Università degli studi di Padova

Facoltà di Ingegneria

Tesi di Laurea in

Ingegneria dell'Informazione

EFFETTI DEI SOFT ERROR NEI CIRCUITI INTEGRATI

Relatore

Prof. Alessandro Paccagnella

Candidato

Federico Zanetello

Anno Accademico 2012/2013

Ai miei genitori.

Prefazione

Lo scopo di questa tesi sta nell'affrontare uno dei grandi problemi attuali nel mondo del *Large Scale Integration* (LTI): i Soft error. Le cause di questo tipo di errori, come vedremo, sono presenti fin dalla creazione del primo *integrated circuit* (IC) costruito nel lontano 1958 dall'allora venticinquenne Jack Kilby. Inizialmente queste cause non erano nemmeno note: questi soft error semplicemente non apparivano. Solo con l'avanzamento della tecnologia, che segue la legge di Moore sin dal suo concepimento, e soprattutto grazie all'aumento e alla miniaturizzazione dei transistor nel singolo chip, questi errori hanno cominciato ad apparire. Hanno acquisito mano a mano sempre piú importanza, fino diventare uno degli argomenti di studio principali per tutti i produttori di semiconduttori, specialmente quelli che offrono una gamma di prodotti specializzati ad un uso critico in cui non é assolutamente ammesso alcun tipo di errore (almeno a livello hardware).
Con oggi, siamo arrivati ad un avanzamento tecnologico in cui i soft error sono una realtá che non puó piú essere trascurata, nemmeno nei prodotti consumer, come le memorie e i processori montati nei computer che utilizziamo ogni giorno, oppure nel nostro telefonino.
La trattazione di questo argomento é ampia: si spazia dalla fisica di base fino ad arrivare all'informatica. Nel corso della tesi verranno affrontati vari metodi di test e analisi per studiare il ripudio di prodotti reali a questa tipologia d'errore, fino ad arrivare alle varie soluzioni proposte che vengono (o meno) implementate nei circuiti moderni.

Indice

Introduzione

Grazie all'avanzamento tecnologico, siamo entrati in un'era di chip ad alta integrazione, minor consumo energetico e alte frequenze di clock. Tutti questi benefici hanno peró anche un lato negativo: i soft error.
Al fine d'introdurre l'argomento, nel prossimo capitolo verranno inizialmente sintetizzati i concetti fondamentali per affrontare il tema: la differenza tra soft e hard error, spiegando, oltretutto, perché ci si concentra sui primi e non sui secondi; le origini e le cause di tali eventi, che possono avere nature molto diverse e che sono state scoperte soltanto con il progredire degli anni; le varie tipologie di soft error; e le probabilitá che essi si verifichino. Infine daremo un breve riepilogo sulle differenze piú sostanziali tra il transistor Bulk e il transistor SOI (*Silicon On Isolator*).
Il secondo capitolo verterá sull'analisi teorica e sperimentale, introducendo tutti i fattori fondamentali da considerare nell'ambito dei test e proponendo vari risultati raccolti nel decennio scorso da IBM, societá grazie alla quale si é appreso molto di questi eventi. Si noti, inoltre, che le tipologie di analisi introdotte in questa tesi non sono le uniche utilizzate nel mondo reale dai produttori di Silicio. Esiste un'altra zona, qui solamente accennata, di cui si fa sempre piú uso anche grazie alla velocitá con cui si ottengono i risultati senza dover eseguire i test sperimentali, lunghi e costosi: la simulazione al computer.
Il terzo capitolo verterá sulle soluzioni proposte e adottate dai produttori di chip, chiarificando pro e contro di ciascuna di esse e spiegando le motivazioni dietro le quali alcune soluzioni sono piú diffuse rispetto ad altre in certi campi d'utilizzo, mentre in altre realtá é vero l'opposto.
Il quarto e quinto capitolo introdurranno il lettore a degli esperimenti reali effettuati su prodotti moderni, con risultati e considerazioni.

Capitolo 1

Soft error: chi, cosa e perché

1.1 Hard error vs. Soft error

Nonostante il nome possa trarre in inganno, lo studio dei soft error é attualmente molto piú importante rispetto allo studio degli hard error, anche perché questi ultimi sono ben conosciuti nel campo da tempo. Sebbene il verifcarsi di un hard error sia grave, esso avviene probabilmente a causa di un circuito mal progettato, non rispettante le cosiddette *regole di layout*, o per cause esterne, come le scariche elettrostatiche.
Ma andiamo per ordine e spieghiamo di cosa stiamo effettivamente parlando: gli errori circuitali, conformemente ai *fault* dell'informatica, possono essere classificati come permanenti o transitori (detti, a volte, intermittenti). Un errore permanente (hard error) puó essere causato da un errore nella progettazione del chip o da un suo guasto (causato, ad esempio, da una eccessiva tensione di alimentazione). Per scovare agevolmente questo tipo di errori, nell'industria del silicio viene utilizzata una tecnica detta *burn-in* che sottopone i chip ad un test accelerato (ad alta temperatura) subito dopo la loro produzione.
Dissimilmente dai soft error, per eliminare un errore permanente bisogna sostituire il chip o l'intero modulo ad esso associato.
Un errore transitorio, invece, é un tipo di errore che fa variare il valore di un dato (un'istruzione di un programma in esecuzione, un indirizzo di memoria, una variabile, etc.) che peró non danneggia l'hardware del sistema: il danno avviene esclusivamente nei dati volatili su cui si sta operando.
Questo tipo di errore é alquanto fastidioso, poiché non é né prevedibile né, in caso di esecuzioni di dati molto importanti (ad esempio delle transizioni bancarie o in sistemi *safety critical*), tollerato.
I primi progettisti di memorie rimasero molto perplessi dall'osservare nelle memorie DRAM l'occorrenza sistematica di particolari errori, né permanenti, né ricorrenti. Ció non poteva essere spiegato né con i disturbi nella tensione di alimentazione, né con la corrente di perdita né con gli accoppiamenti capacitivi. Questa categoria di errore fu definita *soft error* (errore transitorio).

1.2 Importanza

Inizialmente, i soft error erano noti a causa dei SEU (Single-event upset) che avvenivano quasi esclusivamente nei circuiti di memoria prodotti in tecnologia CMOS (specialmente nelle DRAM).

> Si prenda atto che i soft error sono un sottoinsieme dei *single event effects* (SEE), ossia di un qualsiasi cambio di stato su un componente di un circuito causato dal passaggio di una particella energetica. I SEE hanno come sottoinsiemi i giá citati SEU, ossia soft error prodotti dall'induzione di un segnale transitorio (causato dal passaggio di una particella energetica); i *multiple-bit upsets* (MBU), ossia la corruzione di piú nodi della memoria causata da due o piú errori nella stessa *word*; i *multi-cell upstets* (MCU), simili ai precedenti, causati dalla corruzione di piú celle quando si verificano eventi ad alta energia; e molti altri tipi di errore. Quindi non é detto, come vedremo nel paragrafo 1.4, che un qualsiasi SEU o un altro tipo di disturbo elettronico generi automaticamente un soft error.

Negli ultimi 30-40 anni i soft error sono diventati sempre piú una preoccupazione dei produttori e utenti dei circuiti a semiconduttori: con l'arrivo dei primi chip a 90nm, la situazione é diventata assai pesante a causa della comparsa dei primi soft error non solo nei circuiti come flip-flop e nei latch, ma anche nella logica, espandendo il problema all'intero chip. Per i chip costruiti in tecnologia SOI (si veda il paragrafo 1.6), la transizione da 65nm a 45nm contribuirá nell'aumento del valore del SER (si faccia riferimento al paragrafo 1.8.3) di ogni singolo latch di un fattore che va da 2 a 5.
Due sono i fattori principali che hanno incrementato la suscettivitá dei chip ai soft error:

1. la riduzione delle dimensioni dei transistor: con essa si é ridotta a sua volta la cosiddetta carica critica Q_{critic}, ossia la carica richiesta per variare il valore logico di un nodo di I/O all'interno di un latch. Con la diminuzione di tali dimensioni anche le capacitá parassite si sono ridotte, per non parlare della riduzione della tensione di alimentazione richiesta per il funzionamento del chip: l'insieme di tutti questi fattori ha contribuito a diminuire la Q_{critic};

2. nonostante la riduzione delle dimensioni totali di un latch implichi che la sua superficie sensibile sia diminuita, questo ha permesso alle nuove architetture di implementare molti piú latch rispetto alle generazioni precedenti, si noti, inoltre, che la dimensione media dei microprocessori ha sempre teso all'aumento con l'avanzamento delle generazioni;

Inoltre, i latch sono sempre stati usati come controllori dei SEU (sia nelle memorie che in altri tipi di circuiti) in modo tale da correggere un soft error nel caso che ne fosse avvenuto uno. Se anche questi circuiti sono diventati sensibili ai soft error, il problema assume una nuova importanza.

1.3 Cause & Origini

A questo punto un novizio del campo potrebbe chiedersi: *cosa causa questi soft error? come mai non é ancora stata trovata una soluzione definitiva?*

Un soft error avviene quando la traiettoria di una particella energetica attraversa un nodo appartenente ad una cella di memoria o ad un nodo logico. Tale particella interagisce con la

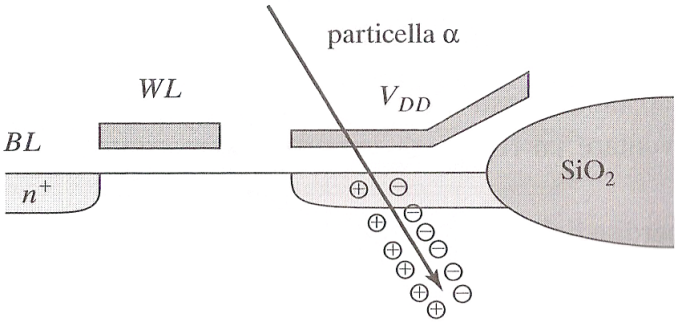

Figura 1.1 – Soft error generato da una particella alfa

struttura cristallina del nodo mettendo in moto una generazione di qualche milione di coppie elettrone-lacuna nel substrato: gli elettroni generati che raggiungono il bordo della regione di svuotamento (anche conosciuta come regione di carica spaziale, RCS) prima di ricombinarsi con le lacune sono spinti dentro il nodo dal campo elettrico, come in un normale processo di diffusione. Se, in questo processo, venissero raccolti una quantitá sufficientemente di elettroni nella giunzione per creare un canale indotto, il valore memorizzato nel nodo potrebbe variare, ottenendo cosí un potenziale errore.

Esistono molte tipologie di particelle: le principali sono introdotte brevemente nei seguenti paragrafi.

1.3.1 Raggi cosmici

I raggi cosmici sono particelle energetiche (particolarmente protoni, neutroni e nuclei atomici) provenienti dallo spazio cosmico che possono avere natura molto varia (l'energia cinetica di tali particelle é distribuita su quattordici ordini di grandezza), cosí come varia é la loro origine, tra i maggiori contribuenti di nostro interesse abbiamo: il Sole (in questo caso si parla comunemente di *vento solare*), le altre stelle, fenomeni energetici come novae e supernovae, fino ad oggetti remoti come i quasar. La maggior parte dei raggi cosmici che arrivano (piú precisamente precipitano, visto che la loro velocitá sfiora quella delle luce) sulla Terra vengono fermati dall'atmosfera, con interazioni che, tipicamente, producono una cascata di particelle secondarie (*air shower*, principalmente neutroni per quanto d'interesse in questa tesi) a partire da una singola particella energetica.
I raggi cosmici sono diventati la causa maggiore di soft error nelle ultime generazioni di microprocessori. Al contrario delle particelle alfa, l'intensitá dei raggi cosmici (o, per meglio dire, delle loro radiazioni) nell'atmosfera varia con l'altitudine, con la posizione rispetto al campo geomagnetico e con l'attivitá solare. La barriera atmosferica ad una data altezza puó essere determinata calcolando lo spessore della massa dell'aria per unitá d'area (chiamata *profonditá atmosferica*). Il campo geomagnetico terrestre riflette verso lo spazio le particelle cosmiche cariche con bassa quantitá di moto: data la posizione di una particella la minima

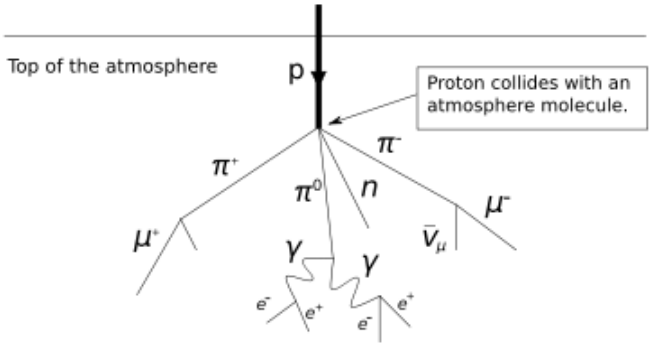

Figura 1.2 – Schema semplificato di una *air shower*: il numero di particelle generate da un tale evento puó raggiungere la grandezza dei miliardi (in base all'energia iniziale e all'ambiente)

quantitá di moto per unitá di carica che essa deve possedere per arrivare ad una determinata altitudine viene detta *Geomagnetic rigidity cut-off*. Per quanto riguarda l'attivitá solare, essa varia in base al ciclo solare, che sappiamo essere di circa 11 anni (vista la natura irregolare, puó durare fino ad un anno in piú o uno in meno), determinata in base al numero di macchie solari presenti nel momento d'interesse: piú macchie sono visibili (anche ad occhio nudo!), piú il sole emetterá energia e materia nello spazio circostante. Tuttavia, tra il periodo di massima attivitá (detto *massimo solare*) e il periodo di minima attivitá (*minimo solare*) la differenza effettiva nel flusso di raggi cosmici medio varia solamente di un $\pm 7\%$.
In fin dei conti, il parametro piú importante per determinare il flusso terrestre delle radiazioni cosmiche é la profonditá atmosferica, che, come giá accennato, risulta essere proporzionale alla pressione barometrica e sensibile all'altitudine.
Il livello del flusso di raggi cosmici medio é di 14 neutroni/cm^2/ora al livello del mare a New York city (che é uno standard), tale flusso aumenta di circa 2.2 volte per ogni km (in altezza): se si fosse interessati ad ottenere con maggior precisione il valore di tale flusso in un determinato luogo, ci si puó recare all'indirizzo web http://seutest.com/cgi-bin/FluxCalculator.cgi ove, inserendo i dati descritti sopra, si potrá ottenere una stima piú precisa del flusso nella localitá desiderata.

1.3.2 Particelle alfa (α)

Queste particelle consistono di due protoni e due neutroni legati insieme nel nucleo di un atomo di elio (He). Le particelle alfa sono emesse da nuclei radioattivi, come gli isotopi di Uranio, Torio e Radio, durante un processo conoscuto come decadimento alfa:
questo avviene in quanto atomi instabili (e dunque radioattivi) trasmutano per stabilizzarsi emettendo appunto una particella alfa (da cui il nome). Alternativamente, tali particelle sono generate dalle impurezze radioattive nei material di cui sono fatti i *package* dei chip: durante la produzione di IC é, difatti, inevitabile l'immissione di impuritá. Le tipiche fonti sono C4 (controlled collapse chip connection) solder bumps e molte altre introdotte durante la lavorazione del wafer. Queste sono le cause principali dell'emissione di particelle alfa.

