UNIVERSITÀ DEGLI STUDI DI ROMA TOR VERGATA

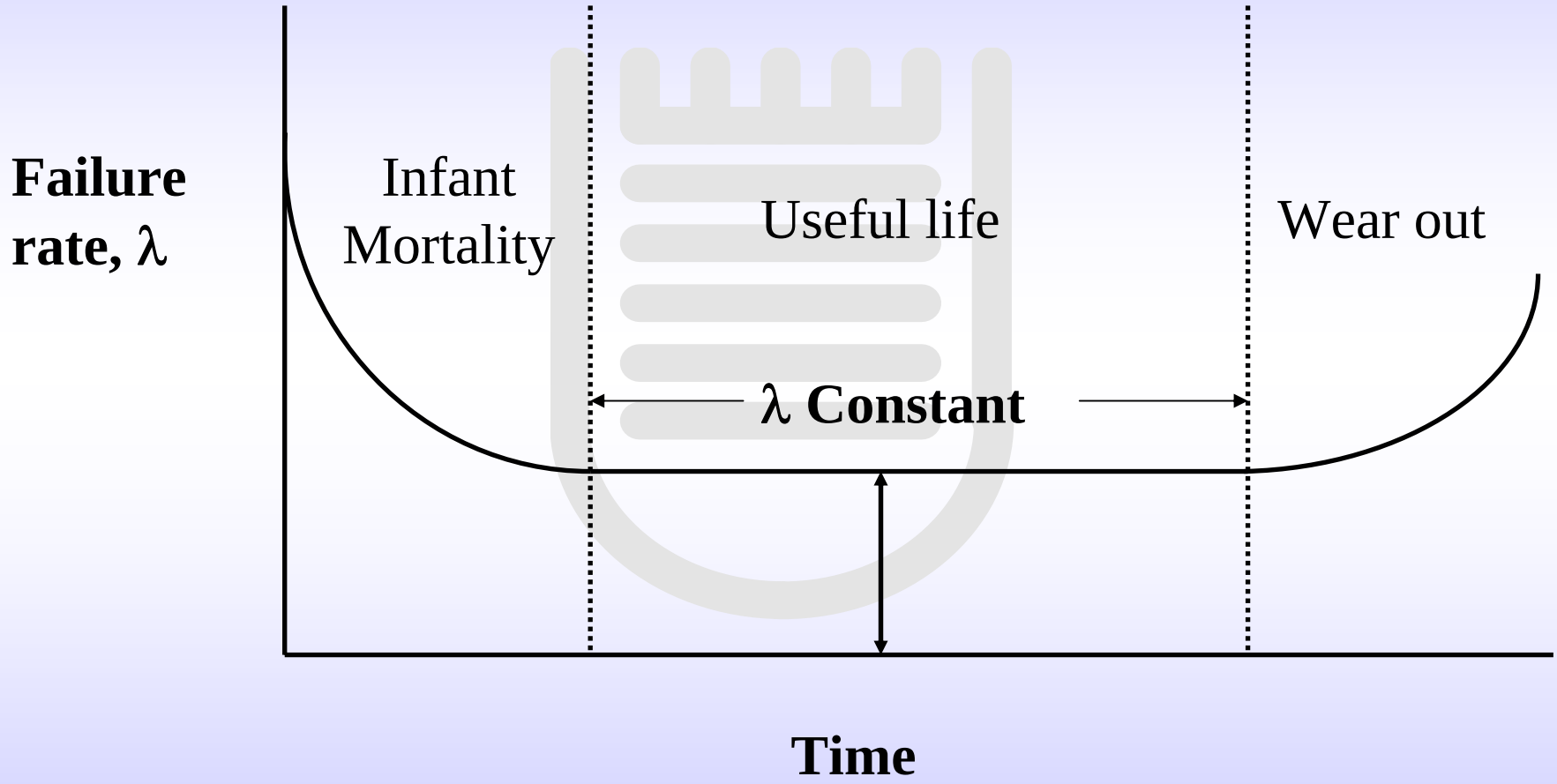
Dipartimento di Ingegneria Elettronica

Sistemi Tolleranti ai Guasti: Generalità ed Esempi Applicativi. Le Memorie di Massa

Svolgimento della Presentazione

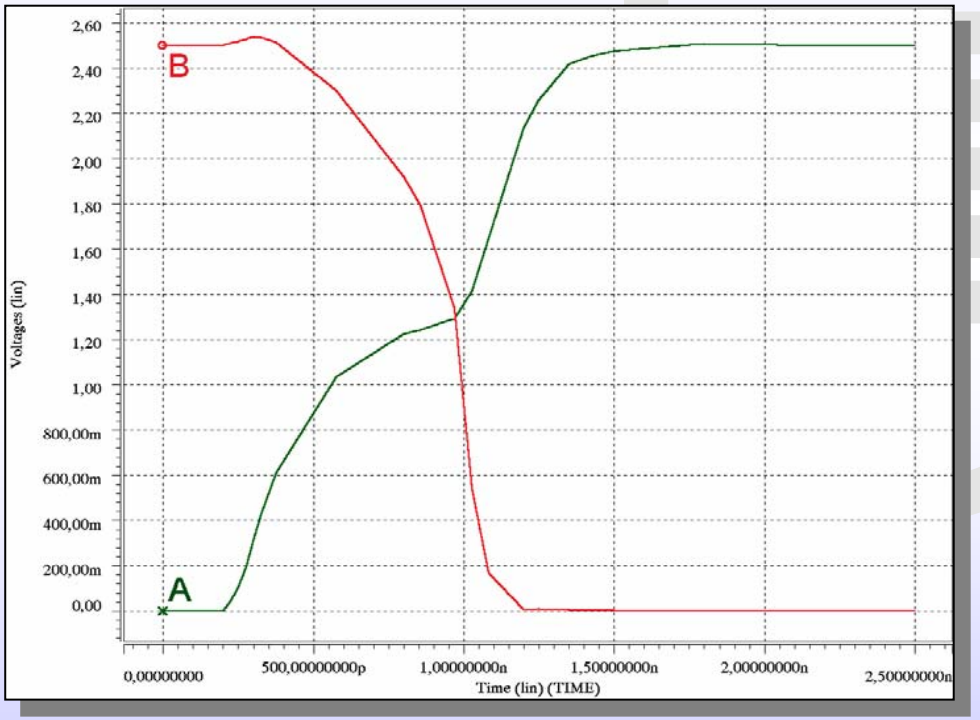
- Principi di base dei sistemi digitali fault-tolerant
- Progetto di una Memoria di Massa allo Stato Solido per applicazioni spaziali
- Fast – Prototyping della SSMM
- Conclusioni

