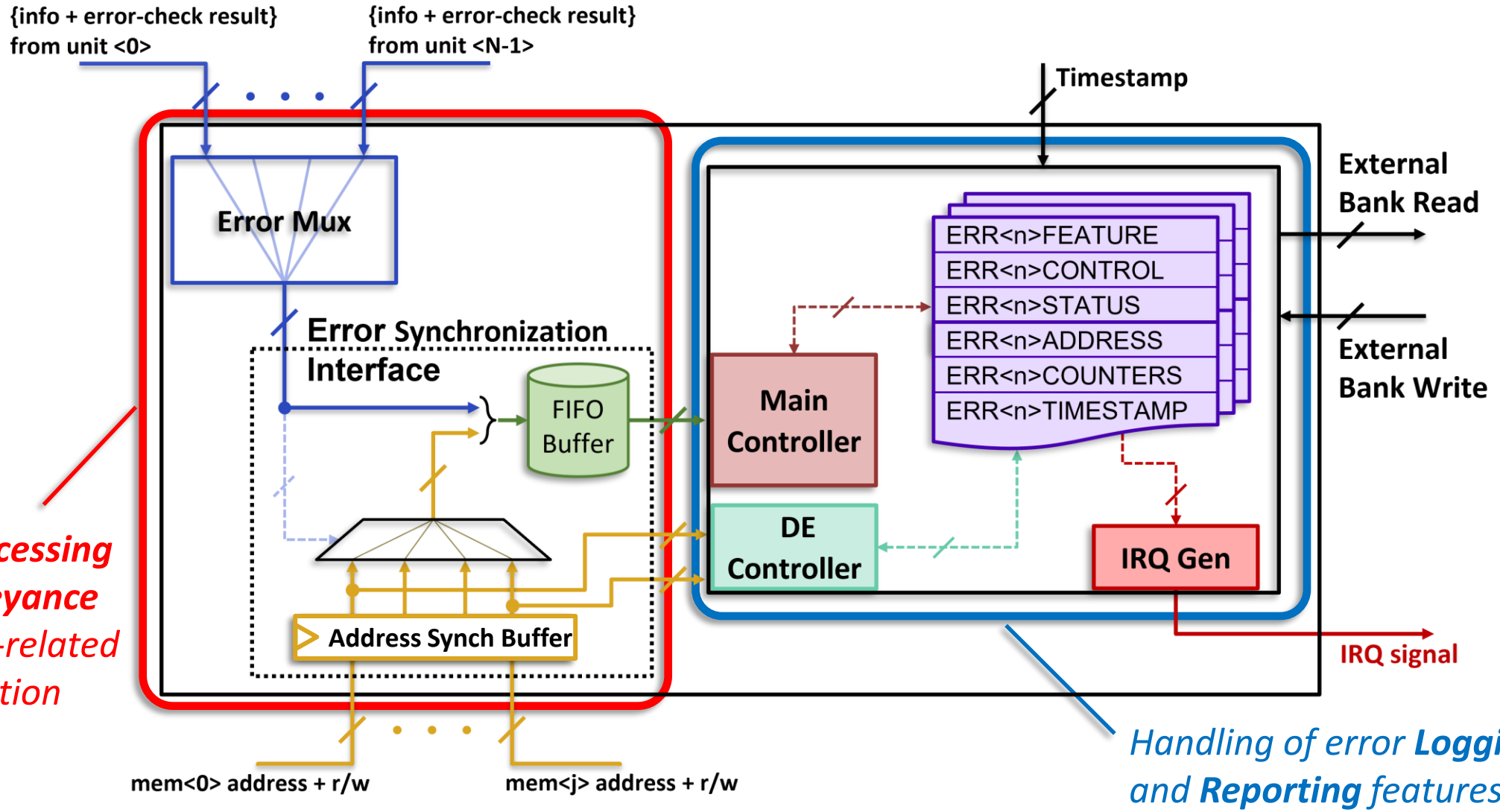
HP: *Error **Detection** and **Correction** HW*



UDE on a data not being consumed

Poisoned data

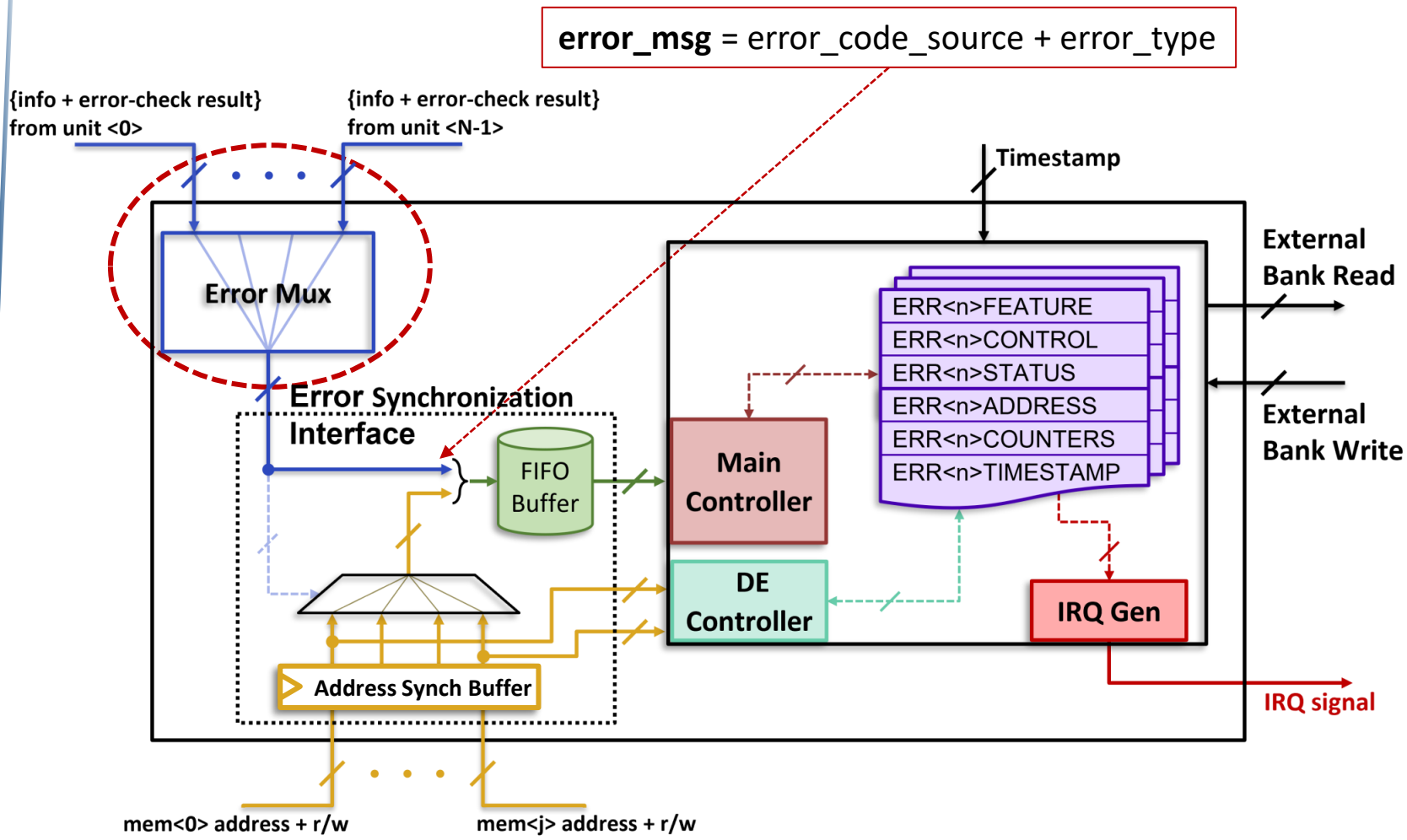
Error Logging and Reporting Architecture (ELRA)