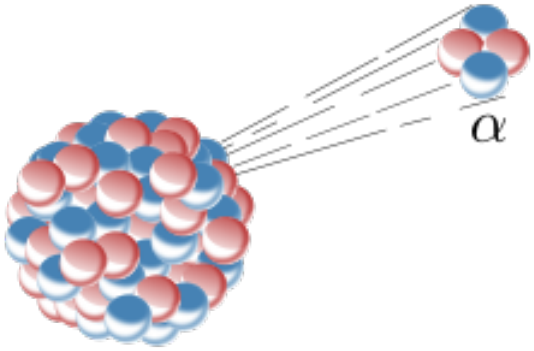

Figura 1.3 – Decadimento alfa: enissione di una particella alfa (protoni in blu, neutroni in rosso)

Tali particelle sono fortemente ionizzanti e possono disturbare un dispositivo in funzionamento creando una cascata di coppie elettroni-lacune nel silicio. Queste cariche possono essere raccolte da una giunzione pn, producendo una variazione del valore di corrente (*current spike*) all'interno del circuito che, se abbastanza grande, si potrá trasformare in un'alea.
Si noti che, al contrario dei raggi cosmici che possono viaggiare per migliaia di anni luce, le particelle alfa interagiscono fortemente con la l'ambiente che le circonda e vengono facilmente assorbite dai materiali, premettendole di viaggiare soltanto per pochi centimetri nell'aria: difatti esse sono molto deboli ed é sufficiente, ad esempio, la pelle dell'uomo o un semplice foglio di carta per fermarle. Lo stesso non vale (come intuibile) per i raggi cosmici che, invece, necessitano vari metri di cemento per essere fermati. Come accennato nel paragrafo precedente, il flusso di paricelle alfa non varia in base all'altitudine ma é solamente in funzione del tipo, dell'allocazione e dell'ammontare delle impuritá radioattive presenti nel chip e nel suo package.

1.4 Tipologia di soft error

Seppur un singolo soft error possa potenzialmente portare ad un errore, generalmente é necessario che piú nodi varino il proprio valore per ottenere almeno un errore tangibile a livello applicazione. Difatti, errori introdotti a livello di chip spesso non hanno alcun effetto nel sistema.
Il rapporto tra soft error e errori effettivi nel sistema viene detto *system derating*, esso é l'inverso del *architectural vulnerability factor* (AVF).
Come evince il grafico 1.4, un soft error a livello macchina puó: svanire, essere successivamente corretto dal sistema, risultare in un evento *machine check stop* o portare ad uno stato di errore. Lo svanimento puó avvenire, ad esempio, quando il dato corrotto viene sovrascritto oppure quando la corruzione avviene in una cella di memoria non utilizzata dalla macchina, come vedremo piú avanti ci sono vari metodi risolutivi per l'eliminazione di un soft error e, in questo caso, si parla di un "soft error corretto" (nel senso che é stato individuato e il dato riportato al suo valore originale). Nel caso di una *machine check stop*, l'errore viene individuato da un circuito secondario che ha l'esclusivo scopo di sondare i valori del circuito

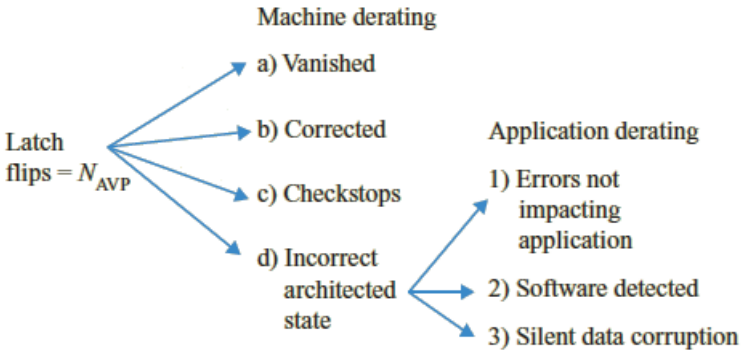

Figura 1.4 – Classificazione dei soft error

e fermare la macchina nel caso di determinazione di un errore: qui, in base alle impostazioni del proprio sistema, la macchina puó riavviarsi o agire in altro modo. Infine abbiamo il raro caso in cui l'errore non viene catturato e, secondo la macchina (almeno fino a questo livello) tutto funziona perfettamente.
A livello di applicazione, ossia quando l'errore é riuscito a sfuggire ai vari controlli del livello macchina, ci si trova d'innanzi a tre possibilitá: l'errore non infierisce l'applicazione (caso fortuito), l'errore viene individuato via software (qui sará il processo a decidere come procedere) oppure, nel caso peggiore di tutti, l'errore non viene individuato nemmeno qui e anche il software agisce come se nulla fosse accaduto: in questa situazione l'errore viene denominato *silent data corruption* (SDC) ed é una grave minaccia alla correttezza dei dati dell'utente finale.

1.5 Casi critici

Ci sono alcune applicazioni in cui i soft error (o qualsiasi altro tipo di errore) non possono essere ammessi: é soprattutto per costoro che si portano avanti gli studi sui soft error. Si pensi al giá citato esempio delle transizioni bancarie, al controllo motori di un aereo (o, peggio ancora, dell'ormai dismesso Space Shuttle e i suoi successori), ai sistemi che gestiscono un acceleratore di particelle o anche ad un piú semplice pacemaker: in ognuno di questi casi un qualsiasi errore puó provocare la perdita di vite o ad una sostanziale perdita di risorse umane.

1.6 SOI Transistor vs. Bulk Transistor

Sebbene entrambe le tipologie di transistor siano in circolazione da molto tempo, con ogni probabiltá nelle prossime generazioni di circuiti prevarrá la tecnologia CMOS SOI (*Silicon-on-insulator*). La differenza sostanziale sta nel materiale di partenza: i transistor SOI sono costruiti in uno strato molto sottile di silicio applicato sulla superficie di uno strato isolante di SiO_2 (si faccia riferimento alla figura 1.5). I vantaggi principali sono la riduzione

delle capacitá parassite, la riduzione dei disturbi dal bulk e le migliori caratteristiche come interruttore (risultando in un transistor piú performante e meno sofferente dell'effetto di canale corto in particolare con riguardo ai consumi). É stato dimostrato che, a paritá di caratteristiche, un transistor SOI fornisce un miglioramento del 22% nelle prestazioni rispetto alla controparte bulk.

Oltre al lato delle prestazioni, i transistor SOI si comportano molto meglio anche per

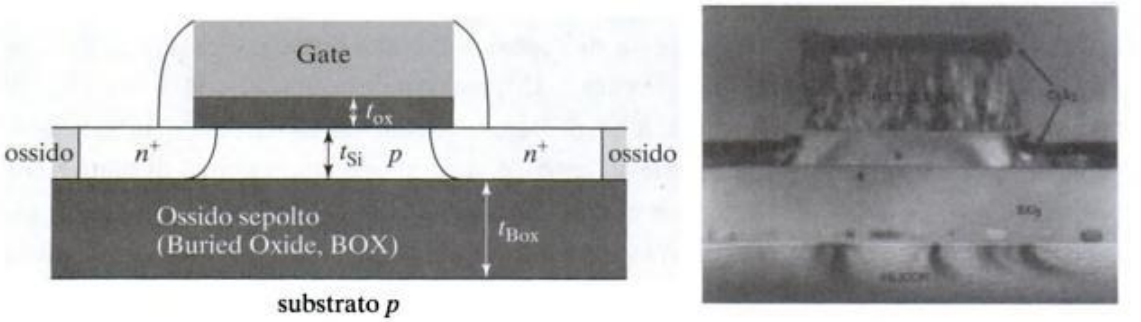

Figura 1.5 – A sinistra abbiamo uno schema semplificato del transistor SOI mentre a destra abbiamo un'istantanea di tale transistor ottenuta tramite un microscopio elettronico a scansione (SEM)

quanto riguarda i soft error: é stato provato che la tecnologia SOI sia dalle 2 alle 6 volte meno sensibile rispetto la controparte bulk. Nella tecnologia bulk l'incanalazione della corrente (evento conosciuto come *funneling*) si verifica in un tempo dell'ordine dei picosecondi (qualche decina al massimo): la carica non raccolta in questo intervallo di tempo é libera poi di diffondere nel chip, parte di essa fluirá nella zona di svuotamento del transistor (entro le successive decine e/o centinaia di picosecondi) causando un transitorio della tensione, per esempio nel nodo di drain dei MOS di una SRAM. Questo meccanismo, nel nodo di drain di MOS di una SRAM, é la causa tipica di soft error.

Nei transistor SOI l'unico contributo per l'evento di soft error é la carica generata nel

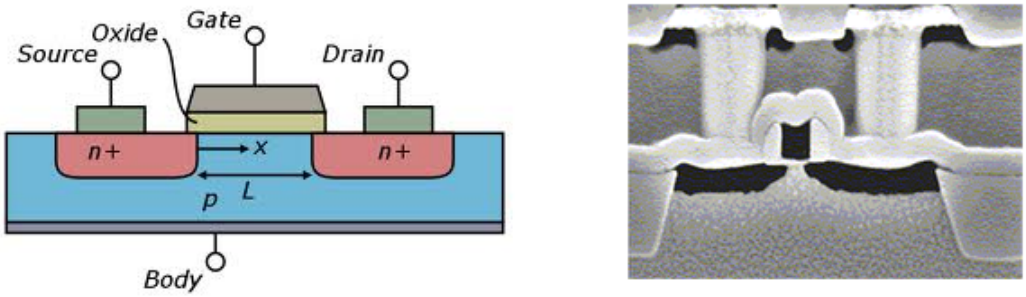

Figura 1.6 – A sinistra abbiamo uno schema semplificato del Bulk transistor mentre a destra abbiamo un'istantanea di tale transistor ottenuta tramite un microscopio elettronico a scansione (SEM)

sottile strato di silicio: questo implica che il numero di elettroni potenzialmente dannosi in questa tecnologia é molto minore. Seguendo lo stesso procedimento del transistor bulk, questa volta la (poca) carica libera raccolta ha un piccolo effetto nel nodo di drain mentre si avranno effetti maggiori (causanti soft error) se le cariche vengono raccolte nel substrato. Difatti allimpatto della radiazione si ha una velocissima creazione di coppie elettrone-lacuna (< 0.1ps) con una disposizione spaziale a forma di cilindro attorno al percorso dello ione che impatta, e densitá elevata. Quando la traccia dello ione arriva vicina (o attraversa) alla regione di svuotamento, i portatori vengono rapidamente raccolti dal campo elettrico, creando un ampio transitorio di corrente/tensione a quel nodo, questo evento viene chiamato innesco del BJT parassita. Si noti che, comunque, gran parte dei modelli di transistor SOI non presentano un nodo nel substrato.
Il motivo principale del ritardo alla diffusione su larga scala della tecnologia SOI é dovuto al maggior costo di produzione; peró, grazie all'innovazione tecnologica, questo sta diminuendo sempre piú, dando la possibilitá anche a questa tecnologia di una adozione su larga scala nel mondo dell'elettronica.

1.7 Flip-flop vs. celle SRAM

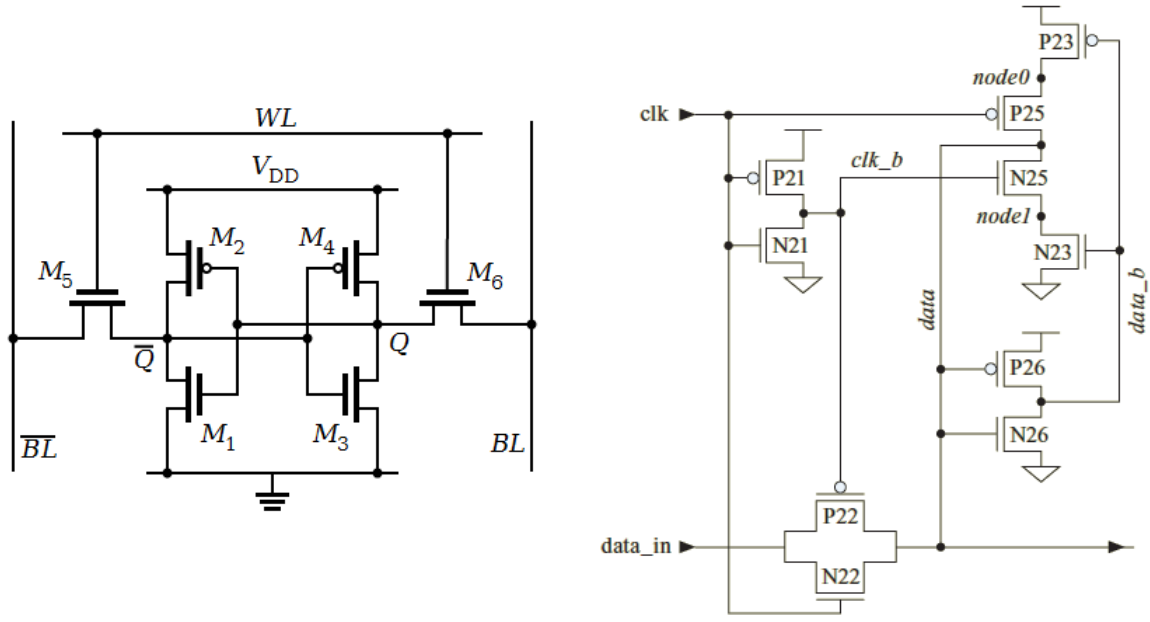

Figura 1.7 – Le due strutture a confronto: a sinistra la cella SRAM, a destra un tipico flip-flop statico

Una cella di memoria SRAM (Fig 1.7) é composta da sei transistor, due (M_5 e M_6) dei quali servono per accedere al "nucleo" di memorizzazione vero e proprio, composto da 2 n-MOS (M_1 e M_3) e 2 p-MOS (M_2 e M_4) connessi in modo da formare una coppia di

inverter (che crea un loop rigenerante il dato). Se WL é imposto a tensione nulla, non sono in corso operazioni di scrittura, il nucleo é isolato e sta effettivamente conservando il dato. Supponiamo ora di avere il nodo $\overline{Q}$ ad "1": quando uno ione lo attraversa, esso genera un cilindro di coppie elettroni-lacune lungo il suo cammino, coppie che vengono separate dal campo elettrico. La raccolta degli elettroni causa una caduta del potenziale di tale nodo; il verificarsi o meno del bit-flip dipende dalla velocitá di compensazione del p-MOS con una corrente di lacune. Se non é sufficientemente veloce si innesca la variazione degli stati di accensione dei transistor, con conseguente variazione del dato memorizzato, e quindi errore. Diversamente dalla celle SRAM , i flip flop non sono simmetrici. Inoltre, i circuiti flip flop presentano delle serie di transistor con nodi interni (*node0* e *node1* in figura 1.7) e un gate di trasmissione (raffigurato come un transistor p-MOS *P22* in parallelo con un n-MOS *N22*). Le coppie di transistor P23-N23 e P26-N26, mutualmente incrociate, possono propagare l'effetto di una radiazione fino ai nodi critici *data* e *data_b* (ossia i nodi di output ove il dato é letto direttamente o in complemento, rispettivamente). Supponiamo ora di trovarci in uno stato di memorizzazione (segnale di *clk* a livello basso) con il dato salvato (nel nodo *data*) ad "1", che il segnale d'entrata *data_in* sia attualmente a "0" e che uno ione attraversi il gate del transistor n-MOS N22. Se lo ione ha sufficente energia, il transitor n-MOS N22 ha la possibilitá di trasferire il valore di data_in all'interno del flip-flop attivando: prima l'inverter formato dalla coppia di transistor P26-N26, poi la coppia P23-N23; completando, cosí, un ciclo di memorizzazione non desiderato conclutosi con il flip del valore salvato.