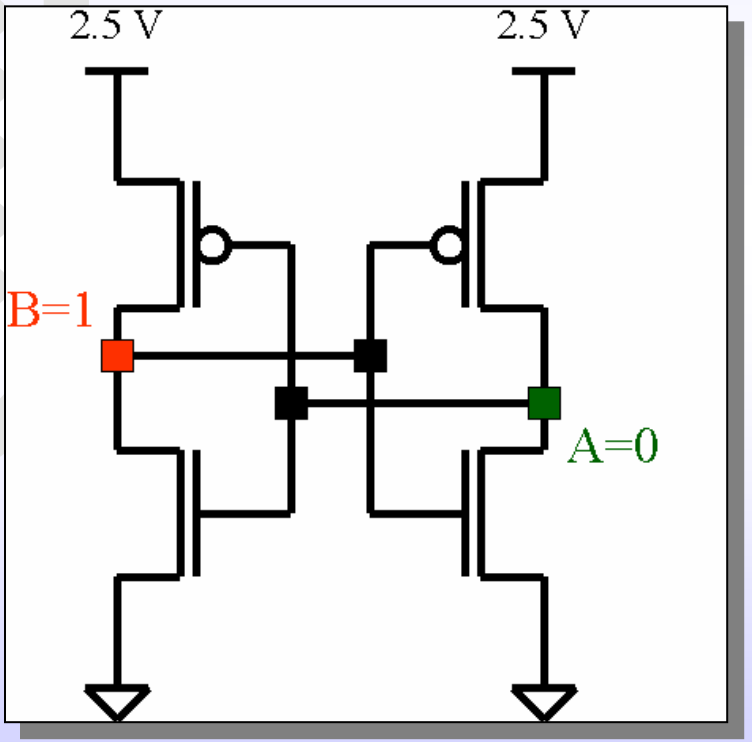
The Bathtub Curve



Guasti dovuti alle radiazioni

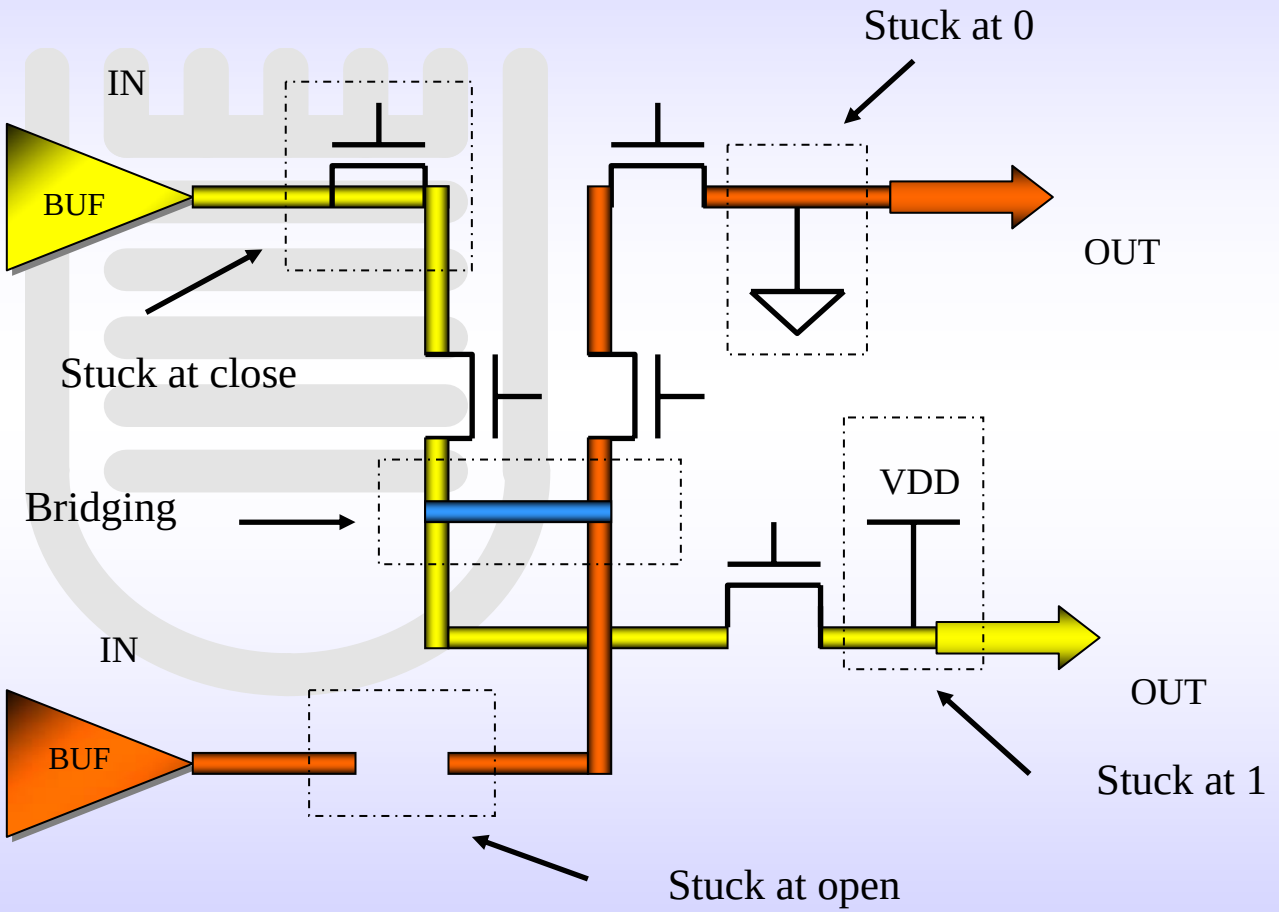
Single Event Upset di una cella di memoria SRAM





guasti nelle interconnessioni

- Interconnessioni: costituite da pass-transistors e metallizzazioni.
- Modelli di guasto
  - Stuck at 0/1;
  - Stuck at open/close;
  - Bridging tra linee;



Aumento dell'Affidabilità

- **Aumento della Robustezza del Componente Elettronico**
  - **Utilizzano processi tecnologici: SOI, Rad Hard o Space Qualified**

- **Tecniche di Rilevazione e Correzione dei Guasti:**
  - **Utilizzo di comp. elettr. allo stato dell'arte (*COTS*)**
  - **Alte prestazioni**
  - **Bassi Costi**

Molto Costosi

Pochi componenti elettronici sono realizzati in questo modo

Ridondanza

Parte del sistema che non è necessaria per il funzionamento

Sistemi Self-Checking & Fault-Tolerant

RIDONDANZA

STATICA

- ✓ Triple Modular Redundancy
- ✓ Codici a correzione di errore

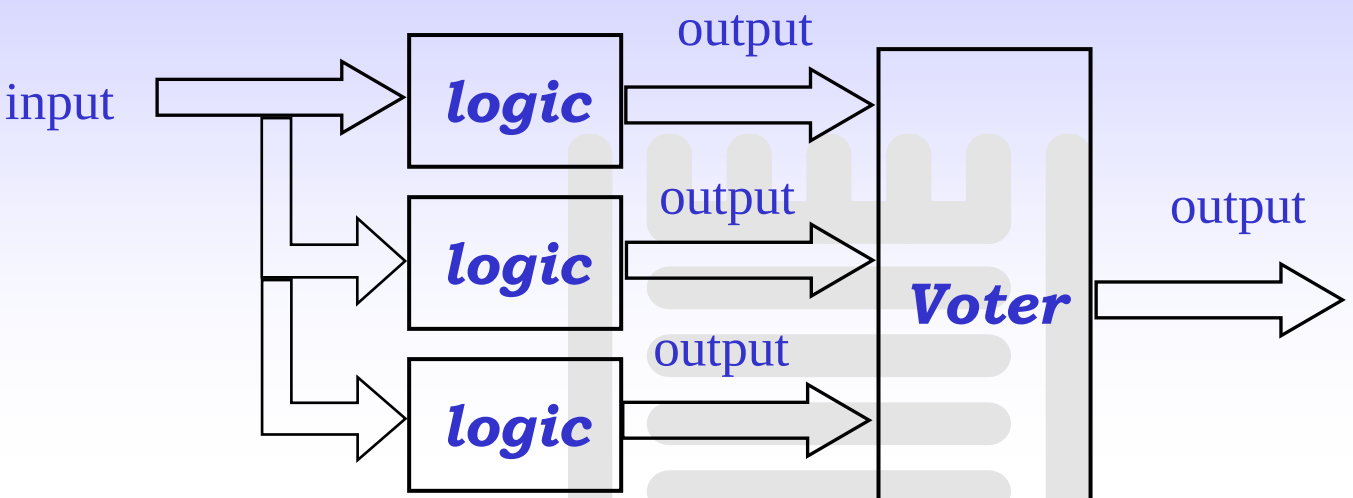
- ✓ Maggiore Overhead Hardware
- ✓ MTTR =0 (Mean Time To Repair)

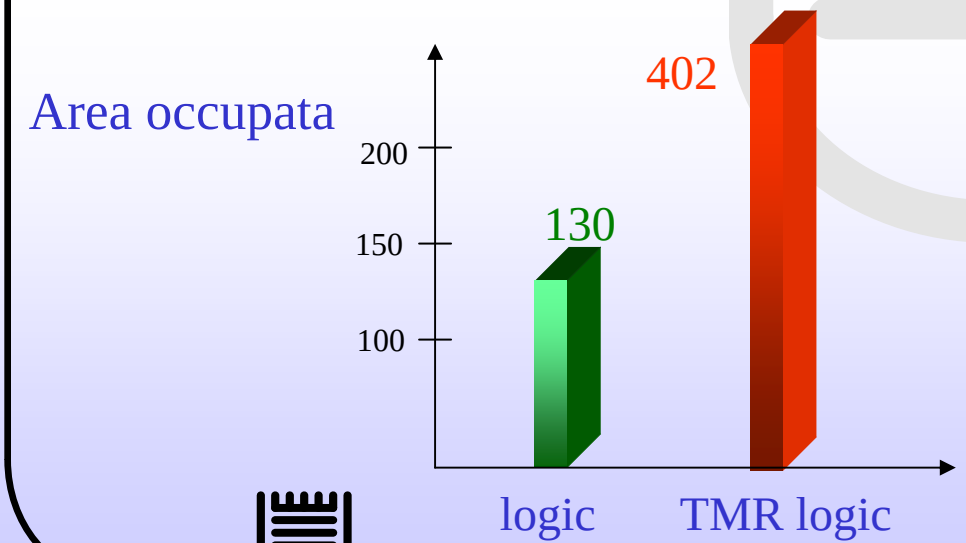
DINAMICA

- ✓ Fault detection & correction

- ✓ Minore Overhead
- ✓ MTTR > 0

TMR (Triple Module Redundancy)