Pre-processing *& **conveyance** of error-related information*

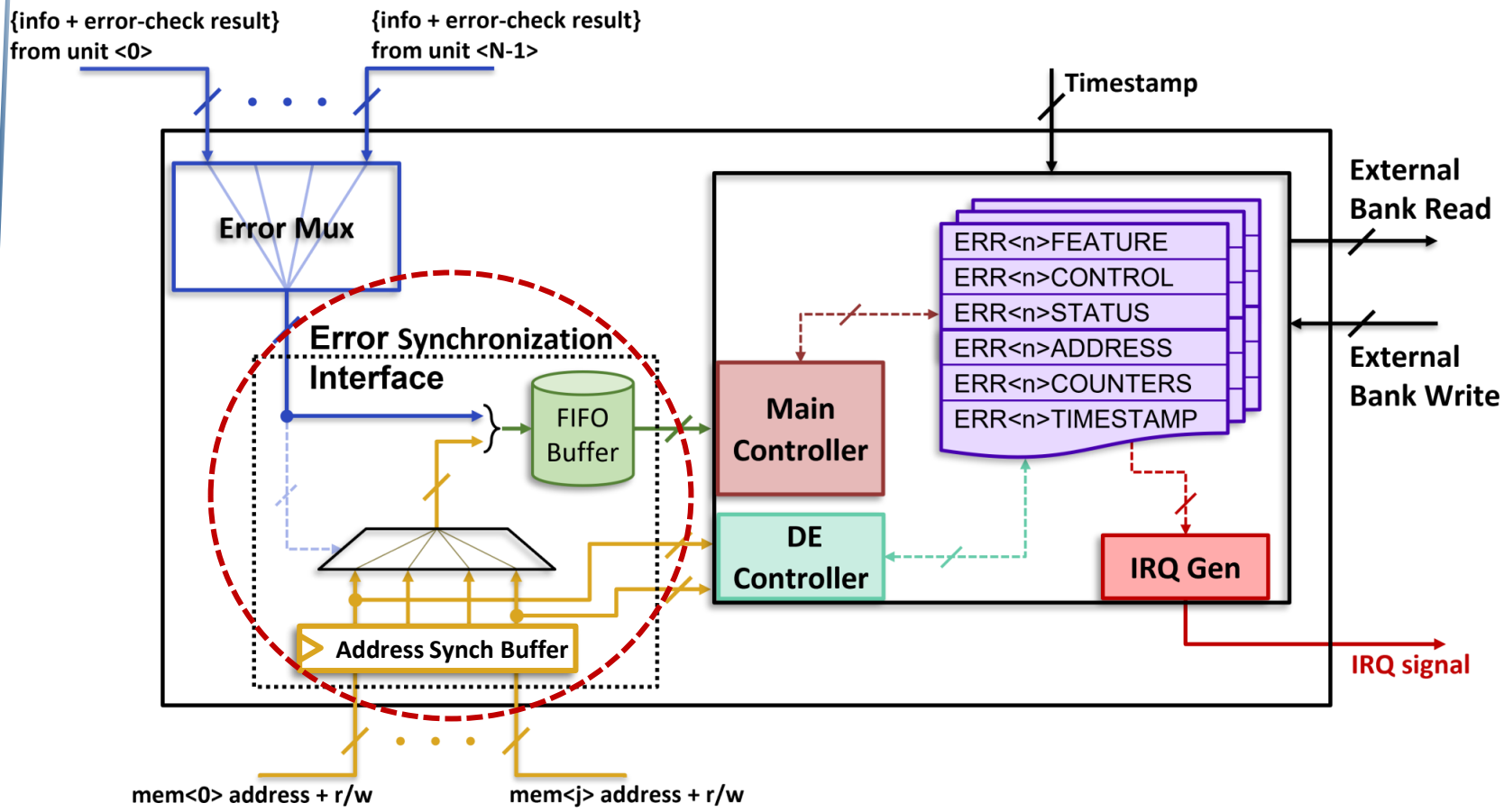
*Handling of error **Logging** and **Reporting** features*

Error Logging and Reporting Architecture (ELRA)



- **Error Mux** receives the error check signals generated by the ECC of the monitored HW units
- It handles the specific case and generates an **error message**
  - Error message: **Source ID, Error Type** (CE, UUE, UDE)