1.8 Unitá di misura

Per caratterizzare la sensibilitá di un circuito alle radiazioni ci sono due quantitá di misura complementari: il *single event upset* (SEU) *cross section* (sezione d'urto per SEU, cm^2) ed il *SEU rate*, meglio noto come *soft error rate* (SER). Oltre a questi, abbiamo anche il *linear energy transfer* (LET), rappresentante la misura di energia trasferita alla materia da parte di una radiazione ionizzante.

1.8.1 Linear Energy Transfer vs. Stopping power

Misurata in kiloelettronvolt per micrometro [Kev/μ], la *linear energy transfer* (LET) indica la quantitá d'energia trasferita al materiale durante il suo attraversamento da parte di una particella ionizzante. LET é strettamente relazionato alla *stopping power* (potere frenante) del materiale. Il potere frenante misura la perdita d'energia di una particella per unitá di distanza, [dE / dx], durante il suo attraversamento in un determinato materiale. Mentre la stopping power si concentra nella perdita, LET si interessa del trasferimento d'energia al materiale attorno al percorso della particella.

1.8.2 SEU cross section

Il *single event upset cross section* é un parametro intrinseco di un chip, che specifica la sua sensibilitá, o meglio la sua area sensibile, nei confronti di una specifica radiazione incidente (neutroni, protoni o ioni a energia singola, flusso integrale, ecc): esso viene misurato usando

un fascio di particelle prodotto da una opportuna sorgente, quale un acceleratore (per un approfondimento, si veda il paragrafo 2.1). Il valore di questo parametro varia in base al tipo di particella utilizzata, alla sua energia, al suo LET e sará funzione di vari paramentri del chip (valore della tensione d'alimentazione, temperature etc.). Le unitá di misura comunemente usate sono cm^2/Mb o cm^2/MB oppure $cm^2/dispositivo$.

1.8.3 SER

Il termine *soft error rate* (tasso di soft error al secondo) rappresenta il tasso d'occorrenza di un soft error in un dispositivo o in un sistema. Tale tasso viene anche espresso in *failures-in-time* (FIT), ossia il numero di soft error incontrati in un miliardo di ore di attivitá del dispositivo, oppure in *mean time between failures* (MTBF), ossia tempo medio tra un soft error e quello successivo (di solito l'unitá di misura sono anni di operazione). Per dare una prospettiva, 1 anno MTBF equivale a circa 114,077 FIT.
Nonostante alcuni dispositivi presentino un MTBF che eccede il tempo di vita del circuito, il SER ad esso associato puó ancora risultare inaccettabile dal cliente che vuole farne uso in attivitá critiche. Si pensi che un soft error, avvenuto anche in pochissimi prodotti, puó risultare ridicolmente catastrofico e rovinare la reputazione di una societá.

Capitolo 2

Analisi

Per studiare la reazione di un chip alle sorgenti maggiori di soft error prima (o anche dopo) della sua immissione nel mercato, bisogna effettuare dei test in alcuni laboratori con attrezzature adatte. Tramite queste analisi, si vuole soprattutto determinare sperimentalmente il valore del SER associato ad un particolare chip.

2.1 test accelerato e non accelerato

Ci sono due metodi fondamentali per determinare il SER di un prodotto:

a. il primo é di testare un gran numero di dispositivi per un periodo abbastanza lungo (settimane o mesi), raccogliendo un numero sufficiente di dati, da cui trarre delle stime affidabili: si noti che se questo test fosse eseguito su un unico dispositivo, sarebbero richieste alcune decadi prima di ottenere dei dati utili. Questo metodo viene chiamato *real-time testing* o test non accelerato (*unaccelerated SER testing*). I vantaggi di tale metodo stanno nell'analizzare i soft error reali sul prodotto finale, senza la necessitá di utilizzare risorse artificiali di particelle energetiche. Gli svantaggi, invece, sono nel necessitare di costose apparecchiature, in grado di sorvegliare centinaia o migliaia di prodotti contemporaneamente per tempi molto lunghi. Per migliorare la statistica, si seguono i test ad elevate altitudini;

b. L'alternativa comunemente utilizzata sta nell'ottenere stime del SER piú rapide, utilizzando fonti artificiali di particelle, qui si ha anche la possibilitá di scegliere il tipo di particelle che si vuole emettere. Questo metodo viene chiamato *accelerated-SER* (ASER) *testing*. In questa tipologia di test, il prodotto viene esposto ad un tipo specifico di radiazione artificiale con un'intensitá molto maggiore a quella che si trova in natura, consentendo cosí di usare meno prodotti per l'analisi e di ridurre il tempo di test ad una frazione del tempo, altrimenti necessario nella situazione non accelerata: poche ore o al piú qualche giorno.
   Lo svantaggio di questa tecnica sta nel fatto che le stime devono essere estrapolate con dei modelli e, talvolta usando piú tipi di irradiazioni (eseguendo quindi piú test), per essere sicuri che il SER ottenuto consideri sia i soft error causati da particelle alfa sia quelli da raggi cosmici;

2.2 test dinamici e statici

In base all'obiettivo di un'operazione di test, si puó scegliere tra eseguire un test statico o dinamico indipendentemente dalla natura accelerata o meno del test stesso.
Nel test statico il dispositivo é inizializzato in un certo stato conosciuto, dopodiché esso verrá irraggiato. A radiazione avvenuta si legge lo stato del dispositivo e lo si compara con quello pre-irraggiamento in modo da determinare quali valori sono stati cambiati.
Il test dinamico si differenzia dal precedente perché, durante l'irradiazione, i valori salvati nel dispositivo continuano ad essere letti ad intervalli regolari contando i vari cambiamenti di stato *on the fly*. A fine irraggiamento, o dopo passi prefissati, si fa un'ulteriore lettura completa per verificare che ogni cambiamento di valore sia stato memorizzato. Generalmente, ad ogni accesso effettuato durante la radiazione, il dato viene nuovamente riscritto in modo da correggere eventuali alterazioni di dati, e per ottenere un risultato del test piú proficuo.

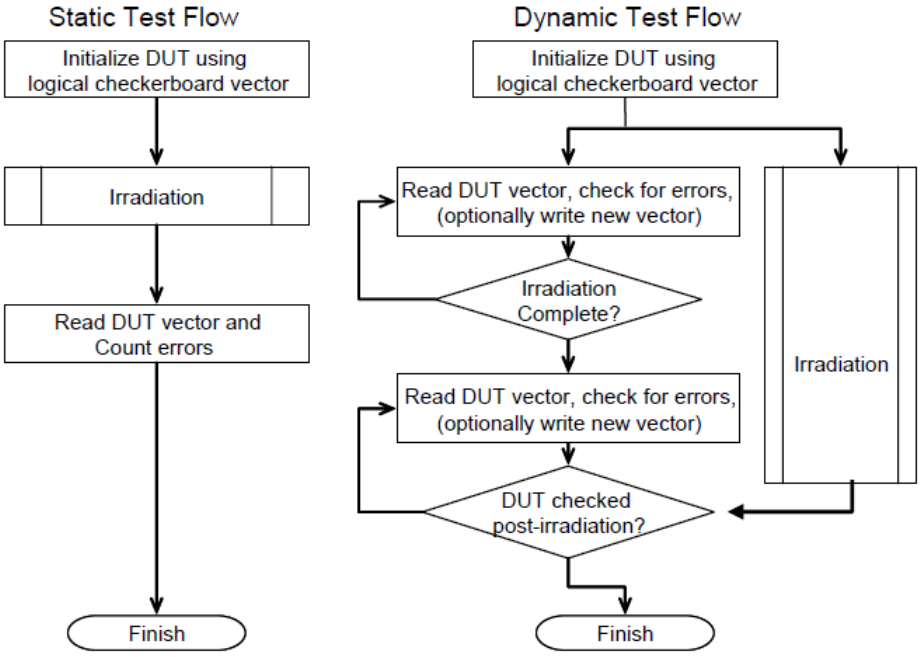

Figura 2.1 – Schemi a blocchi confrontanti i due tipi di test. (DUT: *device under test*)

Si osserva che, nel test dinamico, bisogna tenere nota dei tempi morti che avvengono tra l'accesso del dato e la riscrittura dello stesso perché, nel malaugurato caso avvenga un soft error in questo intervallo di tempo, esso non verrá memorizzato. Nel rapporto finale del test dovranno essere inclusi la natura di questi tempi morti, onde evitare una stima al ribasso del SER.

2.3 Ambienti di test

Negli strumenti fino ad oggi utilizzati per eseguire i test logici descritti sui DUT (chiamati *automatic test equipment*, ATE), si é in grado di distinguere un soft error indotto da una particella alfa da uno indotto da un raggio cosmico. Lo studio specifico dei SER per ciascun tipo di particella necessita l'esecuzione di tutti i test in piú ambienti: generalmente vengono utilizzate le cave sotterranee per proteggersi dalle air shower, studiando quindi soltanto gli effetti delle particelle alfa. Al contrario, vengono effettuati i test a grandi altitudini per evidenziare l'effetto dei raggi cosmici. In alcuni casi estremi si fa ricorso al lancio in orbita dei satelliti scientifici, per effettuare dei test nello spazio ove i dispositivi sono effettivamente bombardati da protoni ad alta energia e ioni di origine cosmica. Il primo esperimento effettuato su un satellite, negli anni 60', mostr che, mediamente, 1600 $particelle/m^2$ con potenza superiore al 1GeV colpiscono ogni secondo i satelliti orbitanti attorno alla terra. Per effettuare questi test nello spazio interplanetario vengono applicati dei fori all'esterno dell'involucro del carico utile del satellite, ove viene deposto l'IC d'interesse con relativo ATE. Ovviamente nei primi lanci di satelliti l'uomo non aveva a disposizione questo tipo di test, detto*testbench*. Ció nonostante,i programmi spaziali sono (quasi) sempre riusciti a portare satelliti e navette spaziali fuori dall'atmosfera terrestre senza problemi. Le motivazioni di questo successo stanno nel fatto che oltre ai test *reali* (ossia effettuati con gli ATE), si possono fare molti test simulandoli con uno o piú calcolatori. Questi test hanno ovviamente dei difetti, come una incertezza residua e la minor affidabilitá, ma comunque risultano molto utili, visto che anche il tempo richiesto e, soprattutto, il costo per questo approccio é di gran lunga minore dei test reali.
Un caso importante é l'utilizzo del dispositivo *indoor*: in questa situazione (che é la piú comune per la maggior parte degli utilizzi di macchinari industriali) si avranno circa 15cm di spessore di lamiera senza considerare altri materiali come tetti, soffitti e altro, che faranno da scudo per le radiazioni. Questo porta ad una diminuzione media dell'energia o del flusso delle particelle d'ogni genere di un fattore di circa 1.6 . L'uso di grandi quantitá di acciao (come quelle utilizzate nelle imbarcazioni moderne) o di altro materiale con un alto numero atomico (ad esempio materiali utilizzati per le schermature con Piombo) possono diminuire significamente l'energia delle particelle, e quindi ottenere un SER ridotto. Si osservi, tuttavia, che tali materiali possono indurre la generazione di neutroni termici i cui effetti possono essere esaltati da particolari tecnologie, come nel caso dell'uso di B^{10}. Difatti, quando si ha un fascio di neutroni termici incidenti su atomi di boro-10, le reazioni indotte rilasciano delle coppie di particelle raggi alfa (possedente qualche KeV) e ioni di lito (nell'ordine dei MeV) che diffondono energia.

Capitolo 3

Mitigazione del SER

Con il passare degli anni, si sono trovate piú metodi per diminuire il SER di un determinato chip, ossia la mitigazione del SER: si puó spaziare dalle soluzioni presenti nel circuito elettronico stesso, fino a quelle software. Nelle applicazioni critiche si useranno persino delle combinazioni di esse: in questo capitolo affronteremo le piú diffuse, confrontandole e descrivendo pro e contro di ciascuna di essa.

3.1 Aumento delle capacitá: Radiation Hardening

La soluzione forse piú intuitiva sta nell'aumentare le capacitá dei nodi d'interesse, in modo tale che la carica critica Q_{critic} sia piú elevata. Ovviamente, questo approccio, sebbene riduca effettivamente il SER in modo sostanzioso, porta con sé lati negativi: l'aumento delle capacitá comporterá un aumento della tensione di alimentazione, se si vuole ottenere un livello di performance pari al circuito senza radiation hardening. Se la performance non é un problema, la tensione richiesta per far variare un valore in un nodo viene aumentata, risultando in un maggior consumo energetico complessivo. Il raggiungimento di una capacitá maggiore puó essere ottenuta in vari modi: 1. incrementando la dimensione del transistor; 2. aggiungendo capacitá passive; 3. cambiando il layout del transistor.

3.2 Riduzione dei soft error causati dalle particelle alfa

Per questo tipo di problemi ci sono tre soluzioni:

- la prima sta nell'usare materiali purissimi (privi, o quasi, di impuritá), che evitino il piú possibile le emissioni di radiazioni alfa;
- un'altra soluzione é di mantenere distante il packaging dalle zone piú sensibili del circuito;
- l'ultimo approccio sta nello schermare il chip con delle pellicole in vari materiali. Conoscere l'origine e l'energia delle emissioni alfa é un nodo cruciale per l'utilizzo di queste pellicole: tali pellicole possono essere posizionate direttamente sui collegamenti

del chip o nel package. Un posizionamento errato puó portare ad un innalzamento del SER anzhiché ad una diminuzione.

3.3 Layer design

Sebbene sia piú un problema di simulazione, che una soluzione per diminuire il SER di un circuito, si osservi lo schema 3.1 relativa ad un chip SOI: esso raffigura una tipica sezione di

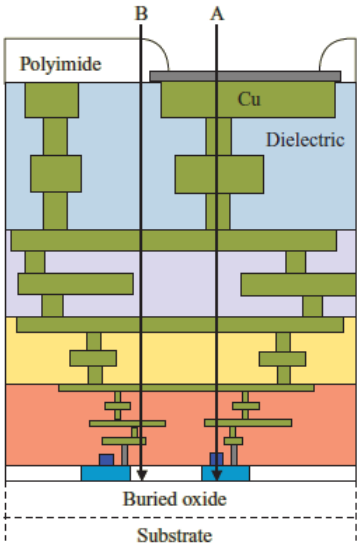

Figura 3.1 – schema rappresentante la sezione di una parte finale (i livelli piú bassi) di un circuito

un circuito, ove sono disegnate due tracce, A e B, indicanti due possibili traiettorie seguite da delle particelle. Le traiettorie sono idealmente rettilinee. Tale approssimazione é lecita per particelle energetiche, che concluderanno la loro traiettoria lontane dalle zone sensibili del chip. Nel percorso A, la particella attraversa piú metallo rispetto a quello trovato nel percorso B: questo significa che A attraversa strati con *stopping power* maggiore (si faccia riferimento al paragrafo), dato che nel dielettrico é minore la stopping power, essendo materiali a inferiori.
Supponiamo ora di avere una particella identica che attraversa i due percorsi: essa arriverá nella superficie dell'ossido con un'energia molto differente.
Sebbene questo sia, come giá osservato, piú un problema di simulazione degli effetti di una particella in un chip, piuttosto che una soluzione alla diminuzione del SER, si potrebbe comunque usufruire di questo punto di vista per ottenere un lieve miglioramento del SER almeno nelle parti di circuito piú sensibili, con opportune schermature, nota l'origine delle particelle alfa incidenti.

3.4 Ridondanza

Sebbene non siano molto diffuse in ambito terrestre per vari motivi qui esposti, le tecniche di ridondanza possono essere sia spaziali che temporali. Nel mondo dei circuiti integrati esse sono adattate quasi esclusivamente per l'uso di dispositivi al di fuori dell'atmosfera terrestre, o in applicazioni critiche.