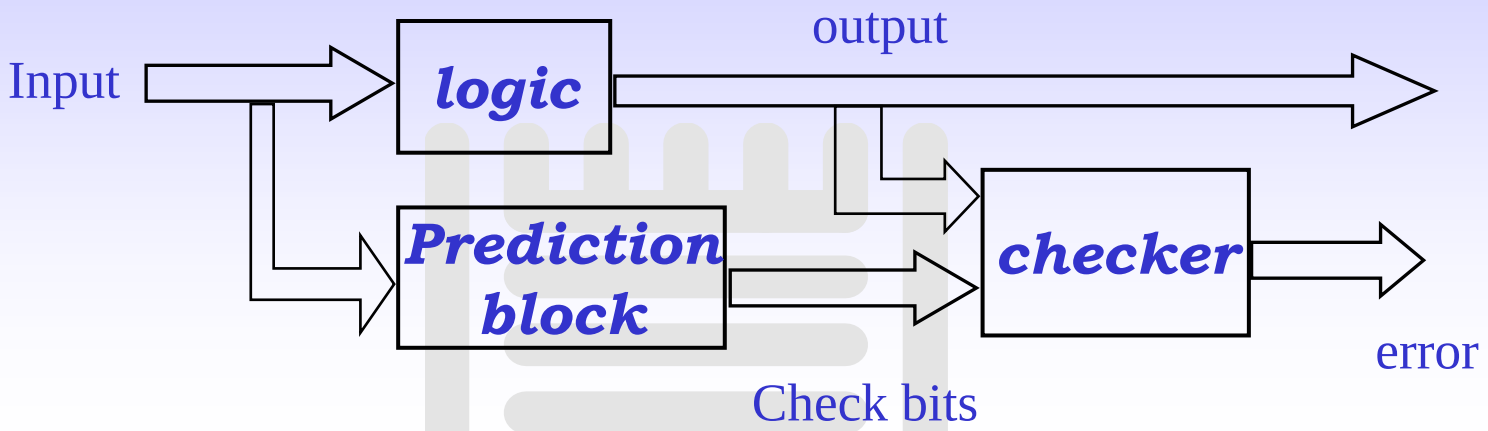


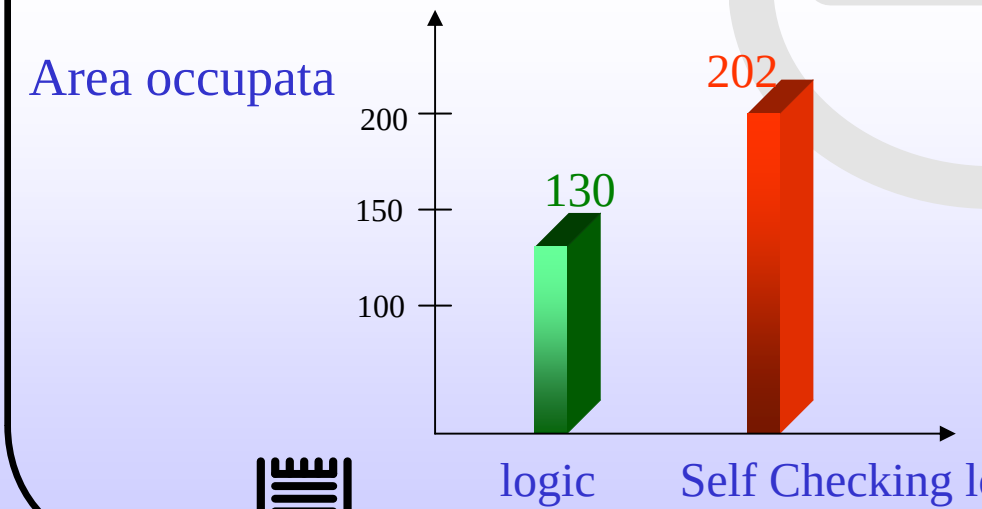


Alta Ridondanza

> 200%

Modulo Self-Checking

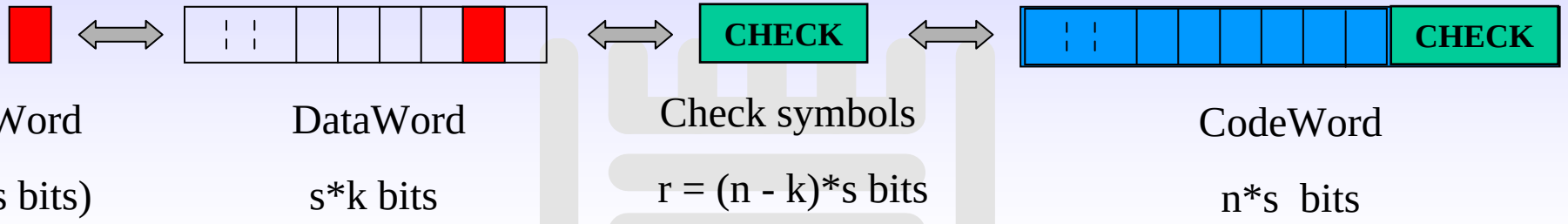




Bassa Ridondanza

77.19%

Codici



Word

(s bits)

DataWord

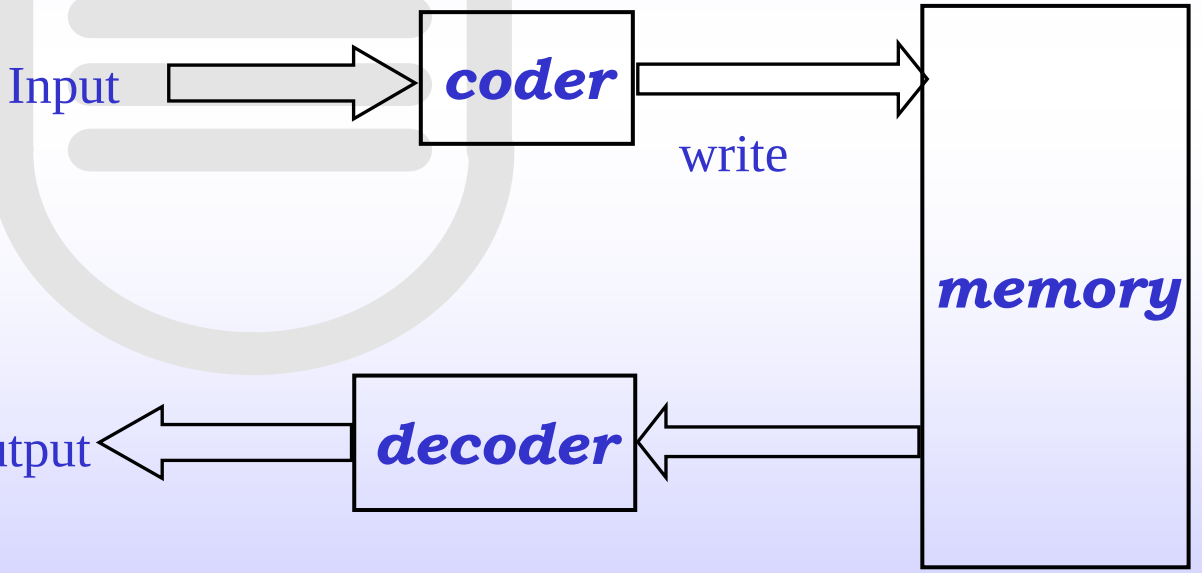
s*k bits

Check symbols

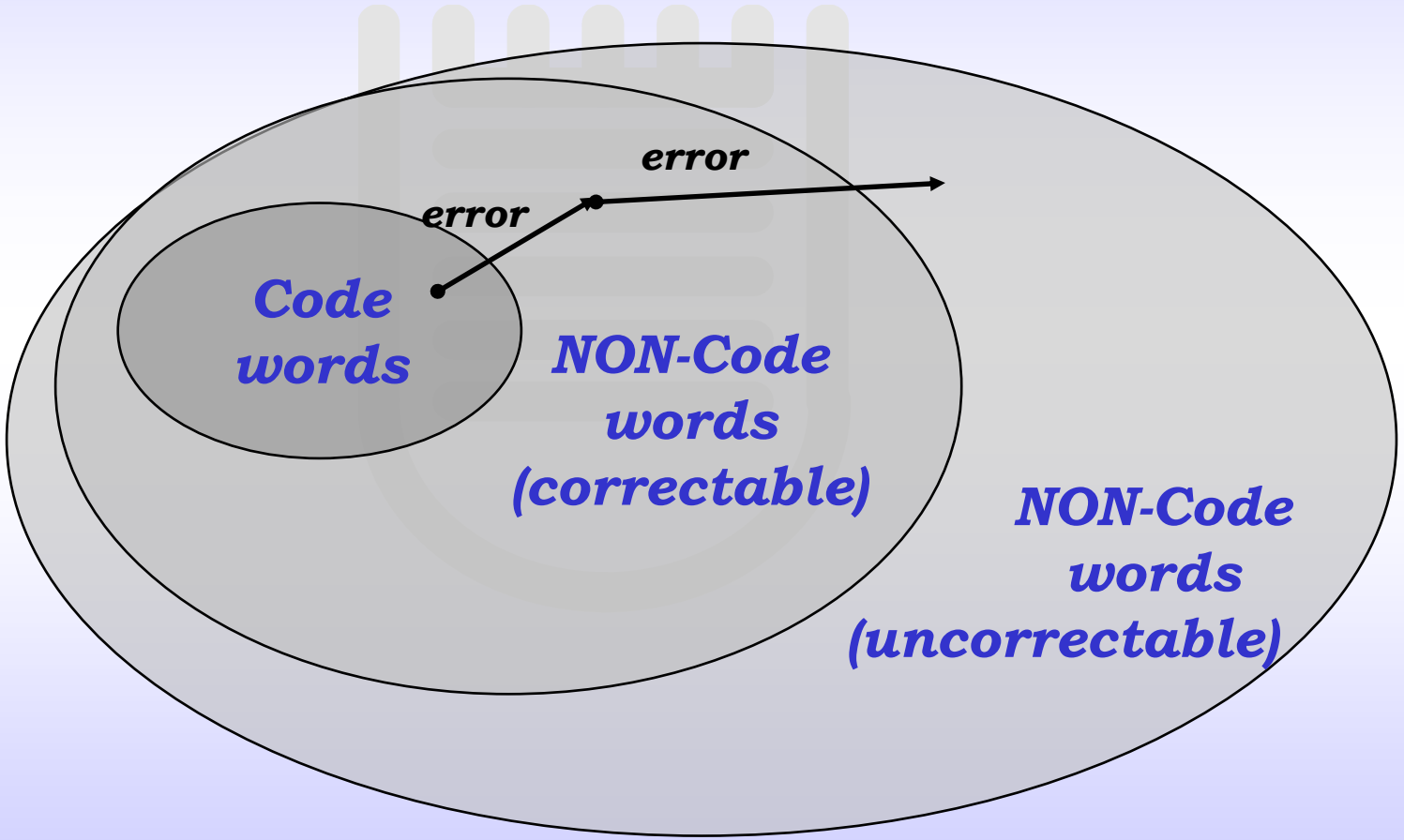
r = (n - k)*s bits

CodeWord

n*s bits



Spazio dei Codici



Codice di hamming

Hamming code: error-detecting correcting code può:

(a) Rilevare ogni errore singolo o doppio

(b) Correggere tutti gli errori singoli

Matrice di Hamming

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

$$\mathbf{d} = \begin{pmatrix} d_1 & d_2 & d_3 & d_4 \end{pmatrix}$$

Data word

$$\mathbf{c} = \mathbf{d} * G$$

$$c_1 = d_2 \oplus d_3 \oplus d_4$$

$$c_2 = d_1 \oplus d_2 \oplus d_3$$

$$c_3 = d_1 \oplus d_3 \oplus d_4$$

$$\mathbf{c} = \begin{pmatrix} d_1 & d_2 & d_3 & d_4 & c_1 & c_2 & c_3 \end{pmatrix}$$

code word

Codici di Reed Solomon

Nei codici di Reed-Solomon RS(*m,n,k*) una codeword è composta da *n* simboli di *m* bits, con una data-word di *k* simboli
Nel caso di *m*=8 (1 byte = 1 simbolo) si usa la notazione abbreviata RS(*n,k*).

