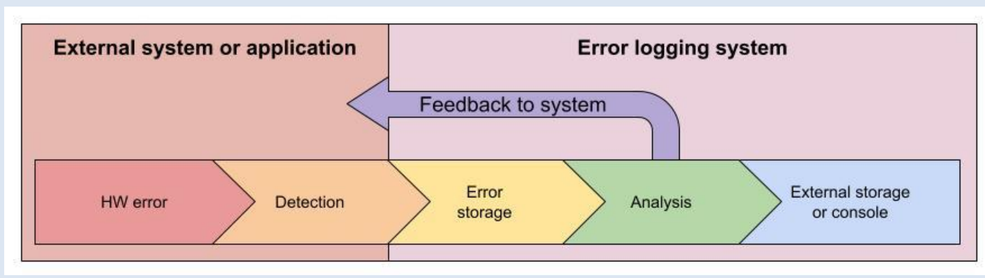
The RERI-Lite error logging framework

Michiel Koenderink, Bruno Endres Forlin, Gerard Rauwerda, Marco Ottavi
CAES Group – University of Twente, the Netherlands

Introduction

RISC-V cores that are employed in space environments have a higher risk of hardware errors.

Detecting and resolving errors as fast as possible can prevent **errors** from **propagating through the system** and causing new errors.



One general **Error Logging System (ELS)** to detect, log, analyse and resolve all error information can improve the reliability of a system.

RERI standard

The RISC-V organisation has specified **a format for a bank with error records** that can be used to store information about detected errors.

Offset	Name	Size	Description
0	vendor_n_imp_id	8	Vendor and implementation ID.
8	bank_info	8	Error bank information.
16	valid_summary	8	Summary of valid error records.
24	Reserved	32	Reserved for future standard use.
56	Custom	8	Designated for custom use.
64 + 64 * i	control_i	8	Control register of error record i.
72 + 64 * i	status_i	8	Status register of error record i.
80 + 64 * i	addr_info_i	8	Address-or-info. register of error record i.
88 + 64 * i	info_i	8	Information register of error record i.
96 + 64 * i	suppl_info_i	8	Supplemental information register of error record i.
104 + 64 * i	timestamp_i	8	Timestamp register of error record i.
112 + 64 * i	Reserved	16	Reserved for future standard use.

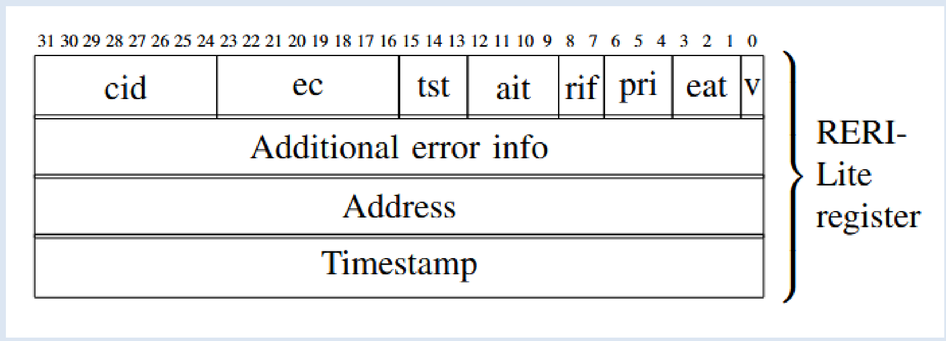
The standard RERI record format is large and **inconvenient for embedded systems** with limited resources.

Adjusted RERI variants will be difficult to integrate with other systems like an ELS.

Our proposal: RERI-Lite

A new, smaller and more flexible **standard** error record format **dedicated to embedded systems**.