3.4.1 Triple Modular Redundancy

La piú semplice ridondanza spaziale é la Triple Modular Redundancy (TMI) che replica due volte (portando ad un totale di tre copie identiche) i nodi critici (latch e flip-flop) del circuito. L'output di questa ridondanza viene deciso tramite un sistema di voting (*voter*) che si intende immune agli errori, ossia assai poco sensibile ai SEU. Se uno dei tre sistemi fallisce, gli altri due sistemi possono mascherare e correggere l'errore. Si osserva che il circuito non corregge l'errore, semplicemente invia nella linea di output il valore corretto. Questa soluzione soffre per la grande richiesta di spazio sul silicio e per l'ampio consumo di energia. Per quanto riguarda la velocitá, invece, la degradazione é minima, visto che al percorso critico per ottenere il dato viene aggiunta solo la circuiteria del *voter*. Un fallimento del voter comporta il fallimento dell'intero sistema: per tale motivo un buon sistema TMR deve possedere un voter con circuito molto piú affidabile degli altri componenti.

3.4.2 HIT, DICE e DICE con Guard-gates)

Altri tipi di ridondaza spaziale per contrastare l'avvenimento di SEU nei latch sono: HIT (figura 3.2); DICE (figura 3.3); Guard-gates DICE(figura 3.4);

La cella HIT

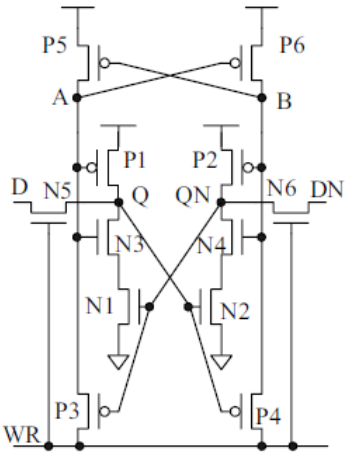

Figura 3.2 – cella HIT

La cella Heavy Ion Tolerant (HIT, figura 3.2) é composta da 12 transistor organizzati come due memorie interconnesse da sei cammini di feedback. Per le operazioni di scrittura si ha la necessitá di avere l'ingresso sia diretto sia in complemento. Per quanto riguarda la lettura, invece, si avranno a disposizione sia l'output diretto che negato.
Uno svantaggio della cella HIT risiede nelle dimensioni dei transistor: esse risultano critiche per quanto riguarda il ripristiare il valore corretto nel nodo in cui é avvenuto il SEU.
La cella ha quattro nodi sensibili: facendo riferimento alla figura3.2, questi nodi sono Q, QN (ossia ambo gli output) e i nodi interni A e B. Se soltanto una particella attraversasse uno di questi nodi causando un upset, la correzione avverrebbe immediatamente, grazie al contributo degli altri tre nodi. Se, al contrario, si verificassero due o piú upset, il dato salvato nella cella sará corrotto.

DICE

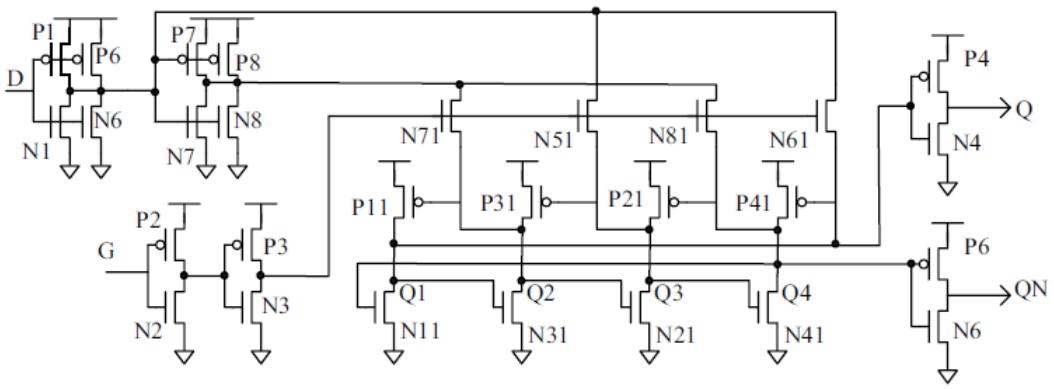

Figura 3.3 – Latch utilizzante la tecnica di SEU-hardening DICE

Il latch sfruttante il Dual Interlocked Circuit Element (DICE, figura 3.3) utilizza ben 28 transistor, sebbene alcuni di loro siano delle coppie di inverter, utilizzate sia come buffer CMOS (P2-N2 e P3-N3) sia per rendere piú veloce la commutazione del segnale d'ingresso (come in P1-N1 e P6-N6 oppure in P7-N7 e P8-N8)). Per la scrittura del dato, questa volta, sará necessario solamente l'ingresso diretto, visto che l'ingresso negato viene generato all'interno del circuito. Anche qui si avranno a disposizione entrambe le uscite dirette e negate. Anche in questo caso i nodi sensibili sono quattro: facendo riferimento alla figura3.2, essi sono Q1, Q2, Q3, Q4. Come per la cella HIT, anche qui si ha un'ottima risposta del circuito nel caso di SEU in uno solo dei nodi sensibili: la correzione sará persino piú veloce. Mentre si otterá una corruzione del dato nella circostanza in cui si verificassero SEU in due o pi nodi.

Guard-gates DICE

Il guardian-gates DICE (figura 3.4) utilizza solamente 16 transistor consistenti in una serie di interconnessioni di quattro guard-gates, anche conosciuti come Transition AND Gates (TAGs). Al contrario delle versioni precedenti, la linea di input combacia con la linea di

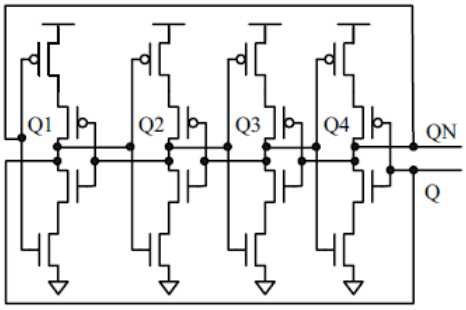

Figura 3.4 – DICE con guard-gates

output, ed entrambe sono disponibili sia diretta che negata: per far un uso profiquo del circuito, si utilizzerá una delle due linee come ingresso e l'altra come uscita, tenendo conto che l'output sará sempre la negazione del dato salvato.
Per quanto riguarda la resistenza alla corruzione, guard-gates DICE si comporta meglio delle precendenti configurazioni. L'unica via per ottenere un errore sta nel suscitare SEU in almeno tre dei quattro nodi sensibili (Q1, Q2, Q3 e Q4 in figura 3.3), evento alquanto raro.

3.4.3 Ridondanza temporale

Le tecniche di ridondanza temporale si basano sull'estrapolazione dell'input in intervalli di tempo maggiori, rispetto alla durata di una transizione ordinaria, e poi votare se il dato é stato corrotto oppure no. Lo schema (si faccia riferimento alla figura 3.5) di questa ridondanza introduce tre segnali di clock (CLKA, CLKB e CLKC), per leggere il dato in diversi intervalli di tempo, e un quarto segnale (CLKD) per la determinazione (ed eventuale riscrittura) del dato. Il circuito comprende nove latch sensibili al livello di clock alto (da U1 a U9), tre invertitori (da U11 a U13) e un voter (anche detto majority gate, U10). Due latch *level-sensitive* in serie con clock complementari (come la coppia U1-U2) formano un edge triggered DFF (*D-Flip-Flop*).
I tre DFF (formati dalle coppie U1-U2, U3-U4 e U5-U6) operano in parallelo. Inoltre, essi formano la circuiteria di sampling nelle tre fasi temporali precedentemente descritte. Infine, i tre latch residui (da U7 a U9), assieme al voter (U10) compongono un circuito TMR che deterá l'output finale. Questa tipologia di ridondanza temporale rappresenta solamente la versione base, facilmente ingannata in caso di semplici transizioni del segnale d'interesse, specialmente in circuiti asincroni. Si osserva che, con l'implementazione del circuito TMR, si vanno ad aumentare area e consumi energetici e, per di piú, anche la velocitá totale del circuito viene lievemente penalizzata a causa dell'aggiunta del circuito di voto.

3.5 ECC

Il metodo di mitigazione microarchitettuturale piú diffuso ed efficace é il *error checking and correction* (ECC). ECC varia molto in base a cosa deve gestire: ci si muove da un semplice controllore di bit di paritá, in un array di SRAM, fino a sistemi molto piú complessi, per

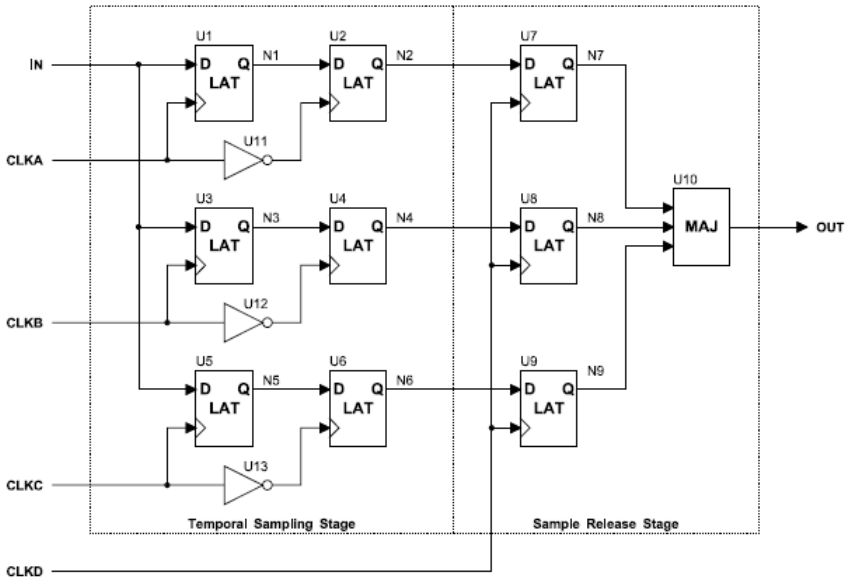

Figura 3.5 – Latch temporale generico

gestire banchi di dati di grosse quantitá. Questa tecnica sfrutta la trasmissione dei dati tra i vari circuiti: durante la trasmissione, insieme all'informazione originale, vengono trasmessi anche dei dati ridondanti, ovvero aggiuntivi rispetto all'originale. Il ricevitore, elaborando la comunicazione ricevuta, puó quindi effettuare controlli sull'integritá dei dati e, se rileva uno o piú errori, puó tentare di correggerli basandosi sulle informazioni veicolate dagli elementi ridondanti. Ovviamente, con questa tecnica non é possibile recuperare qualsiasi tipo di dato, se il segnale risulta essere particolarmente corrotto. Per ogni circuito esisterá una soglia critica, oltre la quale il dato risulterá non piú correggibile. Si osserva che, la maggioranza degli algoritmi ECC puó correggere i *single-bit error* ed individuare, ma non correggere, i *double-bit error*. Tali sistemi sono chiamati *single-error-correcting and double-error-detecting* ECC (SEC/DED ECC). Tramite essi, é possibile correggere gran parte degli errori: difatti, per le memorie SRAM, quasi il 100% degli errori sono *single-bit error*, mentre, nelle memorie DRAM, i *single-bit error* arrivano al 94% degli errori elettronici totali. L'efficienza di ECC varia in base al rapporto tra il numero di bit usati per trasmettere i dati e il numero di bit usati per il controllo: tanto piú alto é il rapporto, tanto piú i dati resisteranno alla corruzione. Si osserva che piú bit saranno usati per il controllo, piú tempo impiegherá il circuito a trasmettere i propri dati visto che, oltre ad essi, si dovranno aggiungere ogni volta anche quelli di controllo, senza considerare che c'é anche un tempo necessario al calcolo di questi bit di controllo.

3.5.1 Esempio pratico

Supponiamo, ora, di possedere un circuito desiderante di trasferire un messaggio, *love*, ad un'altro circuito. La codifica utilizzata per inviare il messaggio é lo standard *American*

Standard Code for Information Interchange (ASCII) che, convertito in binario risulta:

01101100	01101111	01110110	01100101
l	*o*	*v*	*e*

Ogni lettera é rappresentata da 8 bit: la codifica si legge da sinistra verso destra, quindi il primo byte corrisponde alla lettera *l*, il secondo alla lettera *o* e cosí via.
Abbiamo piú possibili soluzioni per inviare il messaggio.

Nessun algoritmo

Nel caso non volessimo usare alcun tipo di ECC, il messaggio sará inviato cosí comé. Mentre questa soluzione non aggiunge nessun costo di codifica e decodifica, se non per il messaggio vero e proprio, c'é un brutto svantaggio: il destinatario non saprá mai se il messaggio ricevuto sia lo stesso inviato dal mittente oppure no, perché c'é alcun modo per verificarlo. Se vi avviene uno o piú bit-flip durante la comunicazione del messaggio, nessuno lo potrá sapere.

Ripetizione del messaggio

Una tecnica lievemente migliore della precedente, sta nell'inviare tre volte il messaggio, in intervalli di tempo differenti. Ricevute le tre comunicazioni, il destinatario potrá simulare un voter, similmente ad un circuito TMR e, quindi, decidere qual'é il vero messaggio inviato dal mittente.
Ancora una volta, non si potrá affermare con certezza se il messaggio scelto dal voter sia identico all'originale. Il messaggio ottenuto, tramite questa soluzione, ha piú probabilitá di essere corretto, rispetto la precedente tecnica. Questa nuova metodologia spreca molto tempo e non viene, attualmente, implementata in alcun circuito reale.

Paritá

Una tecnica un pó piú rassicurante delle precedenti, sta nell'utilizzare il *parity bit* (bit di paritá): si prende il messaggio originale codificato, a cui si aggiunge un nuovo bit alla sua fine, chiamato bit di paritá. Esso rispecchierá la paritá del numero di cifre uguali a 1. Se il messaggio originale contiene un numero pari, zero compreso, di "1", il parity bit sará imposto a 0, sará imposto ad 1 altrimenti. Nel nostro caso, il messaggio contiene diciannove "1" e, quindi, il *parity bit* sará imposto ad 1. Il messaggio finale inviato sará:

01101100	01101111	01110110	011001011	**1**
l	*o*	*v*	*e*	*paritá*

La debolezza dello schema di paritá, sta nel fatto che il destinatario potrá scovare immediatamente se ha avuto luogo un *single-bit error*, ma non potrá individuare quale bit é stato effettivamente corrotto. Nel piú sfortunato dei casi, si avrá un bit-flip nel *parity bit*, ove il destinatario si vedrá costretto a gettare il messaggio originale perfetto. Si osserva che, nel caso di *double-bit error*, il destinatario riceverá un messaggio che supera indenne il test di paritá, seppur corrotto. Se si verificassero piú di due bit-flip, il messaggio ricevuto dal destinatario potrá risultare corrotto oppure no, rendendo questa tecnica inefficiente e inaffidabile.