- la correzione avviene sui simboli

- Si possono correggere sia i guasti permanenti (*e.g.* stuck-at nelle celle di memoria) denominati *erasures*, sia i guasti temporanei (*e.g.* SEU) denominati *random errors*

  - Il numero massimo di *random errors* correggibili è (n-k)/2

  - Il numero massimo di *erasures* correggibili è (n-k)

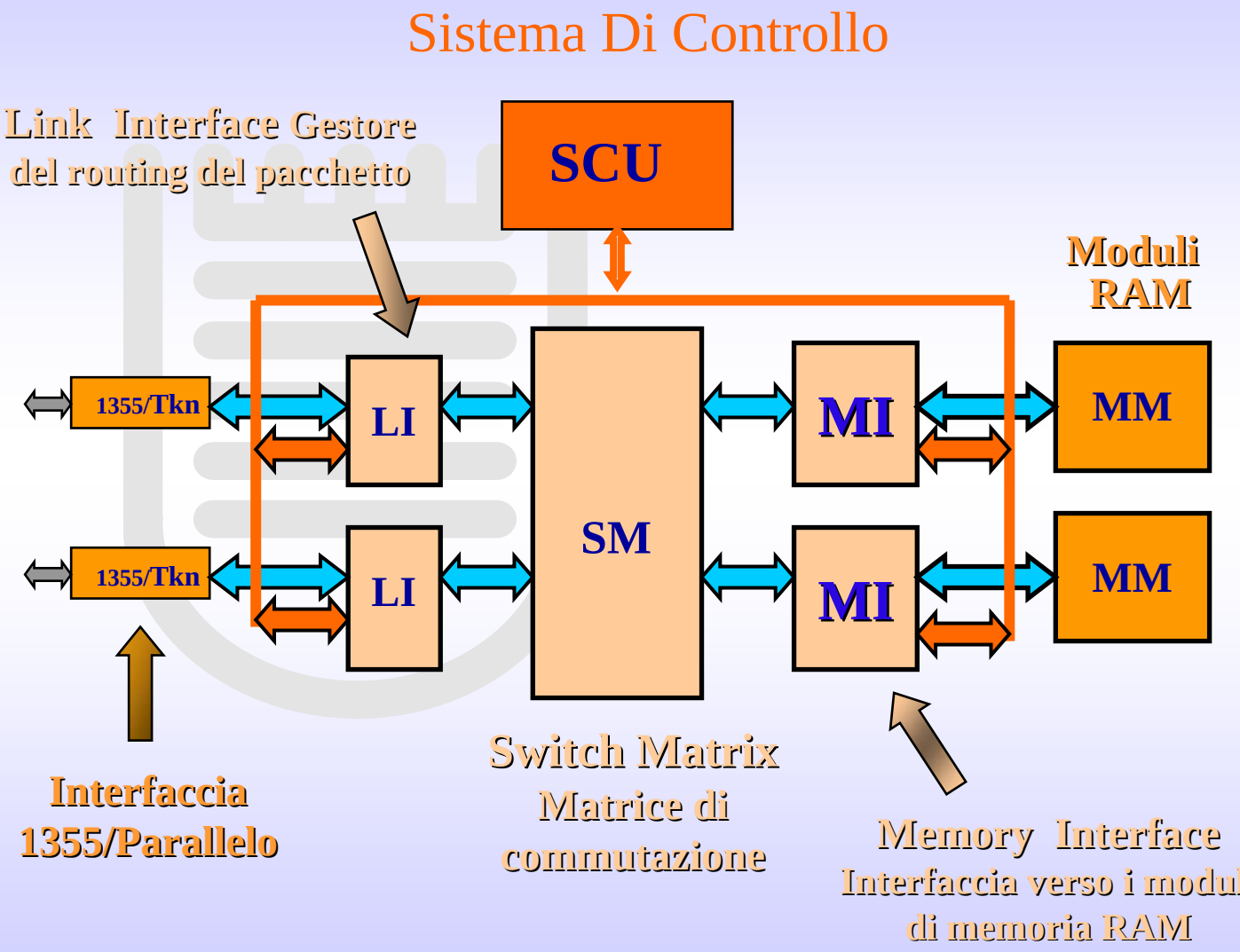
  - In generale si possono correggere 2*er+re≤ (n-k)/2 errori

OBIETTIVI DEL PROGETTO DELLA MEMORIA DI MASSA ALLO STATO SOLIDO

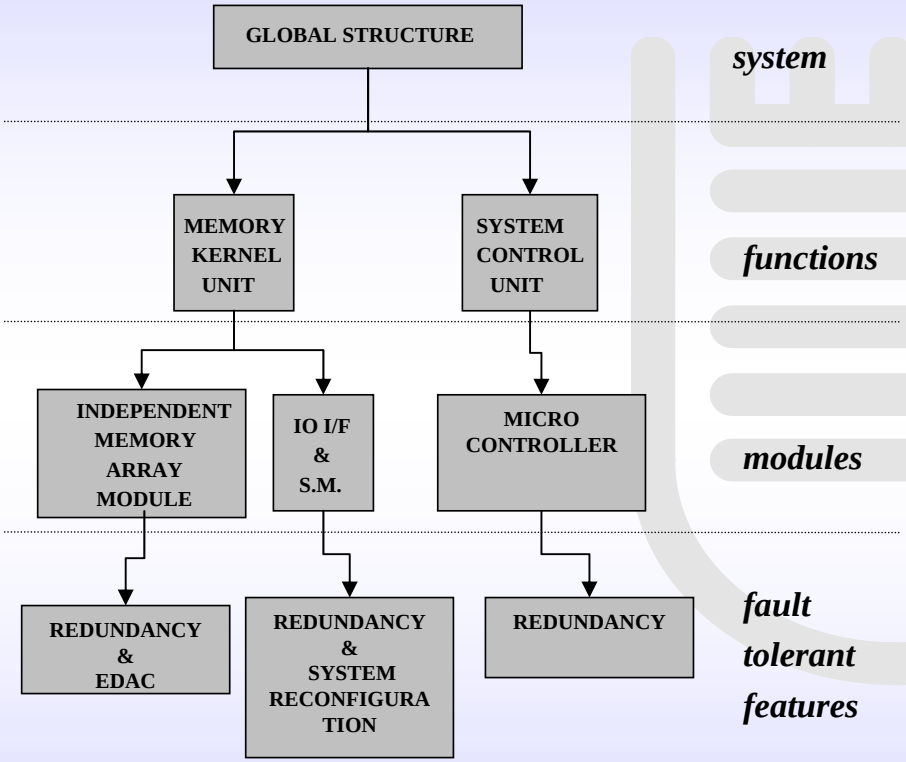
- Uso di componenti commerciali in applicazioni spaziali
- Uso di collegamenti seriali basati sul protocollo IEEE 1355.2 (Spacewire)
- Configurazione del sistema in base alla missione
- Riconfigurazione dinamica del sistema di memorizzazione con gestione distribuita del file system
- Fault tolerance: codici EDAC & ridondanza nei moduli di memoria, “graceful degradation” delle funzionalità del sistema

Struttura della Solid State Mass Memory

- **Architettura scalabile**
- **Accesso concorrente su moduli di memoria diversi**



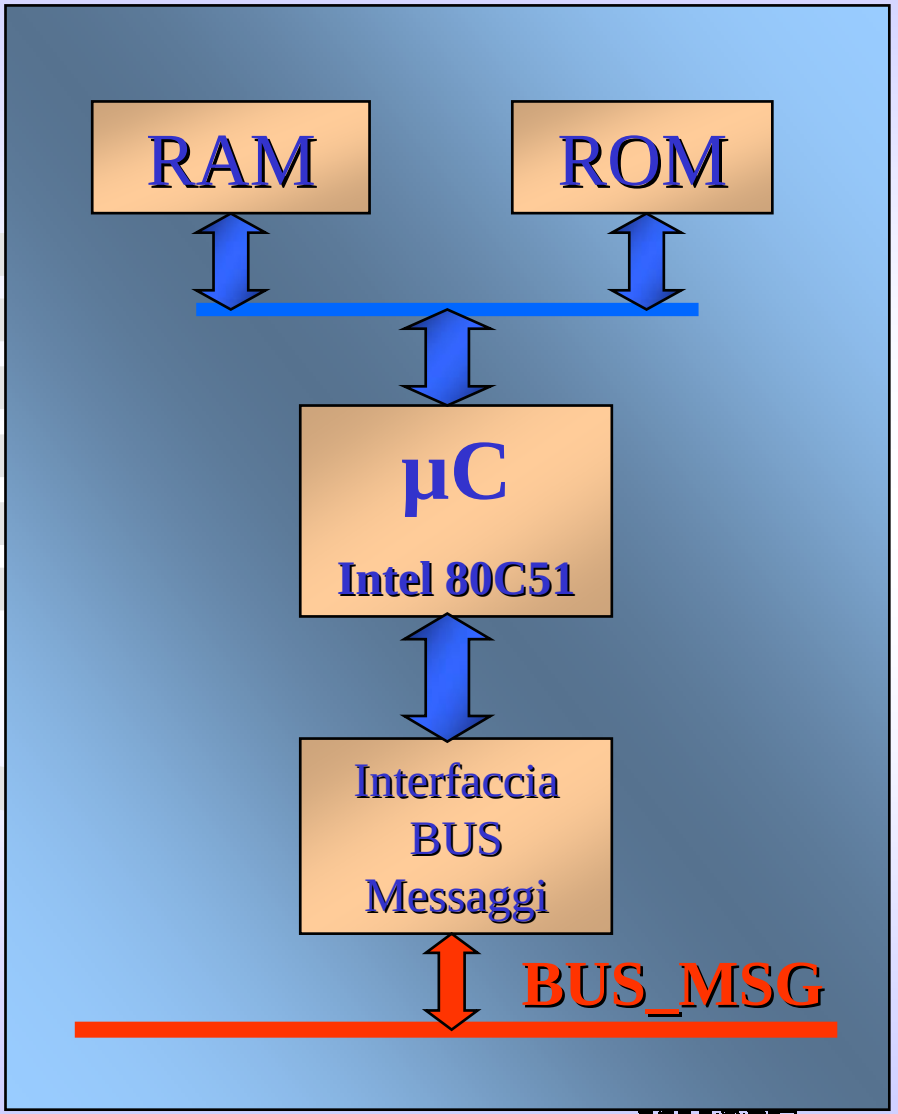
FAULT TOLERANCE DEL SISTEMA



- Ogni modulo è stato progettato con diverse caratteristiche di fault tolerance
- Ridondanza del microcontrollore (SCU)
- Affidabilità nei moduli di memoria con EDAC e ridondanza
- La modularità del sistema di interconnessione permette la riconfigurazione dinamica
- Graceful degradation delle funzionalità di sistema in caso di rotture.