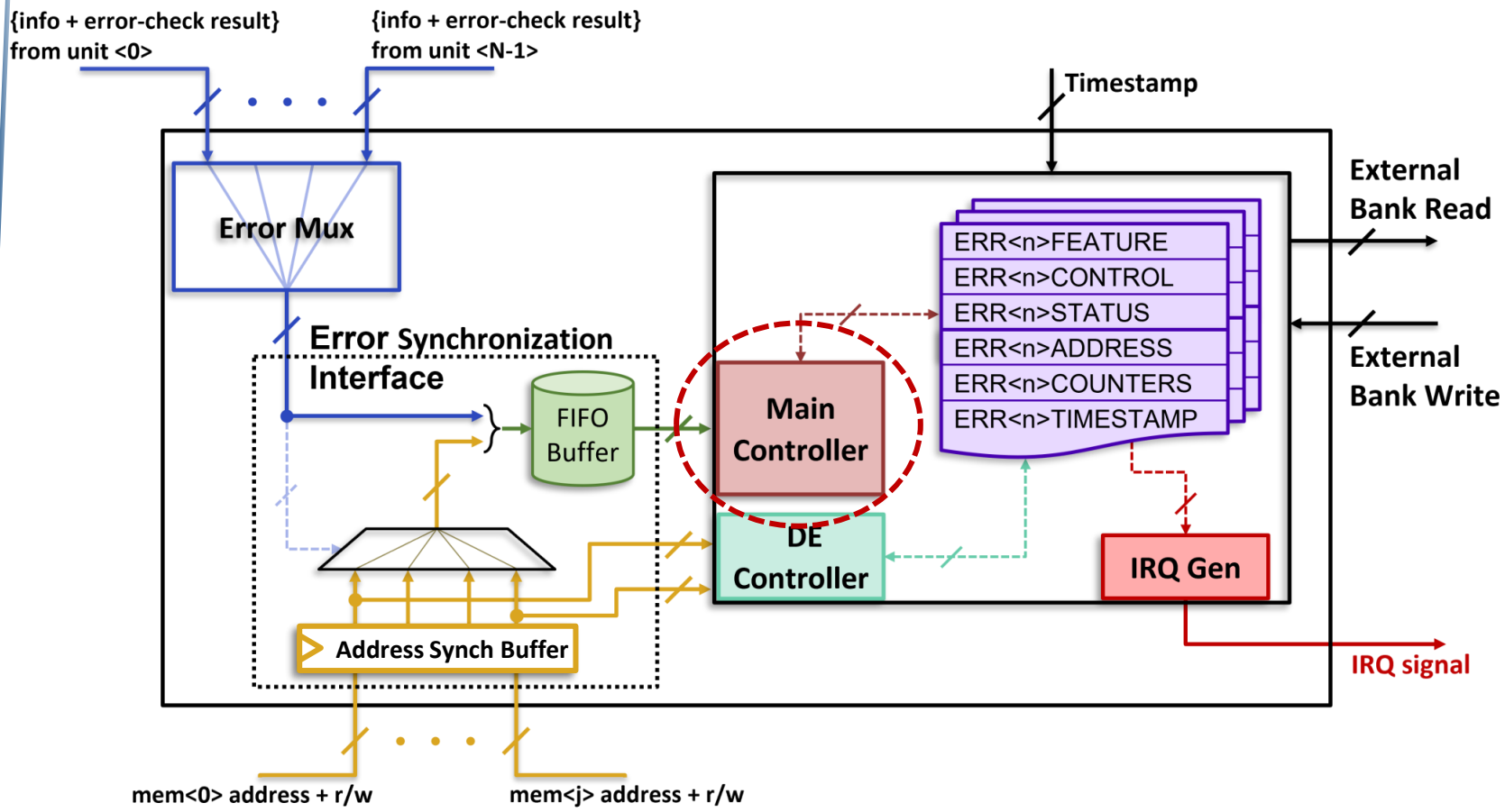
Error Logging and Reporting Architecture (ELRA)



- **Error synchronization Interface** synchronizes
  - **error message** coming from the Mux
  - **address of the erroneous location** coming from the Address Buffer

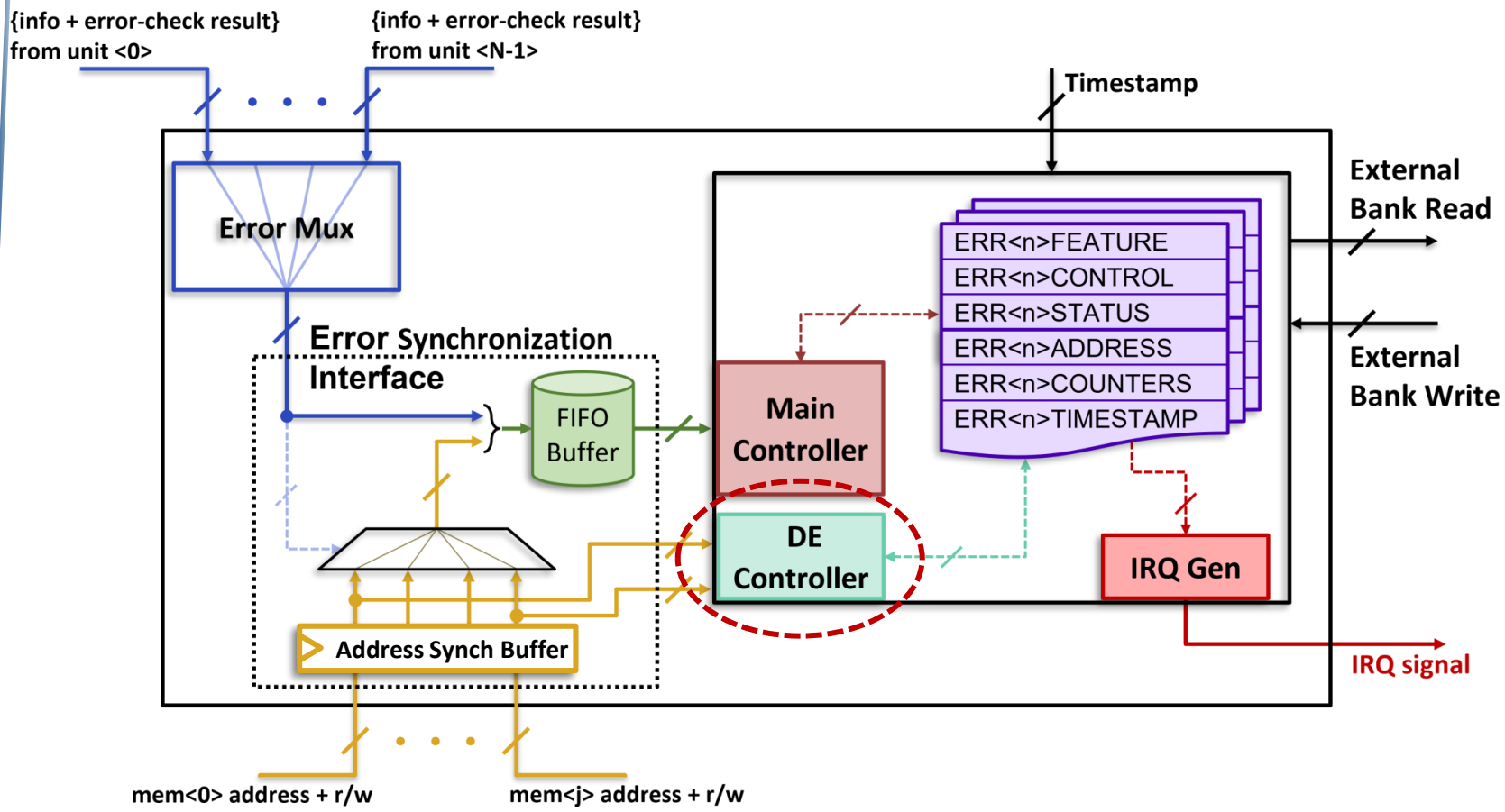
→ information is stored in a circular **FIFO buffer**