Algoritmo di Hamming

Una prima soluzione che viene implementata nei circuiti reali é l'algoritmo di Hamming. Facendone uso l'utente finale non solo potrá determinare l'avvenimento di una corruzione ma, inoltre,potrá individuare quali bit sono stati corrotti consentendo il recupero del messagio originale. Dato un messaggio di n bit, l'algoritmo di Hamming richiederá l'invio di almeno $\lceil \log_2 n \rceil + 1$ bit extra per il controllo. Nel nostro caso abbiamo 32 bit, per cui saranno necessari almeno 5 ($= \lceil \log_2 32 \rceil + 1$) ulteriori bit (come vedremo poi, ne serviranno 6).
L'idea di base, sta nel dividere il messaggio originale in circa $\lceil \ln n \rceil$ sottoinsiemi parzialmente sovrapposti in modo tale che ciascuno comprenderá all'incirca metá dei bit del messaggio originale. Ottenuti i sottoinsiemi, si effettuerá il calcolo del bit di paritá su ciascuno di ognuno di loro.
Inizialmente, il messaggio originale verrá copiato all'interno di una stringa vuota, ove si salteranno i bit di posizione corrispondente alle potenze di due, il conteggio dei bit partirá da uno. Nel nostro messaggio *love* avremo:

P*1*P*2**0P*3**110***P*4**1100 011***P*5**01111 01110110 01***P*6**100101

I bit lasciati momentaneamente incogniti (P1, P2, ... , P6) vengono chiamati *check bits* e qui verranno inseriti i risultati dei test di paritá dei vari sottogruppi. Per determinare tali sottogruppi, bisognerá scrivere in binario la posizione di ogni singolo bit della stringa. Dato un bit incognito P**n**, ogni bit del messaggio originale contenente la cifra "1" nell'**n**-esimo bit della sua posizione corrispondente nel nuovo messaggio, fará parte del sottogruppo che determinerá il valore di P**n**.
Ad esempio, il primo sottogruppo, corrispondente al gruppo del bit P**1**, comprenderá i bit:

*P1P2***0***P3***11****0****P4***11****0****0 **0**11*P5***0****1****11 0****1****11****0****11****0 0****1***P6***1****0****0****1****0**1

Difatti, le loro posizioni corrispondenti sono, da sinistra verso destra:

- 0: 3° posizione → 0000**1**
- 1: 5° posizione → 0001**1**
- 0: 7° posizione → 0001**1**
- 1: 9° posizione → 0010**1**
- 0: 11° posizione → 0010**1**
- 0: 13° posizione → 0011**1**
- 1: 15° posizione → 0011**1**
- 0: 17° posizione → 0100**1**
- 1: 19° posizione → 0100**1**
- 1: 21° posizione → 0101**1**
- 1: 23° posizione → 0101**1**
- 1: 25° posizione → 0110**1**
- 1: 27° posizione → 0110**1**
- 0: 29° posizione → 0111**1**
- 1: 31° posizione → 0111**1**
- 1: 33° posizione → 1000**1**
- 0: 35° posizione → 1000**1**
- 0: 37° posizione → 1001**1**

Il secondo sotto gruppo, corrispondente al gruppo del bit P**2**, comprenderá i bit:

$$P1P2\mathbf{0}P3 1\mathbf{1}\mathbf{0}P4 1\mathbf{1}\mathbf{0}0 \quad 0\mathbf{1}\mathbf{1}P5 0\mathbf{1}\mathbf{1}11 \quad \mathbf{0}\mathbf{1}11\mathbf{0}\mathbf{1}10 \quad \mathbf{0}\mathbf{1}P6 1\mathbf{0}\mathbf{0}10\mathbf{1}$$

Ove, le loro posizioni in binario di ciascun bit saranno:

- 0: 3° posizione → 0001**1**
- 1: 6° posizione → 0001**0**
- 0: 7° posizione → 0001**1**
- 1: 10° posizione → 0011**0**
- 0: 11° posizione → 0011**1**
- 1: 14° posizione → 0011**0**
- 1: 15° posizione → 0011**1**
- 1: 18° posizione → 0101**0**
- 1: 19° posizione → 0101**1**
- 0: 22° posizione → 0101**0**
- 1: 23° posizione → 0101**1**
- 0: 26° posizione → 0111**0**
- 1: 27° posizione → 0111**1**
- 0: 30° posizione → 0111**0**
- 1: 31° posizione → 0111**1**
- 0: 34° posizione → 1001**0**
- 0: 35° posizione → 1001**1**
- 1: 38° posizione → 1001**0**

E cosí via.
In fin dei conti, avremo i seguenti sottogruppi:

- P1: 010100101111101100 (dieci "1");
- P2: 010101111010101001 (dieci "1");
- P3: 110001111011001101 (undici "1");
- P4: 110001111011001 (nove "1");
- P5: 011110111011001 (dieci "1");
- P6: 100101 (tre "1");

Sostituendo, ora, le incognite con il valore del parity bit corrispondente, si otterrá il messaggio finale:

$$\mathbf{00}0\mathbf{1}110\mathbf{1}1100 \quad 011\mathbf{0}01111 \quad 01110110 \quad 01\mathbf{1}100101$$

Che verrá inviato al destinatario: una volta completato il trasferimento, il ricevitore potrá individuare un errore, semplicemente guardando quali bit di paritá sono "sbagliati". Questi, messi insieme, formeranno la posizione in binario del bit dove si é verificato il flip: basterá invertirlo per ottenere il messaggio corretto.

SEC / DED

Questa soluzione, considerata una sottoclasse dell'algorimo di Hamming, unisce la tecnica dell'algoritmo di Hamming con quella di paritá. Cosí facendo non solo si é in grado di correggere i *single-bit error* ma anche determinare i *double-bit error*, da questo il nome *single error correcting and double error detecting code* (SEC / DED code).
Per prima cosa, si applica al messaggio originale l'algoritmo di Hamming, poi, si aggiungerá un bit di paritá al nuovo messaggio. Quest'ultimo bit controllerá tutto il messaggio: avremo un *parity bit* controllante il contenuto originale e i check bits assieme (escluso, se stesso).
Nel nostro caso il messaggio da inviare sará:

$$\underbrace{00011101110001100111101110110011100101}_{messaggio\,con\,Hamming}\quad\underbrace{\mathbf{0}}_{paritá}$$

Visto che é preceduto da vendidue "1", il bit di paritá avrá valore zero.
Una volta ricevuto il messaggio, il mittente potrá trovarsi in una delle seguenti condizioni:

- Bit di paritá e *check bits* corretti: l'informazioni ricevuta é corretta;
- Bit di paritá non corretto e *check bits* indicanti un flip: basterá correggere il flip per recuperare il messaggio originale. Si osserva che, se i *check bits* indicano un errore nella posizione 0, é il bit di paritá a non essere corretto.
- Bit di paritá corretto e *check bits* indicanti un flip: é avvenuto almeno un *double-bit error*, il messaggio non puó essere recuperato;

Se ci si trova in casi diversi da quelli descritti, non si potrá recuperare il messaggio.

SSC / DSD

Un'altro metodo ECC ben conosciuto, e diffuso soprattutto nei sistemi digitali, é il codice *single symbol correction and double symbol detection* (SSC / DSD).
Quest'approccio, prettamente architetturale, consiste nel distribuire ogni informazione in celle di piú *words* della memoria, controllate dall'algoritmo SEC / DED ECC. Per questioni di prestazioni, le informazioni di un singolo dato non potranno mai essere sparse in tutto il chip della memoria, bensí saranno distribuite al piú in una *pagina* di essa. Alternativamente, si potrá spezzettare l'informazione su piú chip, similmente a come avviene per gli hard disk utilizzando la tecnica RAID 0 (dall'inglese *Redundant Array of Independent Disks*). Si ricorda che l'architettura di una memoria (con dimensioni, in termini di byte, tipiche dei nostri giorni) é organizzata, partendo dal basso, come: cella (1 o 2 bit, un transistor) $\rightarrow$ pagina (4KB, 4096 o 2048 celle) $\rightarrow$ blocco (512KB, 128 pagine) $\rightarrow$ piano (512MB, 1024 blocchi). Se la limitazione di salvare un'informazione all'interno di una pagina, non fosse rispettata, si potrebbero ottenere dei tempi di accesso molto lunghi, anche 3-4 volte superiore rispetto ad una memoria rispettante tale limitazione.
Questa soluzione ha trovato ampia adozione nei sistemi server *high-end*, ove, nel caso di avvenimento di un hard error, esso, probabilmente, causerá una serie di errori, occupanti diversi bit adiacenti rendendo, per esempio, lo standard SEC / DED ECC inutile.

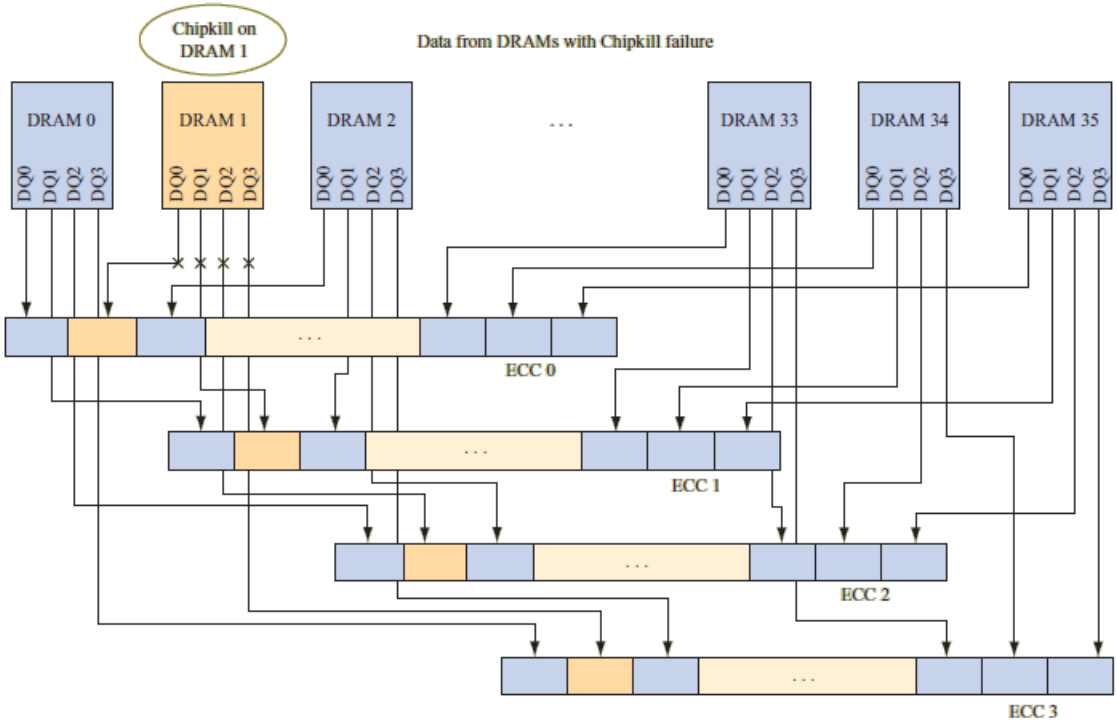

Figura 3.6 – Quattro parole protette da dall'algoritmo SEC / DED ECC distribuite nella memoria DRAM. (DQ: si intendono i data bits della DRAM, cosí chiamati a causa dell'accettato design dei latch ove D é l'input e Q l'output, perció un DRAM DQ é un *data bit* bidirezionale)

Sebbene questa tecnica si comporti egregiamente nel caso di piú upset nella stessa parola, c'é ancora un ultimo volto nel mondo dei soft error da studiare attentamente: l'avvenimento in tandem di soft e hard error.

3.5.2 Affrontare hard e soft error insieme

Questo é il peggior scenario in cui ci si possa trovare, nonchè costante incubo dei sistemi ad utilizzo critico. Un esempio lo si trova facilmente, usando un circuito con memoria DRAM. Supponiamo che avvenga un hard error che colpisce un'unica cella di memoria, e che questo hard error venga mascherato da uno degli algoritmi visti precedentemente. Cosí facendo, tutto procede tranquillamente, fino a quando non si verificherá o un altro hard o un soft error nella stessa parola. Si immagini, ad esempio, di possedere un sistema utilizzante due banchi di memorie DIMM (circuiti integrati di memoria RAM, dall'inglese *Dual In-line Memory Module*), per un totale di 36 dispositivi di memoria DRAM. Tutti questi dispositivi contribuiranno ad ogni ECC word salvata nel sistema. Nel caso di malfunzionamento completo (o in parte), dovuto ad un hard error o ad un qualsiasi altro effetto, di uno qualsiasi di questi dispositivi, l'intero modulo di memoria funzionerá, comunque, impeccabilmente, grazie alle tecniche di correzione affrontate precedentemente. Tuttavia, dopo l'avvenimento di questa rottura, il sistema si ritrova pericolosamente suscettibile ad un hard o soft error in

Tempo prima del riparo(Mesi)	*Memory failure rate* da aggiungere(FIT)
1	102
6	608
12	1207

Tabella 3.1 – FIT da aggiungere al sistema, dopo l'avvenimento di un hard error, in funzione del tempo

tutti gli altri 35 dispositivi. Un tipico dispositivo DRAM dei giorni d'oggi utilizzante SSC / DSD ECC presenta un *memory failure rate* di 13.7 FIT ed un SER di 1000 FIT. Con tutte le assunzioni fatte, si ottengono dei nuovi valori riassunti nella tabella 3.1. Senza errori, un banco di memoria DIMM, possiede un *memory failure rate* di circa 5 - 10 FIT, il sistema intero (a causa di vari componenti) si presenterá un valore dello stesso indice di circa 160 - 320 FIT. Come si vede nella tabella 3.1, piú tempo trascorre dall'avvenimento del hard error piú aumenta il FIT medio abbassando la sicurezza dei dati ad un livello preoccupante. Non esiste un'unica soluzione adottata nel mercato a questo problema. Tutti i produttori usano il personale sistema proprietario opportuamente abbrevettato (IP), i piú noti sono *Chipkill* di IBM, *Extended ECC* di Sun Microsystems, *Chipspare* di HP e *SDDC* di Intel.

3.6 Memoria gerarchica

Tipica di sistemi ad alte prestazioni, questa tecnica fornisce una buona mitigazione del SER: i dati vengono salvati in piú livelli della memoria, in modo tale che, non appena si determini un errore, basterá richiedere una copia del dato originale al livello superiore. Anche questo approccio non porta solo benefici al circuito, ma anche degli svantaggi come, ad esempio, la minor memoria disponibile per l'utente finale (visto che per ogni dato saranno salvate piú copie).

3.7 Introduzione di nuove tecnologie di processo

Una soluzione riguardante la riduzione del SER causato da particelle alfa consiste nell'aggiungere uno o piú nuovi layer prima dell'ultimo strato (anche detto BEOL, *back-end-of-line*) nel layout complessivo del IC. Come rappresentato in figura 3.7, gli approcci principali sono due: nel primo si introduce uno strato di Rame *damascene* sovrapposto al SiO_2 (figura [a]); nel secondo il composto del nuovo strato é invece poliimmide (PSPI, figura [b]).
Il collegamento tra gli strati adiacenti a quello nuovo saranno effettuati tramite dei semplici collegamenti (*via*) in Rame, isolati dal resto del nuovo livello da uno strato di tantalio (Ta) o da nitruro di Tantalio (TaN).
La motivazione che porta al calo del SER, sta nell'aumento generale che si ha nella stopping power dell'IC: difatti, la composizione dei nuovi strati introdotti é stata scelta in modo da massimizzare la stopping power mantenendo un basso costo di produzione. Questa tecnica, inoltre, ha il vantaggio di diminuire il SER indipendentemente dal layout del circuito. Infine,

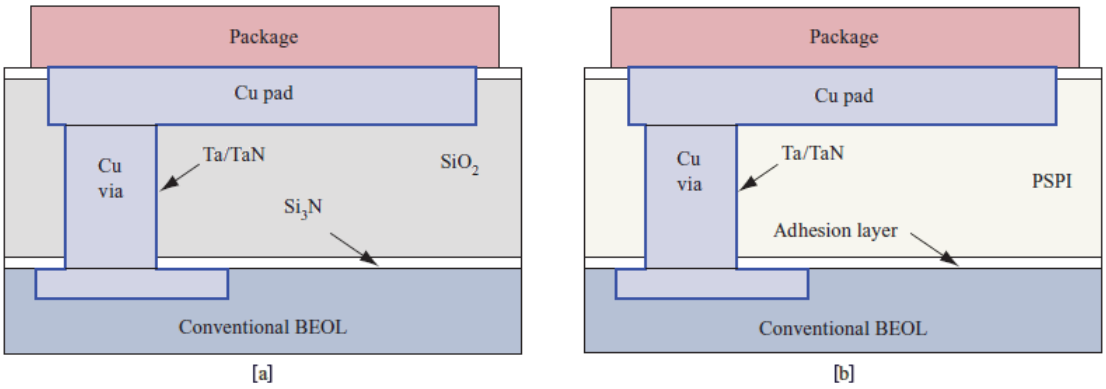

Figura 3.7 – Rappresentazione schematica dei nuovi livelli introdotti sopra il BEOL

l'introduzione dei nuovi layer supplementari non comporta a considerevoli aumenti del costo di produzione.