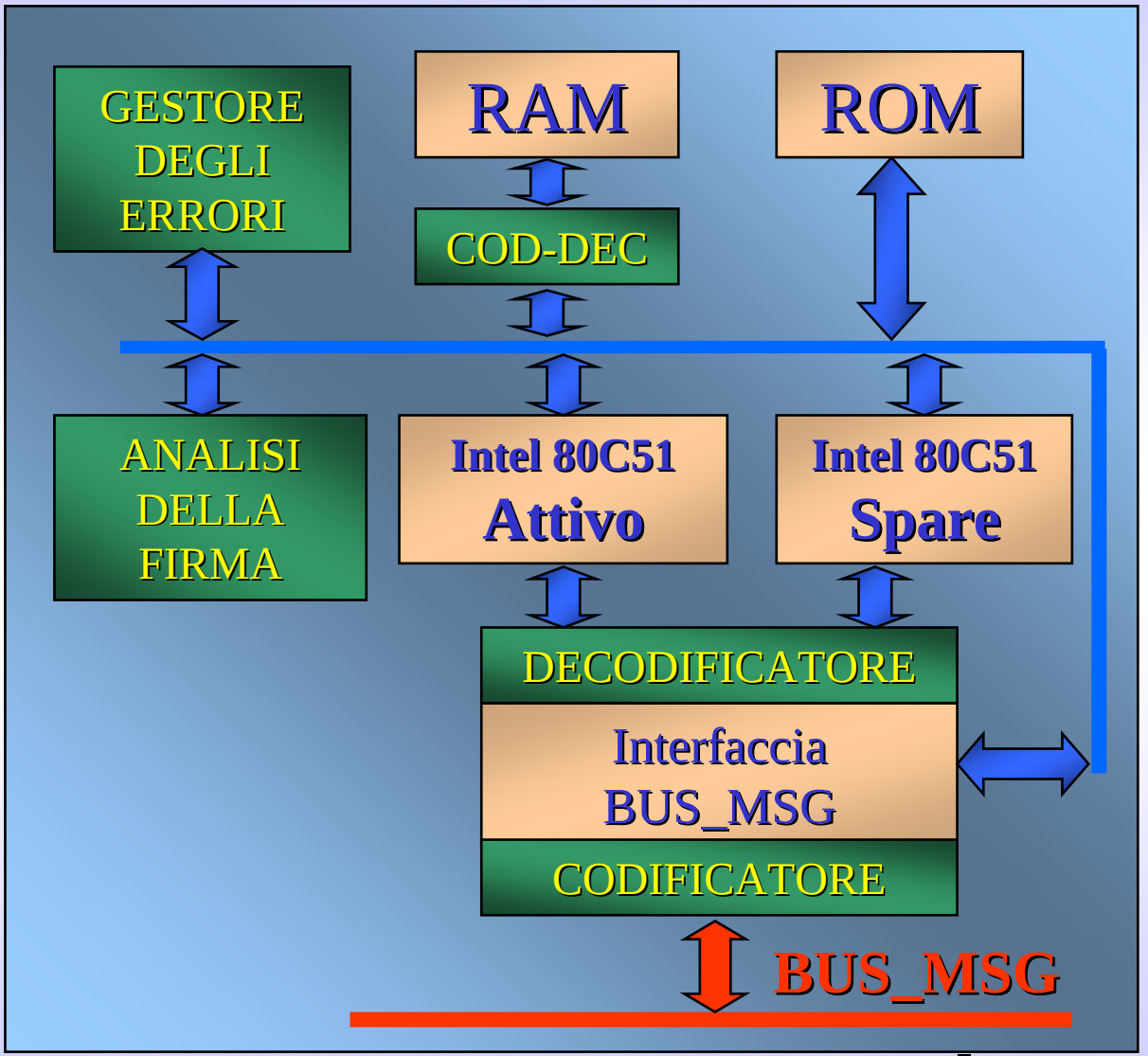
Sistema di Controllo

- Sistema base senza accorgimenti orientati alla Fault Tolerance.
- Nessuna protezione rispetto a possibili Guasti.



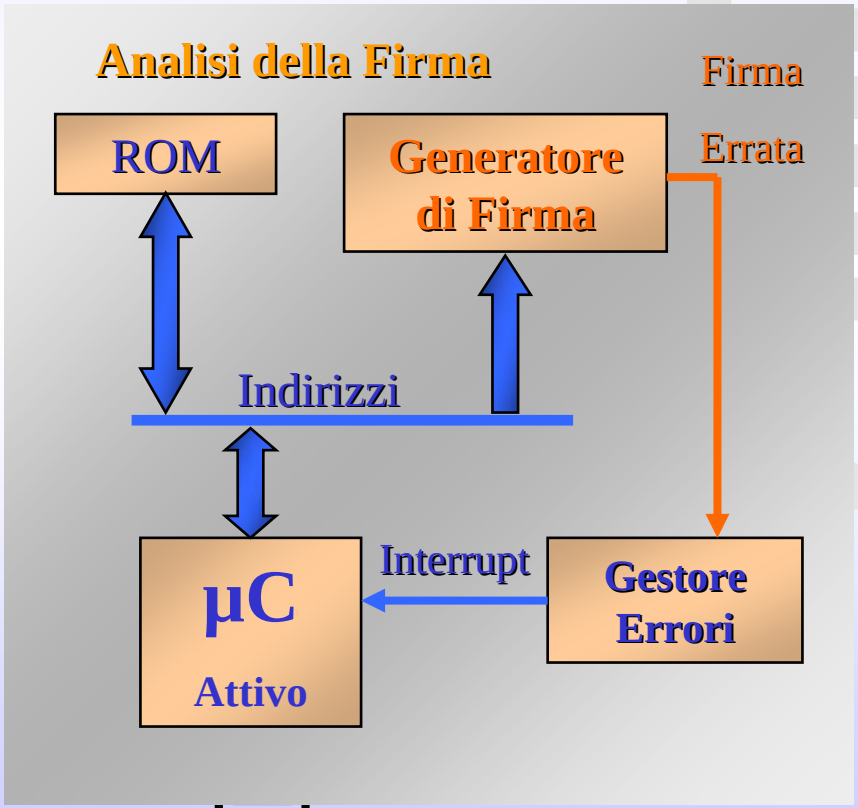
Sistema di Controllo

- Duplicazione del µC
- Protezione del µC tramite Analisi Della Firma.
- Introduzione di un Gestore degli errori.



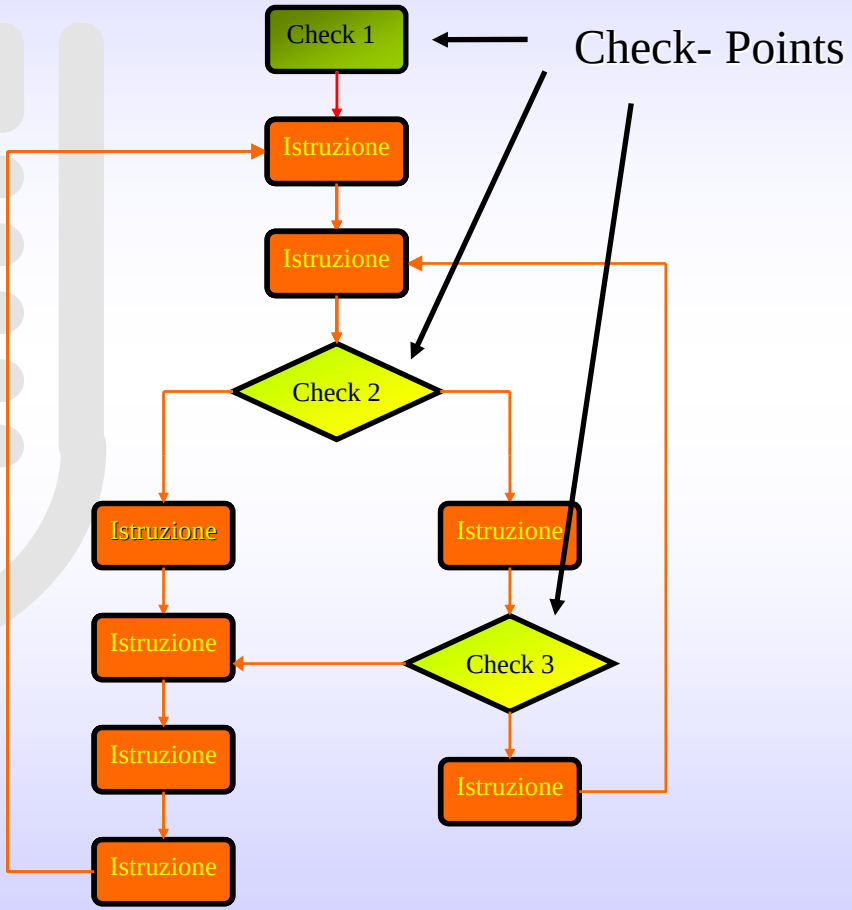
Analisi della Firma

- Controllo dell'evoluzione del programma
- Indipendenza dal programma
- Controllo Run-Time degli errori