Error Logging and Reporting Architecture (ELRA)



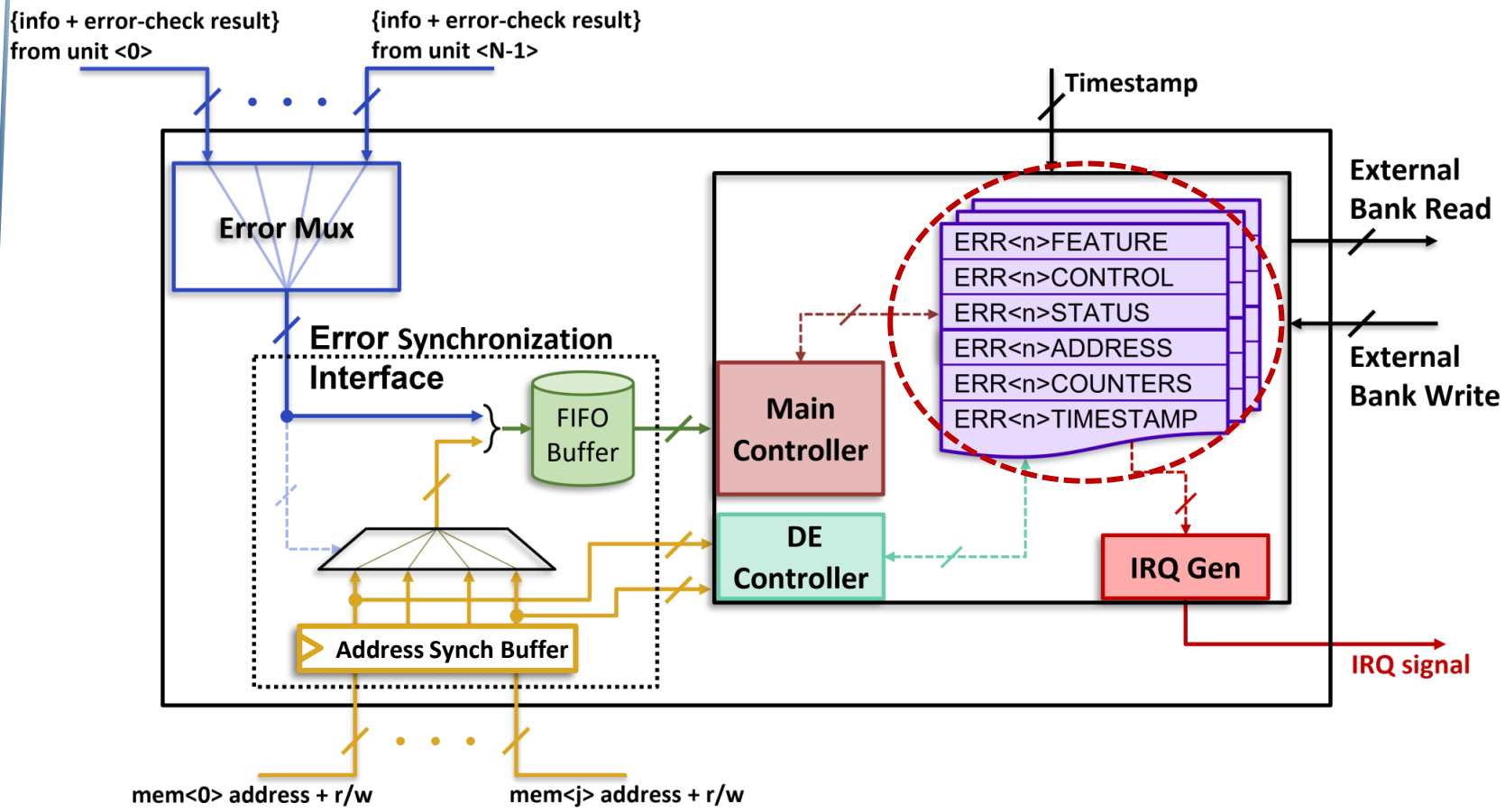
- **Main Controller**
  selects where to locate the new error record - fully associative cache-like approach

Error Logging and Reporting Architecture (ELRA)