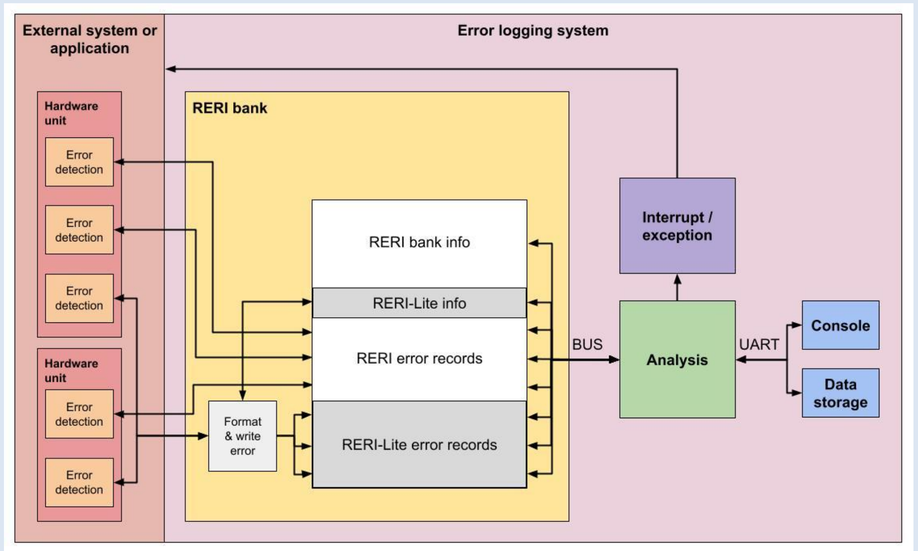
The RERI-Lite format uses four 32-bit registers. The top register is always used, while the other three registers are optional.



The fields use the **standard RERI formats** to keep compatibility with the standard RERI format.

Error logging framework

The RERI specification does not define a framework around the RERI error bank, so a new **error logging framework** was made for the RERI-Lite error record bank.



Implementation results

The framework design was synthesized and implemented targeting an ARTY A7-35T.

- The decrease in area usage can **improve** the **signal speed** and **lower** the risk of new **hardware errors**.
- Error record interactions and transmissions require **less clock cycles**.

Logic type		Amount of RERI-Lite records				
		1	4	8	16	32
Slice LUTS	Total	216	451	1929	3468	7226
	As logic	215	447	1921	3452	7194
	As memory	1	4	8	16	32
Slice registers	Total	254	534	2041	2638	6769
	As flip flop	248	528	2035	2632	6763
	As latch	6	6	6	6	6
Muxes	F7	0	0	50	20	122
	F8	0	0	8	10	8

Records	RERI			RERI-Lite		
Amount	Bytes	RAMB36	Utilised	Bytes	RAMB36	Utilised
1	64	4	8%	16	1	2%
4	256	16	32%	64	4	8%
8	512	32	64%	128	8	16%
16	1024	-	-	256	16	32%
32	2048	-	-	512	32	64%
64	4096	-	-	1024	-	-

Process	Subfunction	Clock cycles	
		RERI	RERI-Lite
Monitoring errors	Detect error signals	3 + N*	3 + N*
Control error record	Write valid record	33.5	9.5
	Reset record	33.5	9.5
Analyse errors	Read error record	149 - 150	14 - 42
	Check error records	3.5	3.5
	One bus transaction	9	9
UART communication	Create UART transmission	1	1
	Transmit UART frame	8681	8681
	Transmit full error record	607639	86806 - 190972

Conclusions

- A **smaller** and more **flexible** error record format, called “RERI-Lite” was proposed to store error information.
- It is dedicated to **embedded systems** and its focus is to be easily **compatible** with the standard RERI error record format and other systems.
- An **error logging framework** around the error record bank was **designed and implemented** to monitor, control, analyse and export error information.

Further development

- Test the RERI-Lite framework in a radiation environment.
- Implement an extended analysis in the ESL framework.
- Provide error feedback to the main system for error recovery.

TRISTAN RISC-V

DCS Dependable Computing Systems

UNIVERSITY OF TWENTE.