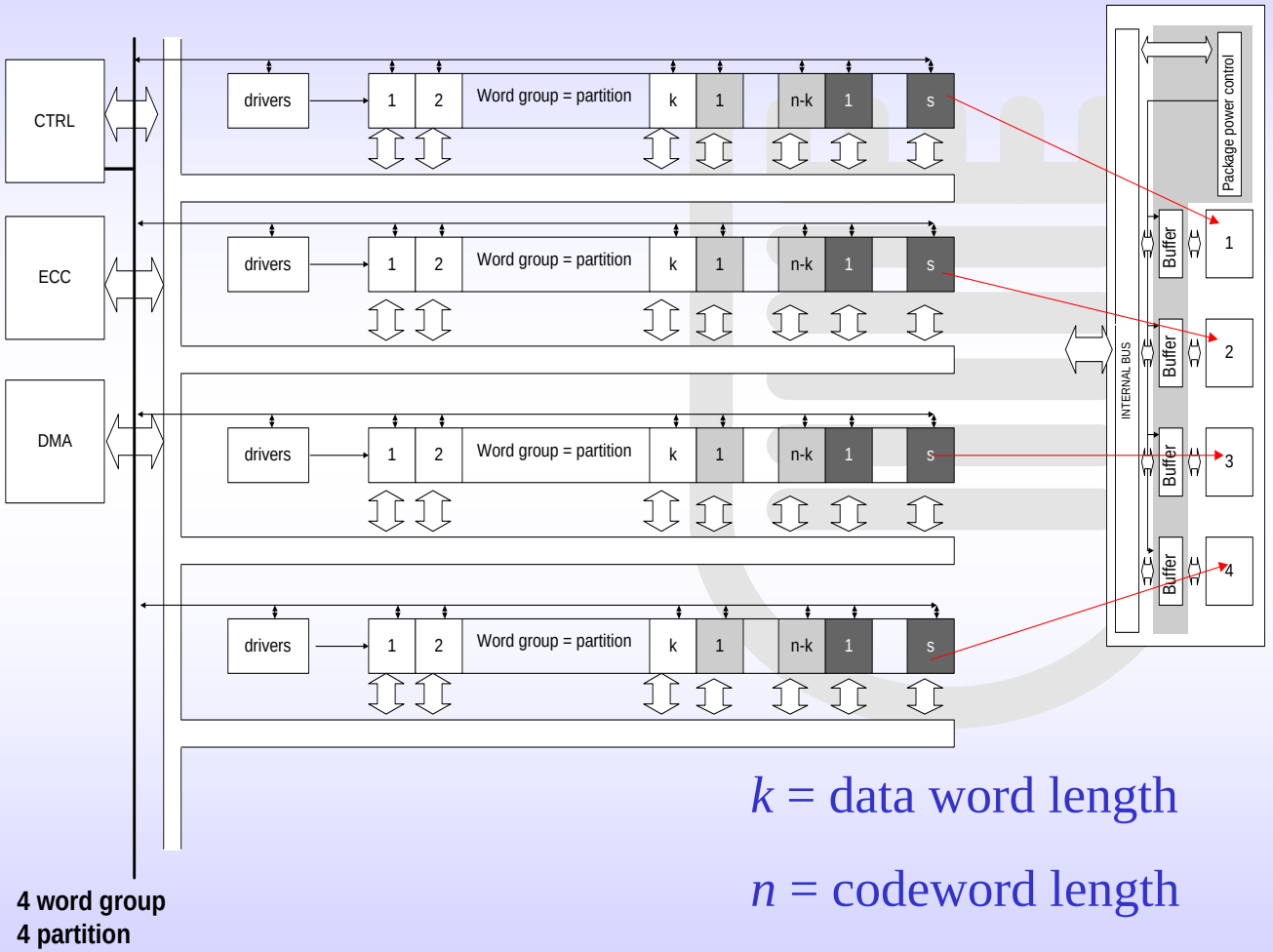


Generico Programma eseguibile

Analisi delle possibili Sequenze



Independent Memory Array Modules



- utilizza differenti codici Reed Solomon per ottenere la graceful degradation

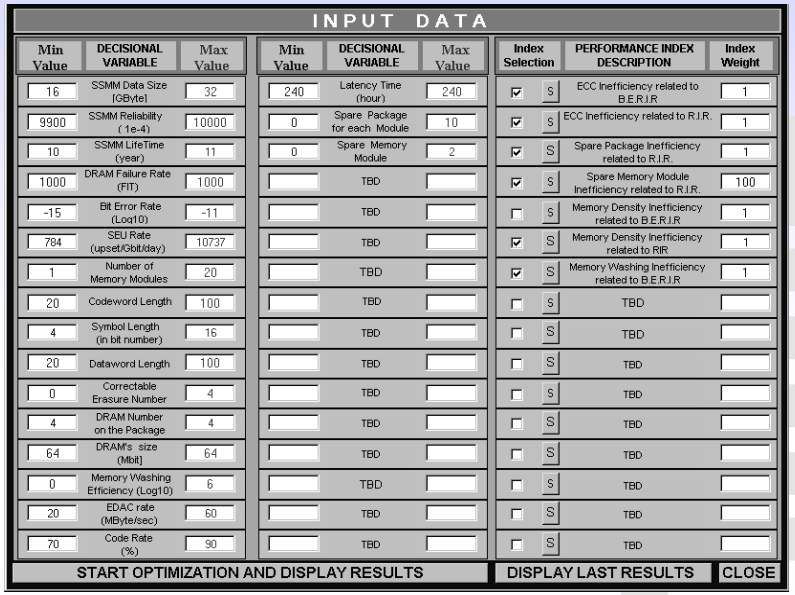
- L'ottimizzazione del memory washing permette il Trade Off tra la disponibilità e la codeword length

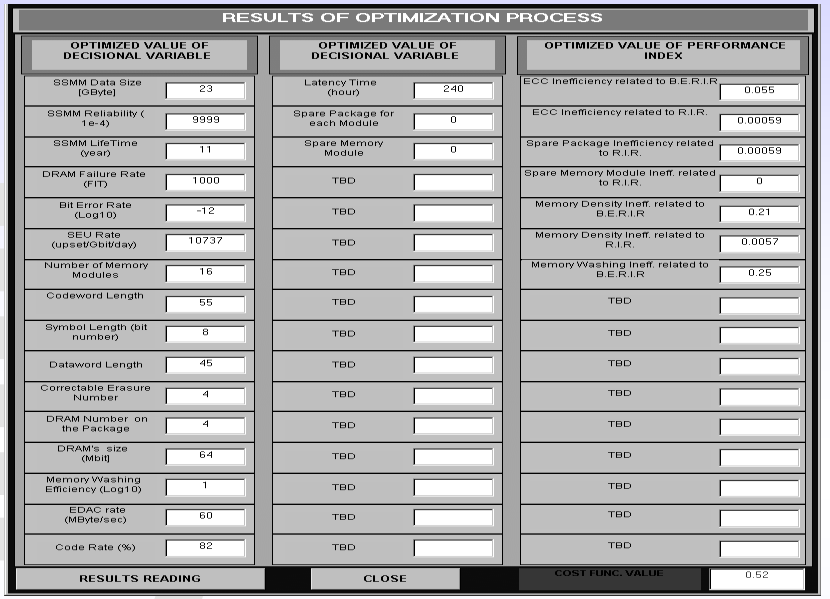
k = data word length

n = codeword length

s = spare package

Moduli di memoria



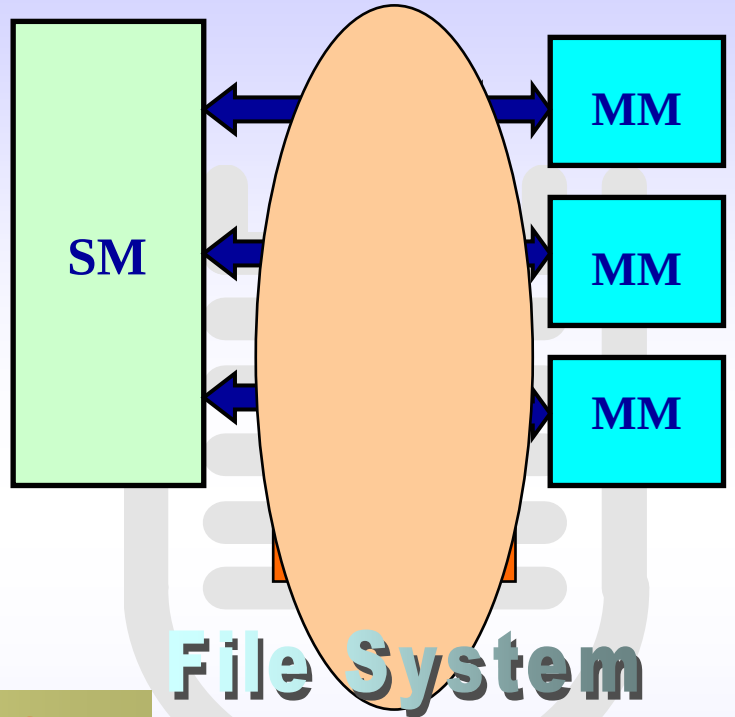


- Capacità dei moduli variabile da 1.125 Gbyte a 4.5 Gbyte
- Codifica Reed Solomon con lunghezza di codeword variabile
- Generazione dei parametri progettuali tramite un tool software originale
- Vincoli progettuali definiti dall'utente finale come input

File System

Requisiti

- Graceful degradation
- Frammentazione di uno stesso file su più moduli di memoria
- Accesso contemporaneo su più moduli di memoria diversi