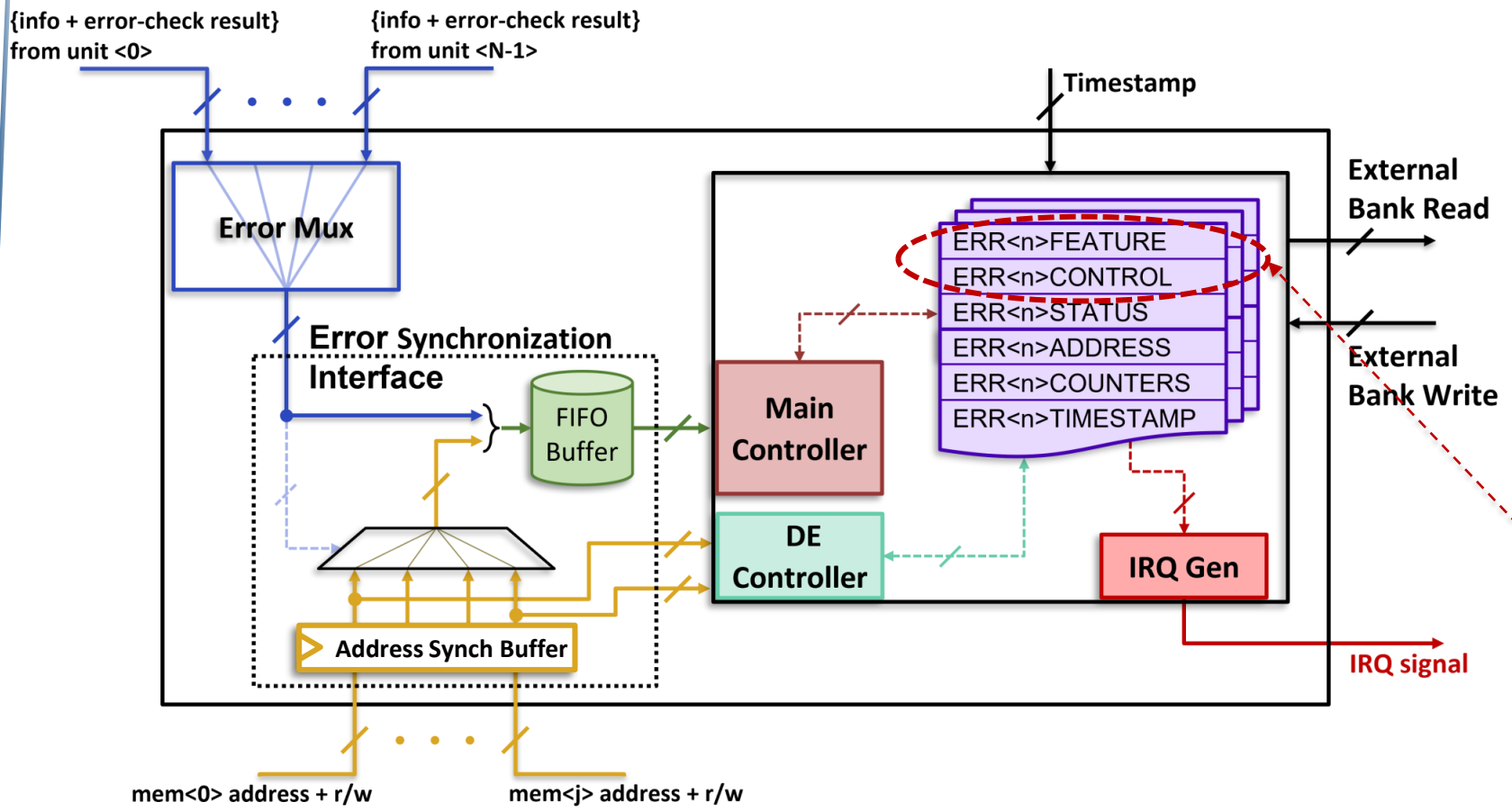
- **Main Controller**
  selects where to locate the new error record - fully associative cache-like approach
- **DE Controller**
  receives addresses and r/w signal and verify whether a UDE location is being accessed

Error Logging and Reporting Architecture (ELRA)



- **Error Record Banks**
  log an error record independently of the error source and error type, among all hardware units monitored

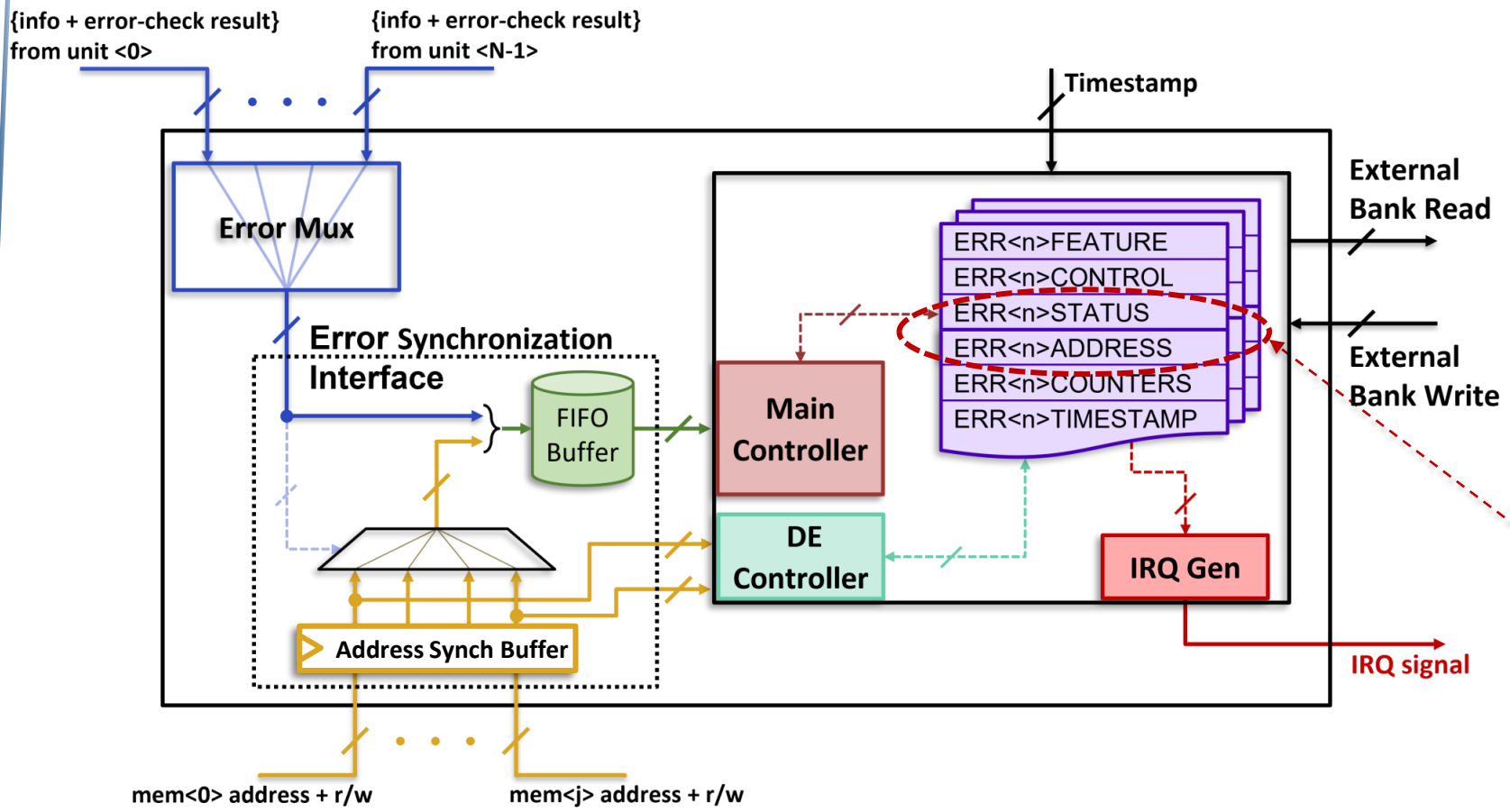
Error Logging and Reporting Architecture (ELRA)