Capitolo 4

Un test concreto: il processore POWER6

Per un dispositivo elettronico é importante essere sottoposto a dei test che ne dimostrino la soddisfazione dei requisiti per quanto riguarda la sua affidabilitá. Questi esperimenti, per quanto riguardano i soft error, sono molto stimolanti e consentono al produttore (nel nostro caso IBM) di testare le proprie soluzioni e la propria circuiteria per la determinazione dei SER, estraendone dei verbali molto utili anche in vista delle future generazioni di processori e quant'altro. I risultati qui riportati hanno lo scopo di sensibilizzare il lettore agli effetti dei soft error in un moderno processore mentre esegue dei tipici task.

4.1 Un rapido sguardo al processore

Al centro dei testi riportati nelle prossime pagine abbiamo il microprocessore POWER6, progettato da IBM: annunciato a inizio 2006 (sebbene i primi esemplari videro la luce nel 2005), é stato immesso nel mercato server verso la fine 2007.
Si tratta di un processore a 64bit, sucessore del POWER5, realizzato in tecnologia 65nm SOI a 10 strati metallici. Esso ingloba oltre 750 milioni di transistor e presenta un die di 341 mm^2. La frequenza operativa varia in base al modello di POWER6 che abbiamo in un range che inizia da 3.5GHz fino ad arrivare a 5GHz. Il processore é un dual core con 128 KB di cache di primo livello per core (64 KB per le istruzioni e 64 KB per i dati) associativa a otto vie con due pipeline indipendenti per la lettura di due dati o la scrittura di uno solo nello stesso ciclo di clock. Il processore dispone anche di 4 MB di cache semicondivisa di secondo livello, ovvero ciascun core ha una propria cache L2, ma tutti i core possono accedere alla cache L2 degli altri tramite un bus. I due core condividono ulteriormente una cache di terzo livello da 36 MB su un secondo chip, collegata al processore tramite un bus di banda a 80 GB/s. Internamente il processore ha un bus da 300 GB/s tra i core e la cache interna. Per quanto riguarda i soft error, POWER6 utilizza ECC (per approfondire, ci si rechi al paragrafo 3.5) in tutta l'architettura oltre a Chipkill (trattato nel paragrafo 3.6) per proteggere le cache di secondo e terzo livello. La cache di primo livello usa la tecnica *store-through*, ossia nel caso in cui una word deve essere trasferita dalla CPU alla memoria,

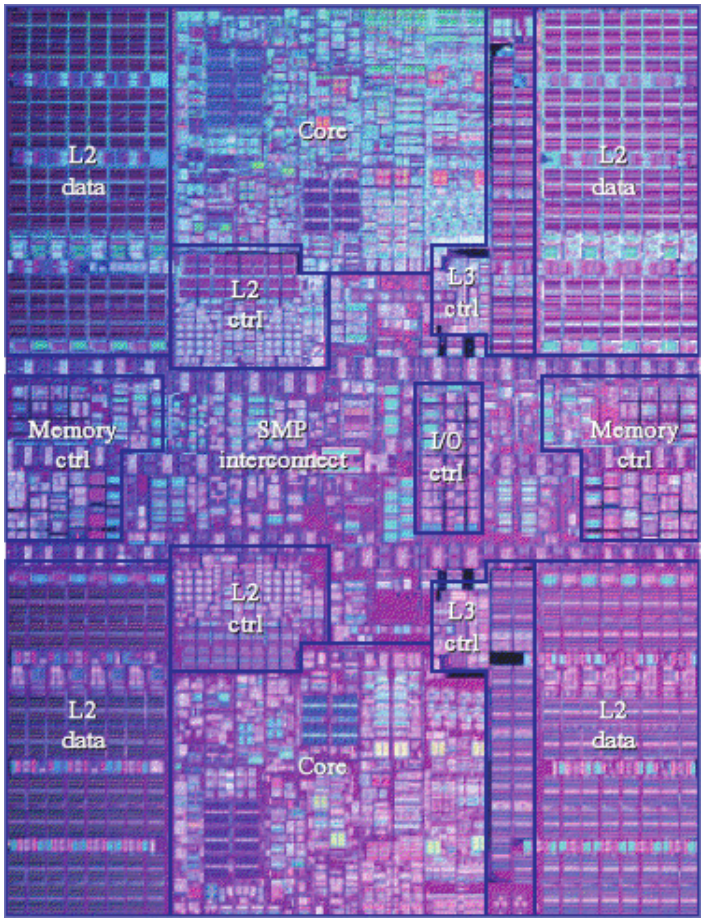

Figura 4.1 – die del POWER6 chip

tale word verrá scritta sia nella memoria cache sia nella memoria principale, eliminando cosí il problema di recupero di un dato nel caso di corruzione della cache.

4.2 Sensibilitá dei latch e dei circuiti di memoria

Per ottenere un'idea della sensibilitá del processore alle particelle, sono stati eseguiti una serie di esperimenti utilizzando un acceleratore producente un raggio monoenergetico di particelle alfa chiamato *3-MV Tandem Van de Graaff accelerator* (VdG). La figura 4.2a illustra i risultati ottenuti servendosi di un raggio di particelle alfa con 5.28 MeV (million electronvolt) d'energia in una catena di latch *master-slave*: gli esiti del test sono visualizzati in funzione dell'angolo di incisione e dello stato sia del latch master (L1, in figura) che dello slave (L2).
L'angolo critico θ_c é l'angolo nel quale il tasso di fault per incidente con delle particelle raggiunge il 10% del massimo tasso raggiunto durante l'analisi. θ_c é stato scelto in questo modo perché, cosí facendo, rappresenta un angolo sufficiente per creare dei soft error e per

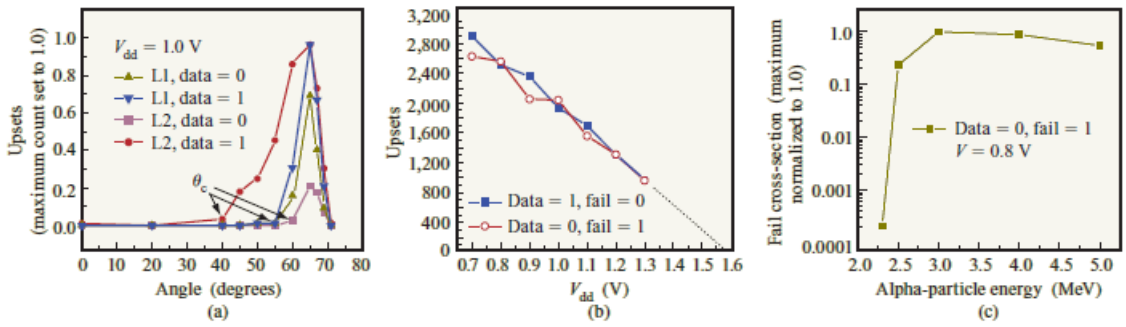

Figura 4.2 – Risultati

ottenere un numero consistente di risultati per il test.
Le figure 4.2b e 4.2c visualizzano i risultati riguardanti un array di memoria SRAM colpito con angolo ortogonale: 4.2b rappresenta il numero di upsets (ossia di variazioni indesiderate del valore in memoria) per ambedue i casi in cui sia salvato uno 0 o un 1 in funzione della tensione mentre 4.2c mostra il tasso di fallimento di 1Mb di SRAM cross-section in funzione dell'energia del raggio di particelle.

4.3 Estrazione della carica depositata

In assenza di livelli BEOL, la carica depositata in un livello di Silicio, dato uno ione incidente con angolo normale, viene calcolata come:

$$Q_{dep} = 44.5 * superficie\ LET\ (in\ MeV/\mu m) * spessore\ Silicio\ (in\ \mu m) \tag{4.1}$$

ove LET sta per particella con energia di trasferimento lineare (*Linear Energy Transfer*). In caso di incisione con un angolo differente, ottentuo dalla semplice inclinazione del chip, la stima della carica depositata viene corretta nel seguente modo:

$$Q_{dep} = 44.5 * superficie\ LET\ (in\ MeV/\mu m) * spessore\ Silicio\ (in\ \mu m)\ /\ cos\theta\ . \tag{4.2}$$

Le equazioni (4.1) e (4.2) assumono che il layer di Silicio sia sottile, come in applicazioni sfruttanti la tecnologia SOI in cui non si superano i 100nm: per dispositivi a tecnologia Bulk o comunque con spessore superiore ai 100nm bisognerá considerare anche il comportamento della particella d'energia mentre subentra nel layer.
Il fatto notevole in questi esperimenti con raggi ionici é che il numero di upset ottenuti sono in funzione sia dell'angolo d'incidenza che della loro energia. Per estrarre piú precisamente la carica depositata, é necessario studiare in dettaglio il trasferimento d'energia nel BEOL, poiché questa struttura porta a significative distorsioni a ribasso dell'energia della particella. Come prima iterazione, ogni livello del BEOL ha una densitá metallica differente rispetto ai livelli adiacenti. Cosí ogni livello avrá un tasso di assorbimento d'energia diverso. La figura 4.3a rappresenta la carica depositata in una simulazione con un raggio ^4He con potenza 5 MeV sparato contro un chip a 10 livelli di metallo (come il POWER6): come si vede

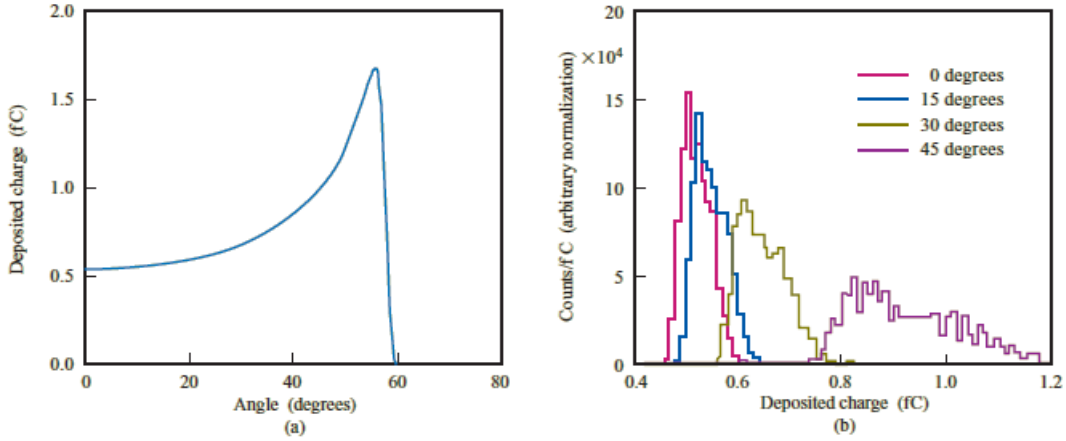

Figura 4.3 – Grafici ottenuti utilizzando il simulatore MCHIDQ prodotto da IBM

nel grafico, la carica depositata varia in base all'angolo di incidenza ma si noti che puó variare anche in base al percorso che la particella intraprende tra i vari livelli prima di arrivare al Silicio. Il secondo grafico 4.3b é stato ottenuto tramite un'altra simulazione ed é raffigurante la distribuzione delle cariche depositate nella superficie del Silicio in base all'angolo di incidenza. Si osservi che i picchi delle distribuzioni riflettono il valore delle cariche depositate del grafico 4.3a indicando quindi una consistenza tra le due simulazioni.

4.4 Schermo contro le particelle alfa da parte del BEOL

Le paticelle alfa sono emesse in direzioni casuali, impedendone una possibile previsione esatta, e praticano una multitudine di percorsi differenti, andando incontro ad una diversa *stopping power* prima di intaccarsi ad un nodo sensibile. La figura 4.4 riassume la probabilitá cumulativa del deposito di una carica prodotta da una particella concepita immediatamente sopra lo stack BEOL, il grafico é in funzione del valore desiderato di carica: la curva *ref* rappresenta la situazione in cui non ci sono ulteriori livelli tra il silicio e l'origine della particella, la curva 2 aggiunge uno strato di 2.8 μm di metallo, la 3, 4 e 5 aggiungono (rispetto alla curva 2) una pellicola protettiva di rispettivamente 1.9, 5.4 e 10.4 μm.
Dato un valore di carica depositata Q, la probabilitá cumulativa di ciascuna curva corrispondente a tal valore rappresenta la probabilitá che si verifichi un evento con deposito di carica maggiore di Q. Si noti che lo strato inserito nella curva 2 viene assunto essere composto da 50% di rame e 50% di ossido di silicio.
Mettiamo in pratica i dati ottenuti nella simulazione con un esempio: se avessimo un dispositivo con carica critica Q_{critic} di 0.8 fC, uno strato addizionale di 2.8 μm puó ridurre il numero di upset generati da particelle alfa di un ordine di magnitudine. Per un dispositivo piú sensibile, ad esempio con Q_{critic} pari a 0.6fC, per ottenere lo stesso aumento di repulsione agli upset ottenuto nel caso precedente bisognerá aggiungere anche una pellicola di circa 5.4 μm.