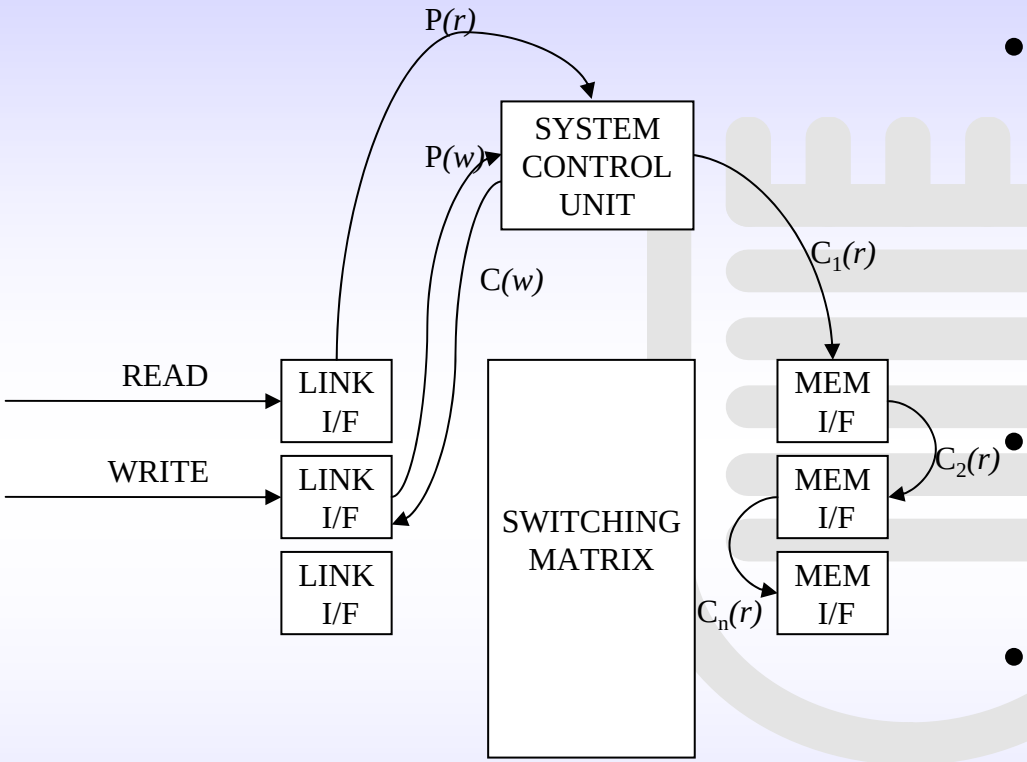
Funzioni agenti sui file

- Scrittura (creazione automatica)
- Lettura
- Cancellazione

Soluzione

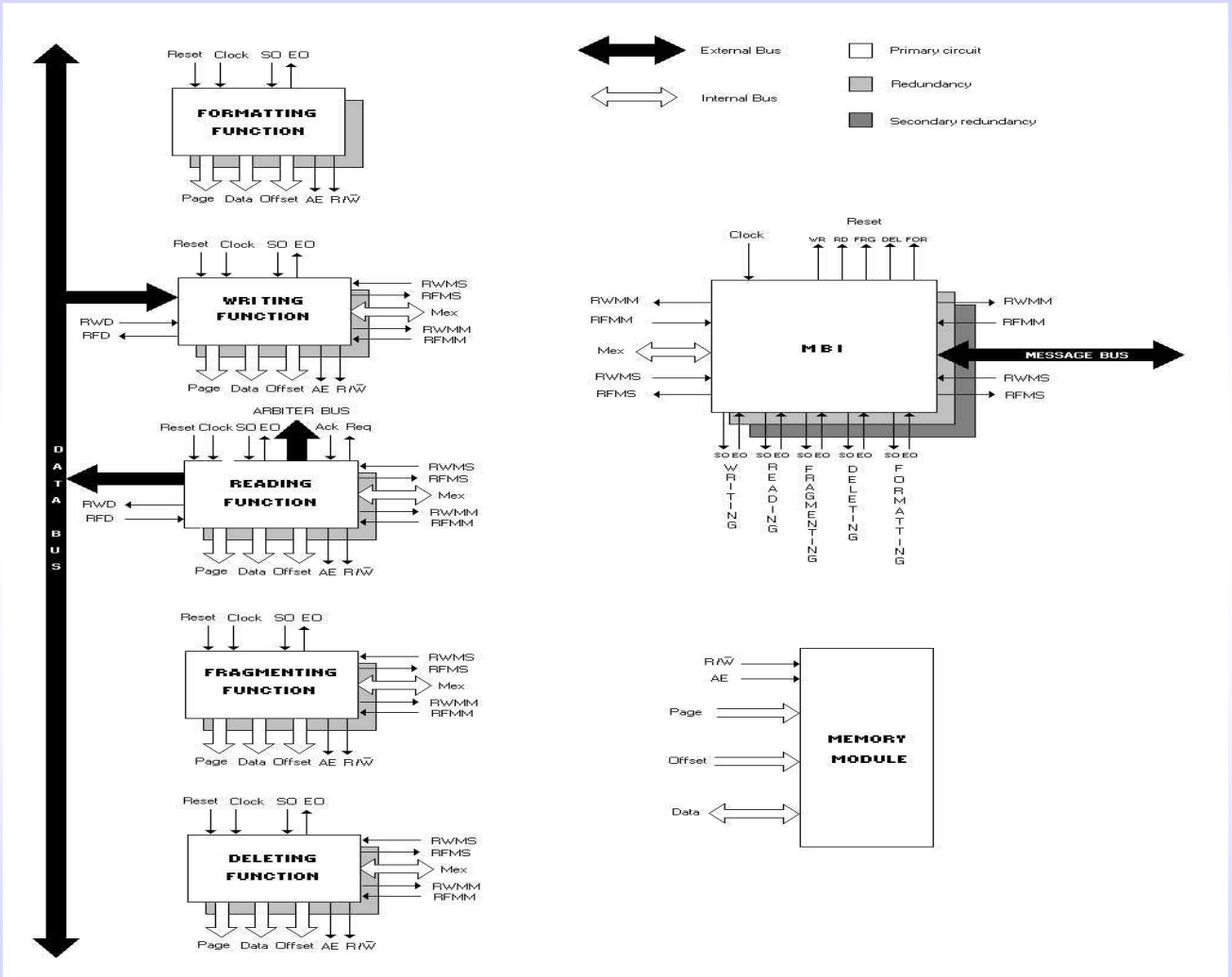
- Architettura:
  - Distribuita
- Implementazione delle directory:
  - Flat
  - Massimo 256 file
- Metodo di allocazione:
  - Lista concatenata di blocchi
- Amministrazione dei blocchi liberi:
  - Lista collegata

File System



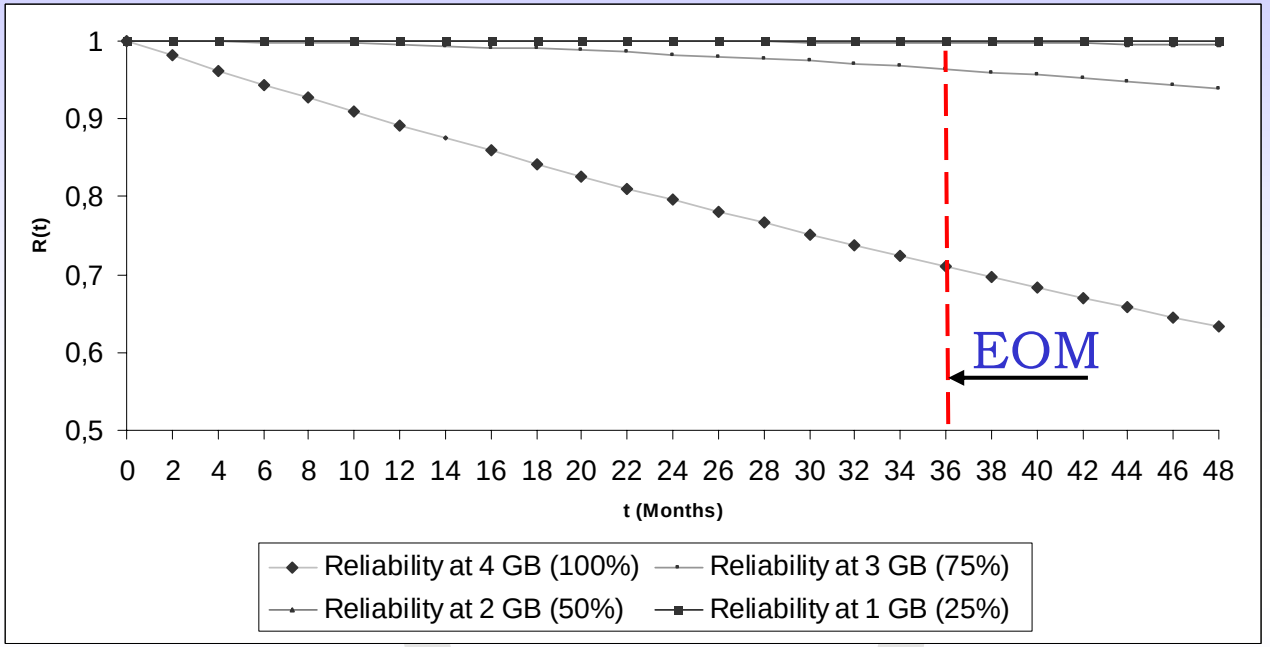
- Il Microcontrolle (S.C.U.) effettua il “file system management”:
- Operazioni di *READ*
- Operazioni di *WRITE* attravero i link s*pacewire*

Memory Interface



- Realizzazione HW delle funzioni di base (lettura,scrittura cancellazione)
- Graceful degradation delle funzionalita