- **Error Record Banks**
  log an error record independently of the error source and error type, among all hardware units monitored

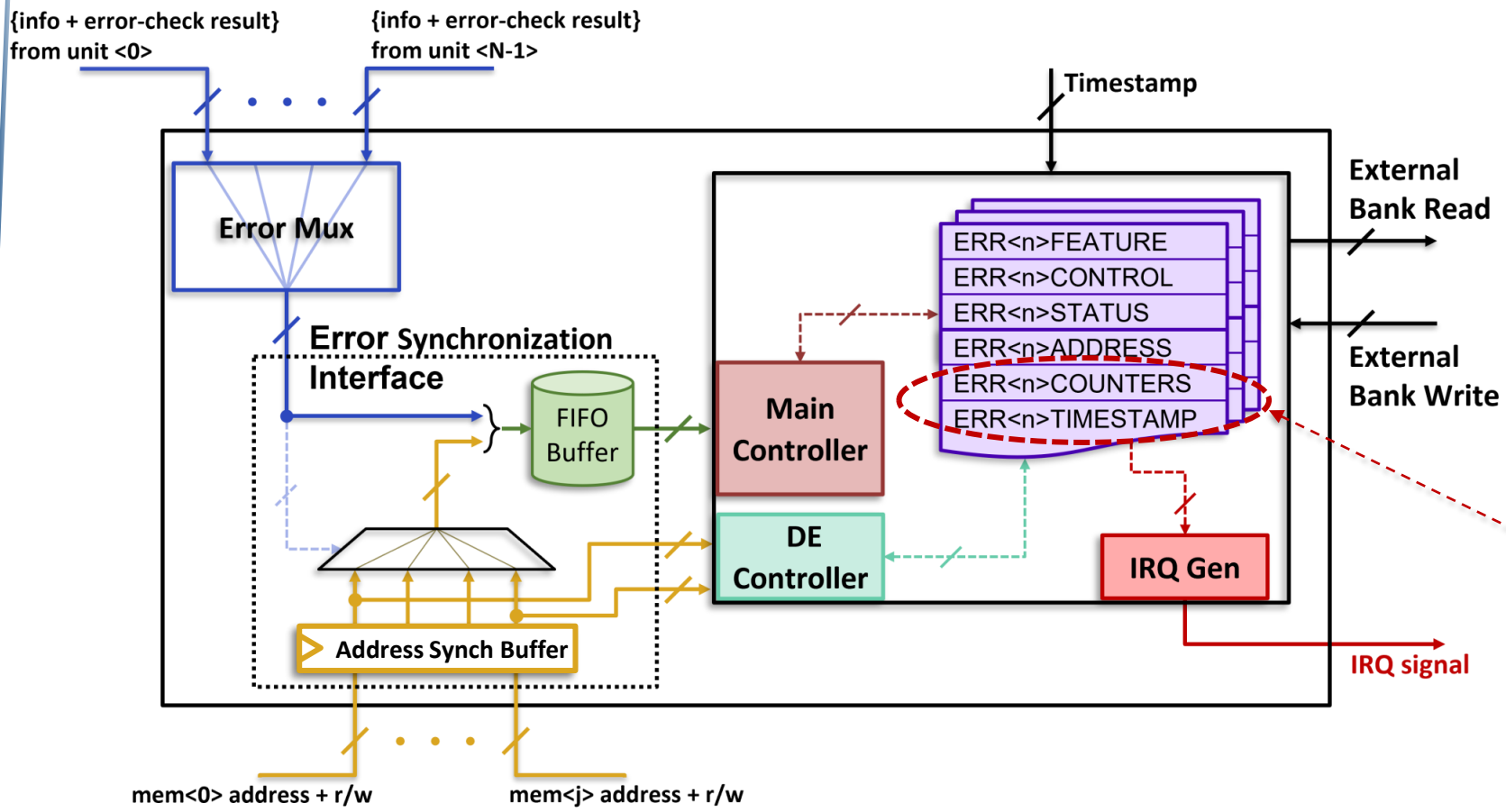
- **Feature & Control Regs**
  define main characteristics (set @ implementation stage) & store a set of enable signals for the different features (set @ runtime)

Error Logging and Reporting Architecture (ELRA)