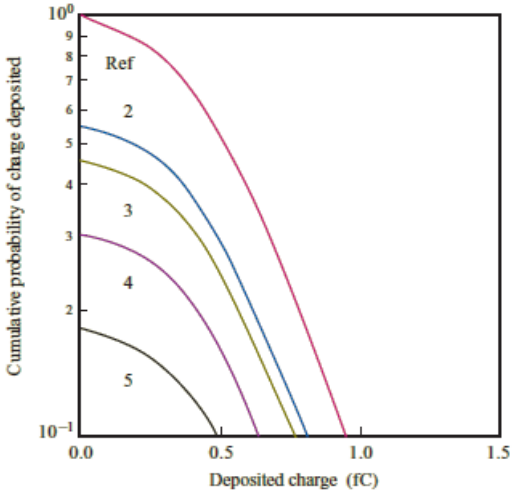

Figura 4.4 – Grafico ottenuto utilizzando il simulatore MCHIDQ prodotto da IBM (l'asse delle ordinate é in scala logaritmica)

4.5 Esperimenti e simulazioni

Nei prossimi pararafi vedremo quattro tipologie di test:

- una simulazione degli effetti dei neutroni prodotti normalmente dai raggi cosmici, ottenuta tramite un'irradiazione diretta di un raggio protonico che colpisce uniformemente tutta l'area del processore POWER6 con protoni con circa 150 MeV d'energia;
- Mambo, un simulatore al computer sviluppato da IBM con cui si possono scegliere dove far cadere le particelle (e, quindi, decidere quali latch colpire) ed, essendo un simulazione virtuale, si ha il vantaggio di poter visualizzare dettagliatamente la degradazione dei segnali nei nodi interessati;
- Un altro simulatore al computer chiamato *statistical fault injection* (SFI), tipicamente utilizzato per la stima dell'AVF (paragrafo 1.4) che introduce dei flip nei latch del circuito in modo totalmente casuale cosí da poter osservare dei fault: tale simulatore puó essere usato su tutto il chip o solo su una zona di particolare interesse. Per quanto riguarda gli errori, si potrá scegliere di confrontare l'*architectural state* del chip con lo stesso stato del chip simulato ma senza introduzione di errori, oppure confrontando gli output dei chip nei due stati appena accennati;
- il quarto ed ultimo esperimento comprenderá un'irraggiamento neutronico diretto al processore effettuato nella Los Alamos National Laboratory;

4.5.1 Protoni

I prossimi risultati qui descritti provengono dal Francis H. Burr Proton Therapy Center of the Massachusetts General Hospital. L'ATE utilizzato é capace di eseguire un qualsiasi

set d'istruzioni nella memoria cache e leggere dinamicamente lo stato del chip. La dose di protoni generata viene imposta dalla macchina in fluenza (*fluence*) ossia il flusso di protoni (quantitá di protoni passanti per unitá d'area per unitá di tempo) integrato nel tempo. La fluenza utilizzata nell'esperimento ha un valore di $8.2 * 10^7$ $protoni/cm^2/MU$, ove MU indica la *monitor unit*, ossia in un MU una dose media di 82 milioni di protoni colpiscono un centimetro quadrato del nostro chip. Con questa configurazione la macchina genera un flusso protonico di sei ordini di grandezza piú grande rispetto ad un flusso *naturale*.
L'esperimento consiste in due test: il primo é un test statico (per approfondire, si vada al paragrafo 2.2), con lo scopo di calcolare la propensitá dei latch a cambiare valore quando sono esposti a questo tipo di raggio; il secondo si appoggia ad un test dinamico (paragrafo 2.2) sfruttato per osservare l'impatto dei flip dei latch durante l'esecuzione di un'applicazione.
In entrambi i test la cache di terzo livello é stata completamente disabilitata ed elettronicamente alienata dal resto del circuito: nessuna risorsa le é stata dedicata e nessun dato o istruzione ha mai interagito con questo livello di cache.

test statico

Per prima cosa, é necessario verificare che il raggio sia distribuito omogeneamente in tutto il chip, una volta verificato questo si puó procedere con l'inizializzazione dell'esperimento. Ricordiamo che il fine di questo test é la determinazione del tasso di flip dei latch per MU. Il primo passo dell'esperimento sará la scrittura di un pattern giá conosciuto in tutti gli *scan-ring* del chip: una volta effettuata quest'operazione, si attiverá il raggio per un numero fissato di MU. A fine scansione si confronteranno i dati salvati sui *scan-ring* con il pattern originale: a causa dei pochi flip avvenuti durante l'esperimento, é stato necessario ripeterlo piú volte prima di ottenere, statisticamente parlando, un numero rilevante di informazioni. La figura 4.5 mostra la distribuzione fisica dei flip ottenuti nei latch durante i test affiancata

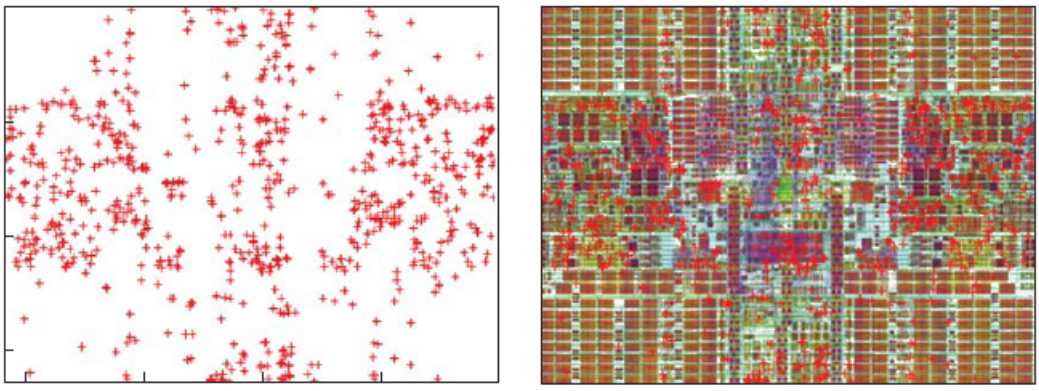

Figura 4.5 – La figura di sinistra illustra il layer dei latch che hanno visto un flip causato dall'irradiazione mentre la figura di destra sovrappone la foto del die del chip con il layer della figura di sinistra

ad una sua sovrapposizione con la foto del die. Come ci si poteva aspettare dalle premesse, gli upset sono avvenuti soprattutto nelle zone con circuiti di logica piuttosto che in regioni dove risiede la cache.

test dinamico

Il test dinamico é stato impiegato utilizzando un *architectural verification program* (AVP), ossia un software stimolante il processore generando istruzioni casuali: questo AVP é stato progettato in modo che controlli le azioni che si succedono all'interno del processore verificando e individuando l'avvenimento degli errori. Durante l'esecuzione dell'AVP, l'ATE monitorizza e verbalizza tutti gli errori individuati: sia gli hardware-detected checkstops che, ovviamente, gli errori individuati dallo stesso AVP (per ulteriori informazioni riguardanti le tipologie di errori, ci si rechi al capitolo 1.4). Ogni tipo di errore individuato viene categorizzato in base alla sua origine: SRAM, file di registro o latch. Nel caso di errore individuato da AVP, tale errore viene distinto dagli altri tramite un appunto sul report finale.
Durante l'esecuzione del test sono stati irraggiati piú di un trilione (mille miliardi) di protoni ad alta energia causando migliaia di flip, errori (poi corretti) e qualche checkstop event. Solo tre (3) errori sono stati determinati esclusivamente dall'AVP (il che indica che solo tre errori sarebbero potuti diventare un SDC).

4.5.2 Mambo

Anche questo test é stato suddiviso in due fasi: la prima con lo scopo di valutare la probabilitá che un errore di *incorrect architected state* (per ulteriori informazioni riguardanti le tipoligie di errori, ci si rechi al capitolo 1.4) sia individuato dall'AVP (lo stesso usato nel test dinamico appena descritto nel paragrafo 4.5.1); la seconda fase, invece, utilizza un benchmark (software) chiamato *bzip2* per osservare quali *fault* immessi nel chip possono essere individuati dall'AVP.

Prima fase

In questa prima parte AVP viene eseguito per un numero fissato d'istruzioni, generalmente parecchi milioni, durante il quale vengono inseriti casualmente degli errori architetturali. L'immissione degli errori avviene bloccando momentaneamente lo stato della macchina e leggendo l'istruzione che é in esecuzione: in base al tipo d'istruzione, un operando o un altro valore viene corrotto con un singolo bit flip. Una volta fatta ripartire la simulazione, si attende l'esecuzione dell'istruzione per poi analizzare l'impatto del flip e categorizzarlo in base al risultato.
L'esito della simulazione viene categorizzata in tre categorie:

1. Nessun impatto;
2. Comportamento inatteso individuato dal programma monitorante o da AVP;
3. AVP ha individuato l'errore inserito.

Seconda fase

In questa parte il test consiste nel dare in pasto a *bzip2* un'immagine JPEG da comprimere e, durante la compressione, immettere un errore similarmente a come fatto nella prima fase: nel caso in cui il programma non venga influenzato dall'errore immesso, l'immagine compressa combacia con l'immagine campione compressa in un altro momento (ovviamente senza errori). Gli errori sono categorizzati in armonia alla prima fase (si faccia riferimento alla lista qui sopra): il primo caso rappresenta una normale esecuzione della compressione con nessuna differenza sull'immagine finale; il secondo caso capita quando *bzip2* o il kernel della macchina individuano qualche tipo di errore; il terzo e ultimo caso accade quando il programma conclude l'operazione di compressione senza osservare errori ma l'output differisce dell'immagine campione (caso SDC).

I risultati di ambedue i test sono presentati nella figura 4.6.

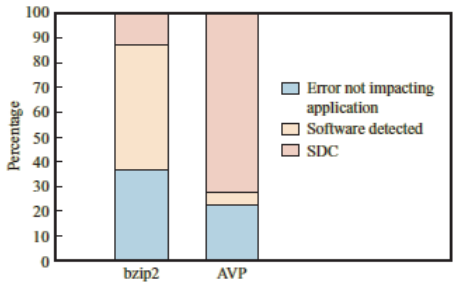

Figura 4.6 – Risultati dell'esperimento Mambo. (SDC: *silent data corruption*; AVP: *architectural verification program*).

4.5.3 SFI

Per questo test si é deciso di optare per una simulazione hardware-accelerata (*hardware-accelerated simulation*) dell'intero chip: ad ogni nuova generazione si introducono decine di milioni di transistor in piú nei chip rispetto alla generazione precedente, risultando in un drammatico incremento dei tempi richiesti per una simulazione software: le simulazioni assistite da un hardware speciale sono il miglior sistema per velocizzare le simulazioni di chip dai grandi design (come il POWER6). La velocitá di questo nuovo approccio puó arrivare ad essere ordini di magnitudine piú grande rispetto alla mera simulazione software. Un ulteriore vantaggio della simulazione hardware accelerata é l'esecuzione del codice in parallelo anziché in serie come, invece, avviene nelle simulazioni software.
Un modello *register-transfer level* (RTL), ossia un modello descrivente il comportamento del circuito in termini di flussi di segnali (o trasferimenti di dati) tra registri hardware e le operazioni logiche eseguite su questi segnali, del POWER6 viene sintetizzato e caricato nell'acceleratore hardware: cosí facendo 'acceleratore si comporterá esattamente come il POWER6 con la differenza che opererá ad una frequenza molto piú bassa. Nel nostro test viene utilizzato Awan, un acceleratore hardware di proprietá IBM.

In questi esperimenti i fault vengono introdotti in locazioni selezionate e i loro effetti sono valutati controllando lo stato del sistema e i registri di stato del processore. Una volta inserito un flip in un determinato ciclo di clock, si lascia operare il processore per un numero fissato di cicli (500,000 in questo test) cosí da assicurare che tutti i possibili effetti del fault introdotto siano identificati. Si osservi che gli errori individuati da AVP non sono visibili dalla macchina e vengono schedati in registri speciali.

I risultati di questo test verranno esposti a fine capitolo assieme ai risultati del prossimo test introdotto nel paragrafo seguente;

4.5.4 Neutroni

Mentre per raccogliere dei dati rilevanti nell'esperimento con il raggio protonico ci sono volute all'incirca 20 ore, per lo studio con il raggio neutronico sono stati necessari svariati giorni prima di poter comparare i dati ottenuti. Questo é il grande vantaggio di avere accessibilitá ad un ATE con raggio protronico: i risultati sono disponibili oltre il centinaio di volte piú velocemente. La velocitá non é nemmeno l'unico vantaggio: il maggior tempo richiesto dal raggio neutronico implica dover mantenere sotto controllo sia il raggio sia monitorare piú sistemi contemporaneamente per molti giorni.

I risultati del test con raggio neutronico presentati nel prossimo paragrafo sono stati ottenuti in un intervallo di tempo di circa una settimana.

4.5.5 Risultati

Il software AVP é stato eseguito negli esperimenti d'irradiazione neutronica e protonica e nel SFI. Ogni test ha prodotto un numero differente di flip a causa della diversa natura e per la differente logistica utilizzata. Il numero di flip e la tipologia di fault che essi

	Proton	*Neutron*	*SFI*
Flips	1,748	541	16,817
a) Vanished (%)	95.65	97.35	94.98
b) Corrected (%)	3.49	2.03	3.70
c) Checkstops (%)	0.63	0.37	0.90
d) Incorrect architected state (%)	0.23	0.24	0.42
1) No impact (%)	n/a	n/a	0.08
2) Software detected (%)	n/a	n/a	0.03

Figura 4.7 – Confronto tra i risultati ottenuti negli esperimenti d'irradiazione e dei immissione di fault. (SFI:*statistical fault injection*).

hanno determinato sono raffigurati nella tabella 4.7: nonostante la differenza nel numero

di flip, la percentuale delle varie categorie dei fault osservati corrisponde molto bene fra i vari esperimenti. Come accennato nel paragrafo precedente, nonostante l'intervallo di tempo impiegato nell'esperimento con raggio neutronico per ottenere i risultati fosse piú del doppio rispetto alla controparte protonica, i flip su latch ottenuti sono meno di un terzo rispetto a quest'ultimo. Il test SFI ha bersagliato solamente i latch logici senza considerare le SRAM facendo in modo che molti piú cicli e molti piú flip fossero simulati rispetto gli esperimenti con le irradiazioni.
Le minime differenze tra i risultati osservati tramite SFI e gli esperimenti irradianti possono essere giustificate dalle imprecisioni nel timing del ciclo della simulazione Awan (visto che questa tipologia di simulazione non riflette accuratamente le tempistiche e gli effetti dei flip nella logica combinatoria).

Capitolo 5

Non solo microchip: I/O hub del sistema POWER6

Nella valutazione del SER di un sistema é importante considerare tutti i suoi componenti e non limitarsi al solo microprocessore. É noto ormai da tempo che i segnali connettenti i circuiti integrati possono incontrare degli errori ed é prassi comune implementarci degli schemi di protezione come ECC. Nonostante quest'implementazione sia indubbiamente valida, la complessitá dei dispositivi I/O aumenta di generazione in generazione a causa dell'introduzione di nuovi sottosistemi o all'inserimento di altri circuiti, questo comporta ad avere un circuito pericolosamente sensibile agli errori transitori.
In questo capitolo studieremo gli effetti dei soft error in un I/O hub a 90nm sviluppato in tecnologia bulk ad hoc per il processore POWER6. L'architettura di un I/O hub é molto diversa da quella di un microprocessore. I segnali al suo interno sono resistenti da errori causati dal rumore generato dai bus grazie all'implementazione di ECC. Inoltre, il sistema IBM ha incorporato anche una ridondanza *I/O bridge* che assicura la continuazione dell'esecuzione da parte del hub anche nel caso si verificasse un errore.
I sistemi I/O sono costruiti in modo di massimizzare la connettivitá I/O e in modo da raggiungere la piú alta banda possibile. Nel seguito vedremo dei test in cui la banda utilizzata é, in base alla situazione, massima, minima o in un stato d'utilizzo medio.
Il sistema POWER6 usato in questo studio fa uso di una coppia di microprocessori dualcore, per un totale di quatto core, ed un unico I/O hub: le varie misurazioni sono compiute utilizzando varie combinazioni delle porte PCI-X, PCI Express (PCIe) ed un'interfaccia *high-speed host Ethernet adapter* (HEA).

5.1 Misurazioni tramite irradiazioni

Durante il test viene utilizzato un raggio protonico messo a disposizione dal Francis H. Burr Proton Therapy Center of the Massachusetts General Hospital: utilizzando le stesse procedure viste nel paragrafo 4.5.1, sono stati effettuati alcuni test statici per la determinazione di quali latch dell'hub fossero maggiormente suscettibili ai flip indotti dall'irradiazione di cui ora abbiamo qui i risultati rappresentati in figura 5.2 : come si puó osservare, non c'é

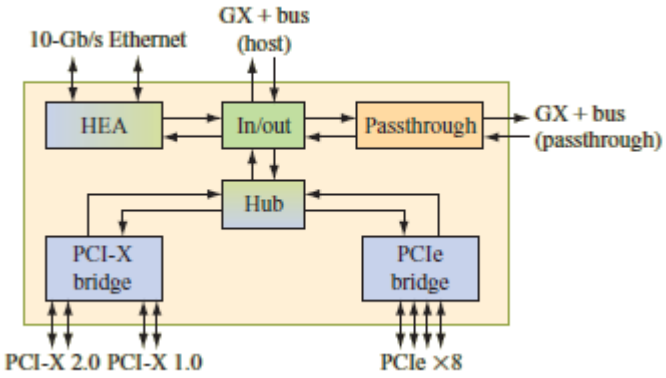

Figura 5.1 – Schema del hub chip illustrante i componenti principali

una zona di circuiteria logica piú suscettibile di un'altra. L'hub puó essere considerato omogeneamente sensibile a questa tipologia di irradiazione.