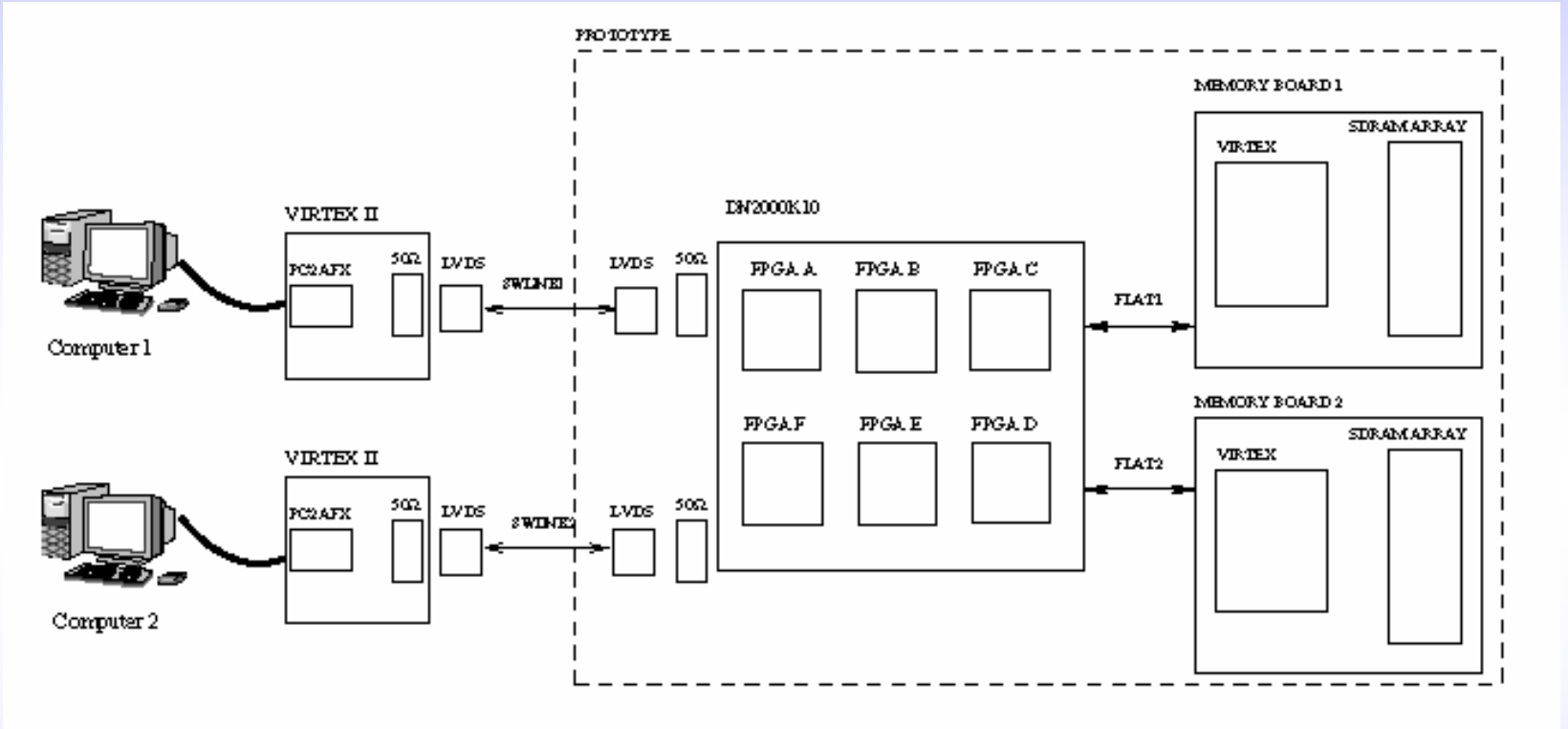
Gradeful degradation del sistema



R(t) t=36 Mesi	4 GB	3 GB	2 GB	1 GB
	0,7097	0,9638	0,9979	0,9999

- Configurazione a Overhead Minimo

Setup Prototipo



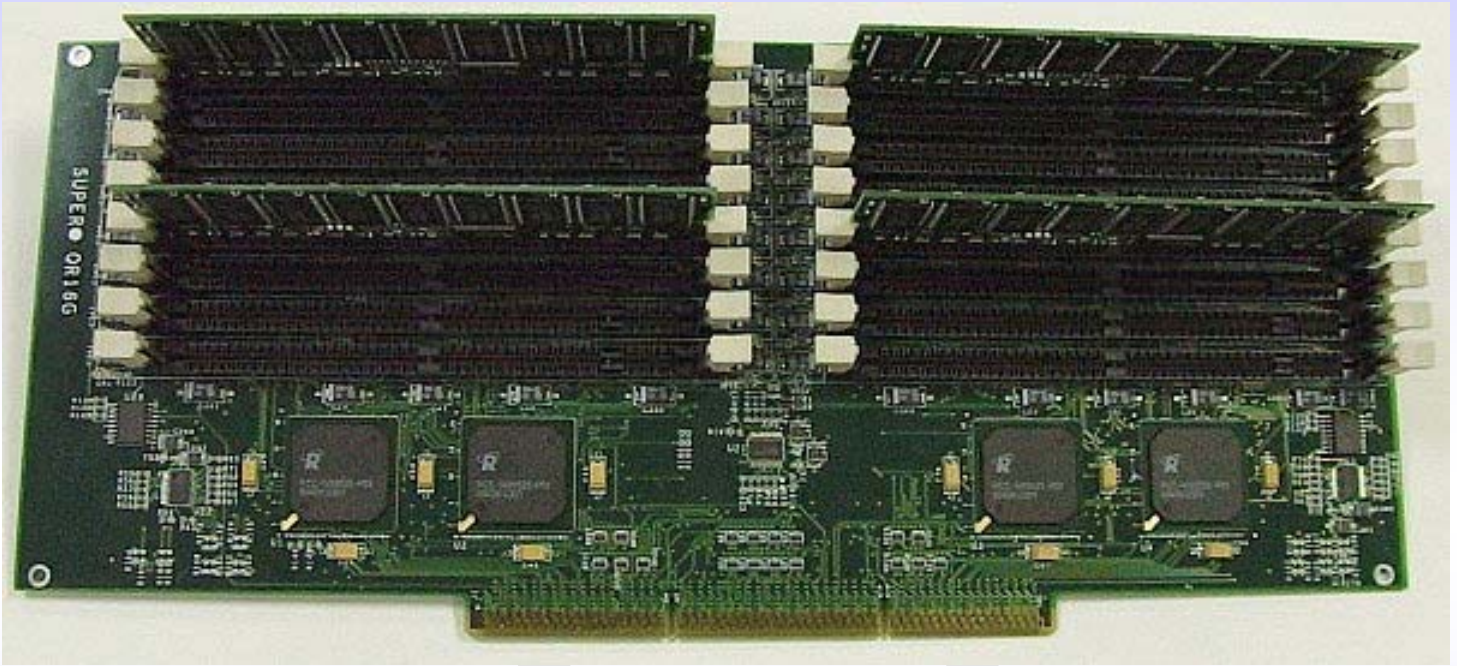
- L'uso di FPGA consente di effettuare campagne di Fault Injection

FAST PROTOTYPING (1/3)

Scheda dn2000k10
La scheda è un emulatore logico PCI-based per fast-prototyping di ASIC.
equipaggiata con:
- 6 Xilinx Virtex XCV1000
- 1 CPLD per la logica di controllo del funzionamento della scheda
- una memoria flash da 8 MB

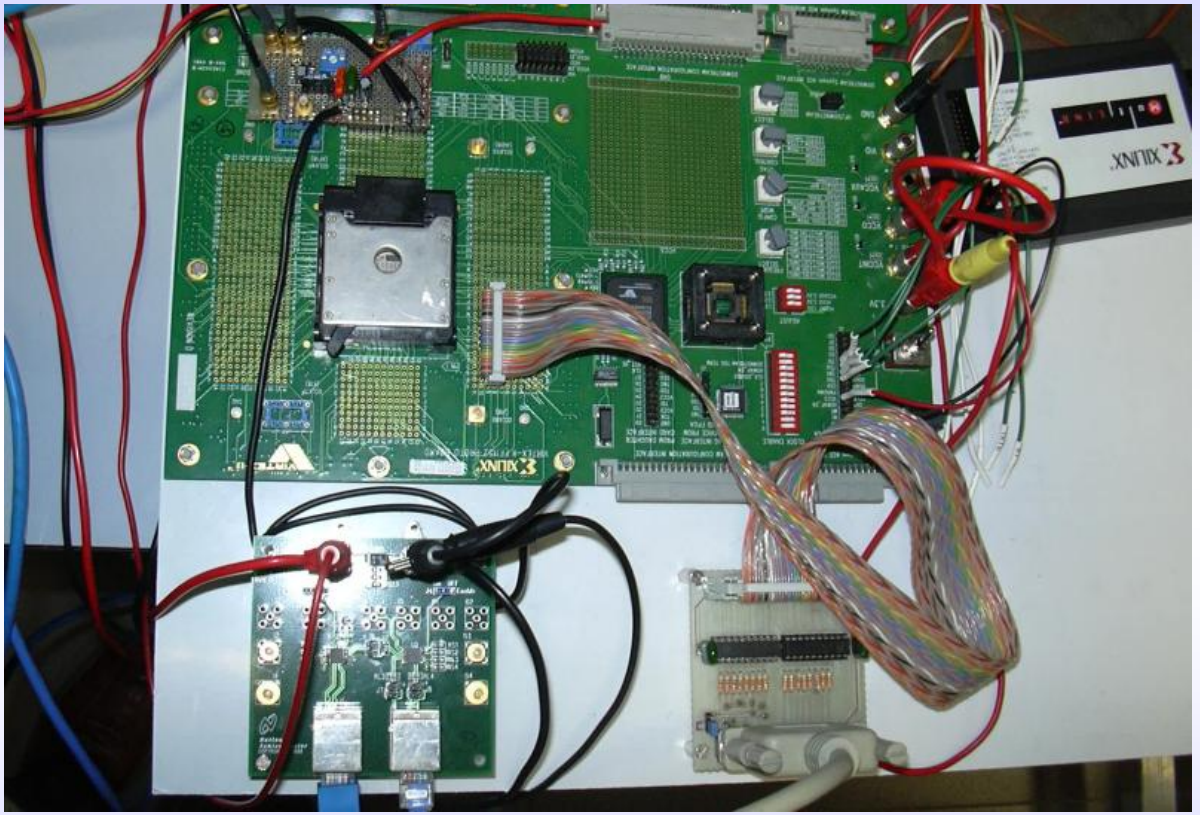
Implementa le I/O interfaces, la SCU e la matrice di switch

FAST PROTOTYPING (2/3)



La scheda sviluppata permette realizza un modulo con 4 Gbyte di memoria SDRAM con un CODEC Reed-Solomon

FAST PROTOTYPING (3/3)



- Connessione verso PC
- Link SpaceWire
- scheda di conversione LVTTL - LVDS

Conclusioni

- L'uso di componenti COTS permette di ottenere prestazioni molto piu elevate dispetto a componenti Rad-Hard
- Le metodologie esposte permettono di ottenere alta affidabilita e la graceful degradation del sistema
- Le tecniche di fast-prototyping permettono sia di diminuire i tempi di sviluppo sia di valutare l'affidabilita dei blocchi del sistema