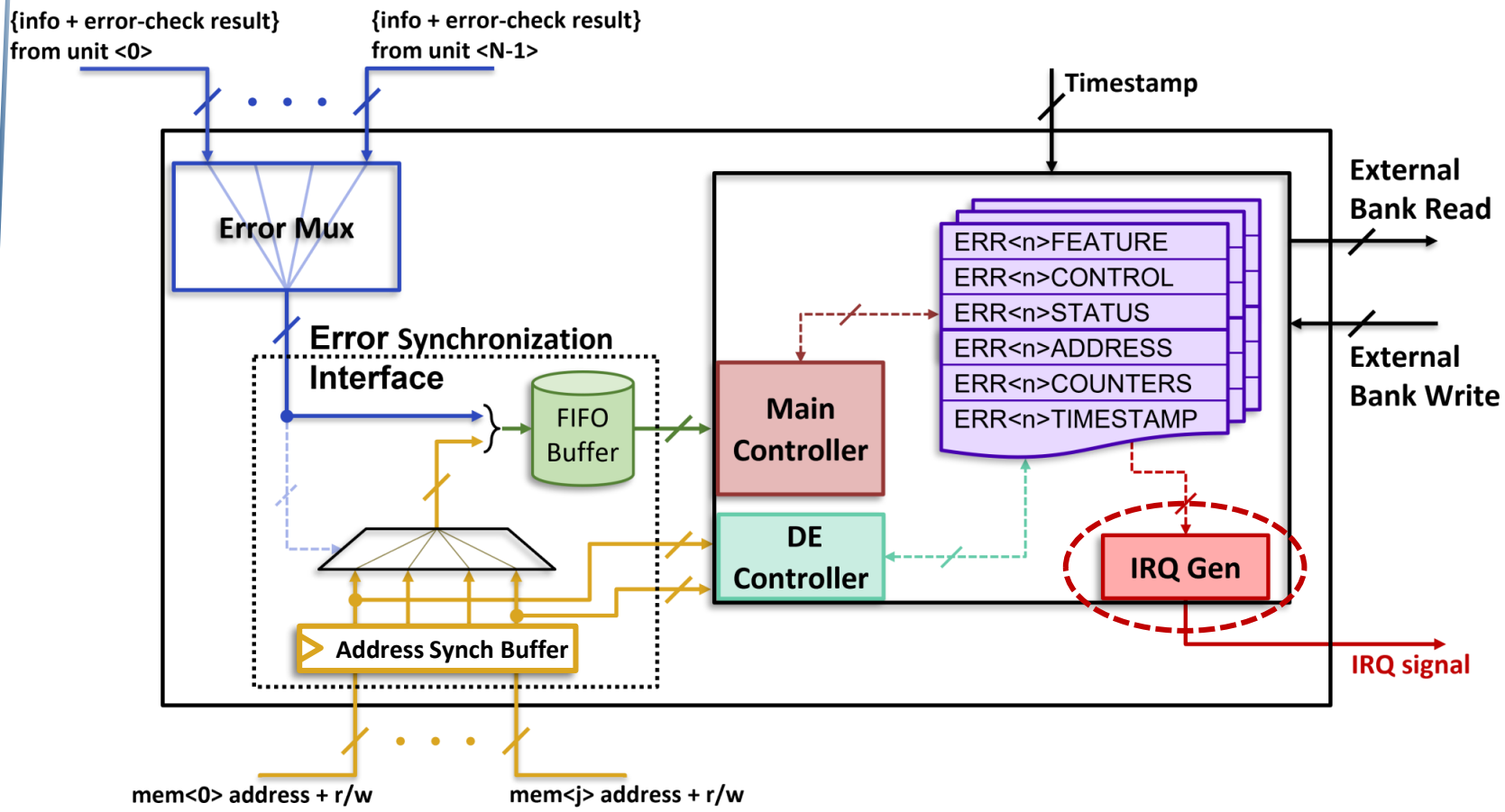
- **Error Record Banks**
  log an error record independently of the error source and error type, among all hardware units monitored
- **Status & Address Regs**
  store information on the error (error_msg, ...) & the address of the erroneous location

Error Logging and Reporting Architecture (ELRA)



- **Error Record Banks**
  log an error record independently of the error source and error type, among all hardware units monitored
- **Counters & Timestamp**
  counting number of corrected errors & timestamp of logged error

Error Logging and Reporting Architecture (ELRA)



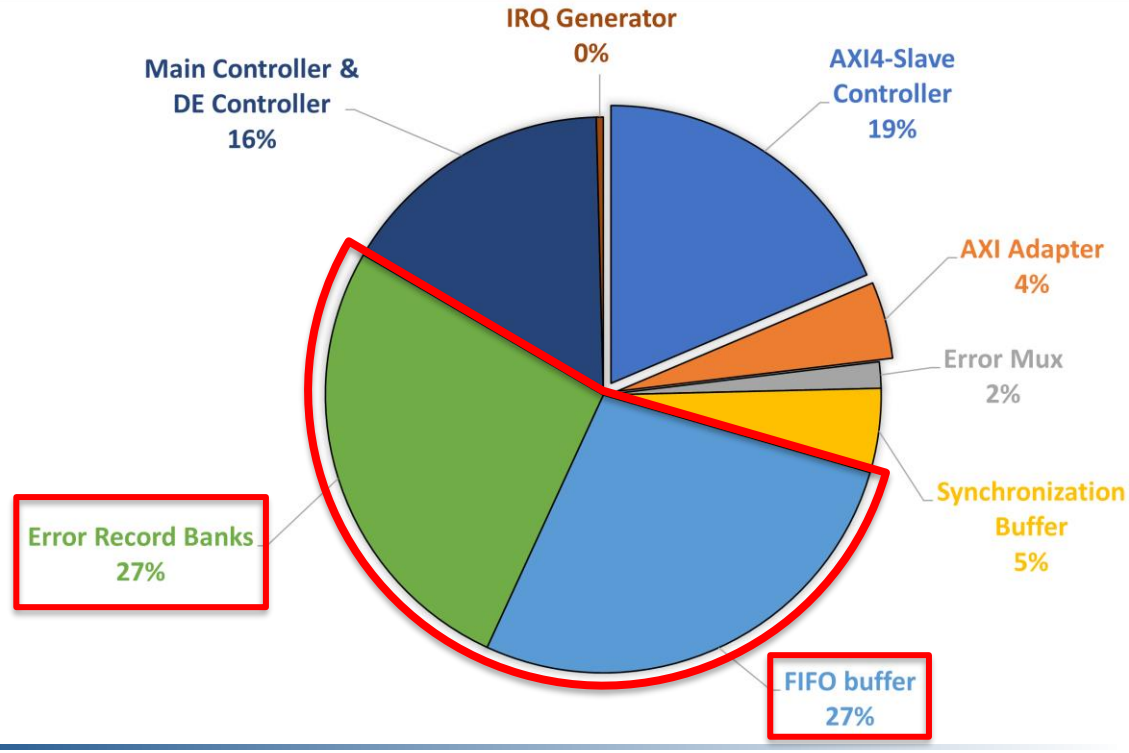
- **IRQ Generator** generates the *Interrupt Request* in different scenarios, depending on the features of the RAS Interface

Synthesis Results on Standard-Cell and FPGA

- **Configuration of the ELRA module:**
  - ***Two*** *Error Record Banks*
  - *Both* ***CE Counters*** *implemented (****7-bit*** *wide)*
  - ***Timestamp 32-bit*** *wide*
  - ***Address*** *bus* ***32-bit*** *wide*
  - ***4*** *FIFO Buffer* ***entries***
  - ***AXI4*** *Memory-Mapped Interface included*

Standard-Cell results:

Technology	Complexity $[kGE]$	Area $[\mu m^2]$	Maximum f_{clk}	P_{din}/f_{clk} $[mW/GHz]$	$P_{leak}/area$ $[nW/\mu m^2]$
$45nm$ (*Nangate Open-Cell*)	7,47	5.962,39	$1{,}2GHz$	5,06	18,11
$7nm$ (*Artisan*)	7,41	569,22	$5{,}5GHz$	1,92	39,71



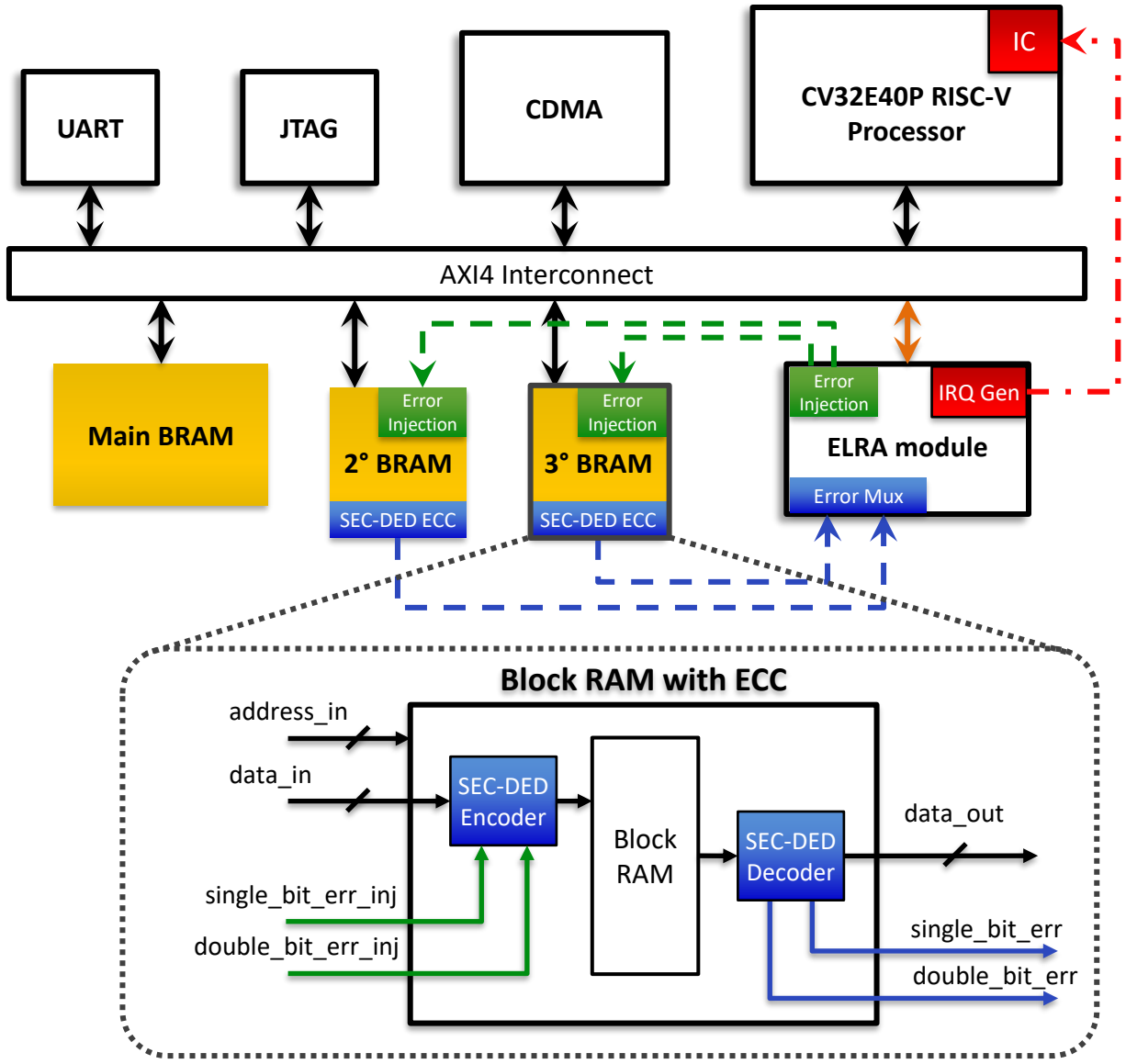
FPGA results:

- *Xilinx* ***Zynq Ultrascale+ MPSoC***

Maximum f_{clk}	LUT	FF	CARRY 8	Block RAM	DSP
320MHz	*459*	*433*	*8*	*0*	*0*

Test Environment Implemented on FPGA

- **Test SoC modules:**
  - *RISC-V CV32E40P core* **+** *Main Block RAM (BRAM)*
  - *Central DMA (CDMA)* **+** *ECC-protected Block RAMs* ***emulate L1 and L2 caches***
  - *JTAG to* ***load C*** *code for testing*
  - *UART to* ***give feedbacks*** *on user terminal*
- **ELRA-to-system interconnections:**
  - ***Error Logging***
  - ***Error Reporting***
  - ***Error Monitoring***
  - ***Error Injection***



Conclusions

Summary of the ELRA module results:

- **Design** of an **HW-SW interface** for RISC-V architecture
- **Verification** and **validation** on FPGA
- **High configurability** of Error Logging and Reporting **at synthesis level**
- **Flexible area occupation**
- **Non-limiting frequency** on Standard-Cell technologies