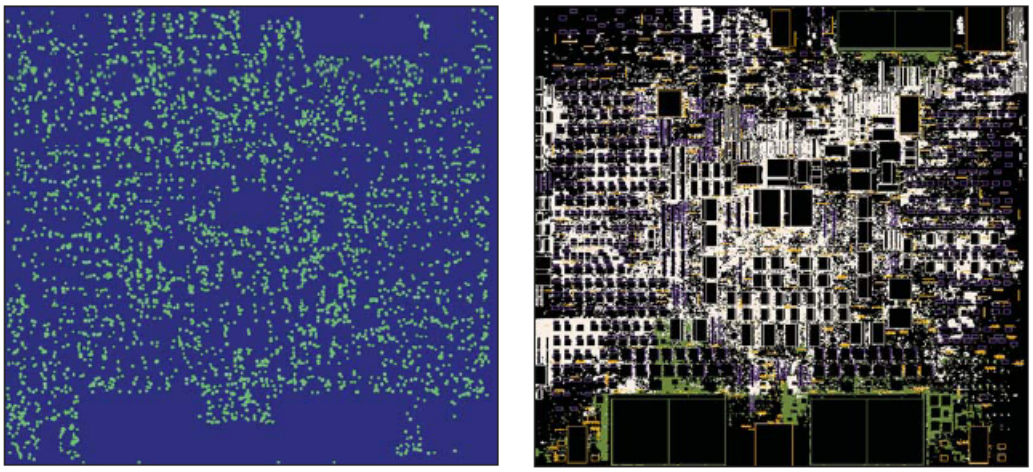

Figura 5.2 – Mappatura dei flip ottenuti durante l'irradiazione e il corrispondente layout dell'hub.

5.2 Individuazione e correzione degli errori

Il sistema POWER6 ha una politica di gestione degli errori includente anche i segnali dell'I/O hub. Gli errori potranno essere individuati sia all'interno che all'esterno del hub: anche per quanto riguarda la correzione, essa potrá essere effettuata sia all'interno che all'esterno del chip come, ad esempio, nella CPU. Nella memoria o in vari bus che connettono l'hub ai vari componenti. Il punto in cui l'errore viene individuato dipende dalla sua origine: un errore nell'interfaccia dell'hub potrá essere individuato da altre parti

del sistema, generalmente il bus GX é tra i primi componenti ad individuare un errore, se non viene individuato qui potrá essere la CPU ad individuarlo oppure potranno essere altri sottosistemi che si occupano della memoria. Comunque sia, la maggior parte degli errori viene catturato all'interno del perimetro dell'hub.
ECC é itegrato ovunque nel sistema POWER6, hub compreso: grazie a ECC vengono determinati tutti gli *double-bit error* e si correggono tutti i *single-bit error*. Sebbene un errore venga individuato il prima possibile, la sua correzione potrebbe essere affidata al componente che ne fa uso, questo fa in modo che ci sia il massimo isolamento degli errori e la massima tolleranza.

5.3 test

Nei prossimi test ci si é focalizzati negli adattatori I/O del hub per studiarne il comportamento. Sono state usate delle copie di esercitatori per gli adattatori Ethernet: ogni adattatore sfrutta un esercitatore collegato a un PCI slot o all'interfaccia HEA. Ogni copia invia e riceve dati simultaneamente tra le due porte Ethernet.
La probabilitá che un soft error influenzi la funzionalitá dell'hub e risulti in uno checkstop (paragrafo 1.4) é tanto piú alta quanto piú l'hub é utilizzato, raggiungendo il picco al 100% d'utilizzo. Si osservi che, fino a quando l'hub non é in uso, esso risulta immune ai flip: ossia il fatto che qualche suo latch abbia cambiato incidentalmente valore non porta a nessuna conseguenza. Questo, peró, non é lo scenario che si ha quando c'é un ampio utilizzo dell'hub: per determinare quanto un dispositivo I/O venga utilizzato basta guardare il flusso di dati entranti ed uscenti da esso. Per studiare il SER del dispositivo bisognerá necessariamente tenere conto di questo flusso.
In questi test sono state utilizzate otto configurazioni diverse usando sempre lo stesso *Ethernet test exerciser* in modo tale da ottenere un flusso di dati sempre casuale ma affidabile per un resoconto finale confrontante i risultati.
Nota: i PCI test utilizzano un adattatore Ethernet da 1Gb/s.

Le configurazioni utilizzate sono:

1. idle;
2. Full BW PCI test;
3. Half BW PCI test;
4. PCI con singolo adattatore test;
5. 10-Gb/s HEA test;
6. 1-Gb/s HEA test;
7. 10-Gb/s HEA + PCI test;
8. 1-Gb/s HEA + PCI test;

I test 2, 7 e 8 sono stati concepiti per massimizzare la banda. Da notare che i termini 1 Gb/s e 10 Gb/s rappresentano il massimo flusso di banda supportato dall'interfaccia Ethernet e non necessariamente la banda utilizzata durante il test. Il test 1, ossia quello in cui si lascia l'hub senza lavoro (nessun flusso in input e/o in output), é un caso particolarmente interessante in quanto un flip introdotto ha il potere di causare un'individuazione di una corruzione in un dato inesistente oppure puó far credere all'hub che ci sia del lavoro da eseguire sebbene, in realtá, non ci sia nulla da fare: in conclusione, quindi, anche nel caso di un utilizzo dell'hub pari a 0% si ha un SER comunque non nullo.
Ogni configurazione definita sopra é indistinguibile per quanto riguarda il numero di porte Ethernet testate: ad esempio, nel test 2 sono state utilizzate otto porte Ethernet, nel test 3 solamente quattro. Rieseguire i test con numeri di porte differenti non porta a nessun cambiamento dei risultati qui riportati grazie all'applicazione di una normalizzazione.

5.4 test dell'adattatore PCI

I test che generano traffico presso le interfacce PCI e GX muovono dati tra la memoria e le porte Ethernet: ad esempio, un esercitatore potrá inoltrare un messaggio tramite il protocollo internet TCP/IP che risulterá in un comando alla porta Ethernet di prelevare tale messaggio dalla memoria e inviarlo al proprio cavo Ethernet, nell'altra fine del cavo ci sará un'altra interfaccia Ethernet (sempre dello stesso sistema, in questi test) che scriverá tale messaggio nella memoria e poi notificherá la CPU dell'avvenuto ricevimento del messaggio, facendo attivare nuovamente l'esercitatore (lo stesso che ha inviato il messaggio) che controllerá l'integritá o la correzione delle informazioni ricevute. L'utilizzo di piú porte Ethernet in parallelo aumenta il throughput totale dell'hub diminuendo il tempo necessario per i test.
Per il test 2, chiamato *Full BandWith PCI test*, sono stati configurati quattro adattatori Ethernet dual-port 1 Gb/s, ma per arrivare ad occupare tutta la banda disponibile sono state aggiunte altre due interfacce PCI-X e due adattatori PCIe arrivando ad un totale di otto interfacce: ogni porta Ethernet da 1 Gb/s puó generare un flusso teorico massimo per direzione di 120 MB/s e, visto che in ogni adattatore Ethernet c'é una dual-port, si arriva ad un picco massimo di 240Mb/s per direzione e per adattatore, ottenendo in totale un valore massimo di 1920 Mb/s (quattro dual port con otto interfacce a 1 Gb/s).
Similarmente al test 2, anche nel test 3 (detto *Half BandWith PCI test*) viene utilizzata la stessa configurazione in cui, peró, viene utilizzata soltanto una porta per ognuna delle quattro interfacce, portando cosí ad un picco teorico massimo di 960 Mb/s (esattamente metá del test 2). Nel test 4, *PCI con singolo adattatore test*, si é applicato la stessa procedura usata nel test 2 ma utilizzando un solo adattatore anziché quattro.

5.5 test HEA

L'*host Ethernet adapter* (HEA) é una funzione Ethernet integrata nel chip dell'hub, essa puó essere configurata come una porta a 1Gb/s o 10Gb/s. I test 5 e 6, rispettivamente *10-Gb/s HEA test* e *1-Gb/s HEA test*, utilizzano due porte Ethernet a 10 Gb/s oppure quattro a 1 Gb/s. I test 7 e 8, rispettivamente *10-Gb/s HEA + PCI test* e *1-Gb/s HEA*

+ PCI test, hanno utilizzato tutte le configurazioni presentate assieme (comprese quelle nel paragrafo precedente): in questi ultimi due casi le interfacce PCI sono state sfruttate a velocitá media per evitare colli di bottiglia causati dalla CPU.

5.6 Risultati

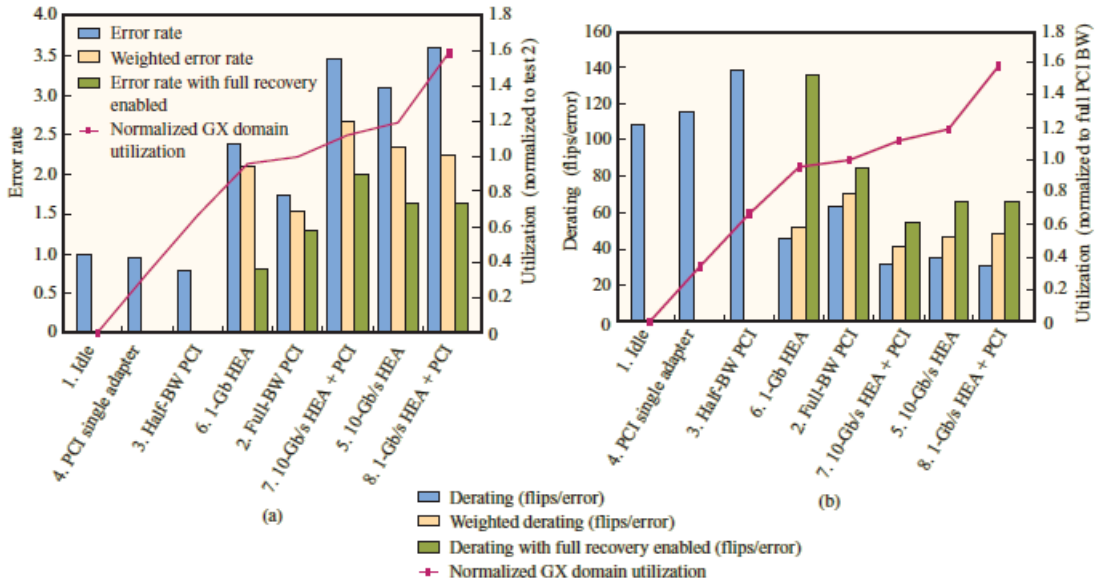

Figura 5.3 – Risultati dei test: (a) tasso d'errore normalizzato al idels test; (b) Riduzione degli errori per ogni test

La figura 5.3a visualizza il tasso degli errori (barre blu) ottenuto per ogni configurazione prendendo come punto di riferimento il test in idle. Le barre gialle mostrano le stesse misurazioni dopo aver raggiunto il 100% della banda dichiarata mentre le barre verdi indicano la correzione del tasso d'errori ottenuti per il fatto che alcuni checkstop ottenuti nel test non accadranno mai in un vero ambiente applicativo del circuito: difatti, durante i vari test, si é utilizzato un firmware a basso livello con molte funzioni di eliminazione e di correzione di errori volontariamente disabilitate.
Infine, le linee porpora esprimono l'utilizzo dei circuiti nelle interfacce GX per ogni test: si noti che, generalmente, piălto é l'uso della GX piú é alto il tasso degli errori.
Il fatto che i test *Half BW PCI test* e *PCI con singolo adattatore test* abbiano risultati statisticamente comparabili con il test in idle ha sorpreso anche i ricercatori IBM che svolgevano le sperimentazioni: é stato concluso che queste due tipologie di test (Half BW PCI e PCI con singolo adattatore) non sollecitano abbastanza il chip per sovrastare la suscettibilitá di errore causata dall'idle noise, ossia errori ottenuti mentre il dispositivo non é in uso. Una volta sovrastato questo idle noise, il numero di fallimenti aumenta linearmente con l'aumento dell'utilizzo del chip. Ora che si possiedono i risultati di test, é

possibile ottenere una stima del tasso di errori massimo in base alla banda. Ad esempio, se avessimo un test con banda pari al 25% del picco massimo teorico, il tasso di fallimenti puó essere calcolato come:

$$f_{st} = \left[\frac{(f_t - f_i)}{0.25}\right] + f_i,$$

ove f_{st} é il tasso d'errore scalato al 100% della banda, f_t é il tasso d'errore misurato nel test e f_i é il tasso d'errore misurato mentre il chip é in idle. Una volta ottenuto il tasso massimo si potrá ottenere la stima del tasso desiderato (come quello di un normale utilizzo), come in figura 5.3 dove é assunto un valore del 20%.
Dai tassi d'errore individuati, possiamo calcolare il tasso d'energia richiesta al raggio protonico per produrre un errore causato da flip di uno o piú latch. Questo ci permette inoltre di individuare il numero di latch flip richiesti per produrre un soft error: la figura 5.3b illustra i risultati appena descritti.
L'aumento di utilizzo e, conseguentemente, l'aumento del SER viene osservato in due modi: incrementando i circuiti attivi nell'hub durante il test (difatti i test HEA + PCI sono quelli con piú tasso d'errore) e aumentando la banda utilizzata: difatti il test 10-Gb/s HEA ha un tasso d'errori maggiore rispetto alla controparte ad 1-Gb/s o al test che riempe completamente la banda dell'interfaccia PCI (test 3).

Questo studio ha dimostrato un'ottima resistenza agli errori dell'hub anche in caso di pieno carico: persino confrontando il tasso d'errori al massimo utilizzo con lo stesso in stato di idle il risultato é piuttosto buono. Come ci si aspettava, inoltre, la resistenza agli errori é ancora maggiore una volta implementati e attivati i sistemi di individuazione e correzione degli errori. Si ricordi, inoltre, che i test qui affrontati si sono basati esclusivamente sulle parti piú suscettibili da errori del chip (sezioni verdi nella figura 5.1): le corruzioni in tutti gli altri componenti del hub (in blu nella figura 5.1) sono completamente recuperabili tramite o hardware o il firmware del dispositivo, il che comporta ad ulteriori miglioramenti in tutto il sistema.

Conclusione

Nel corso di questa tesi é stato studiato il comportamento dei chip moderni nel momento in cui essi vengano colpiti da una o piú particelle sia in un mondo virtualizzato, sia sfruttando dei laboratori con capacitá di monitorizzazione dei circuiti testati, inoltre, alcuni di questi laboratori possedevano la capacitá di riprodurre e generare le particelle di nostro interesse con frequenza molto piú alta rispetto a quella trovata in natura, in modo da velocizzare il processo d'analisi mantenendo comunque un elevato grado di affidabilitá dei risultati. Altresí, sono stati presentati effetti e possibili soluzioni implementabili nei circuiti moderni spiegando pro e contro di ciascuno indicandone in quali ambiti si preferisce usare un metodo piuttosto di un altro. Vorrei puntualizzare che proprio in questo ambito ho percepito che il livello di ricerca possa ancora andare avanti e avrei desiderato vedere delle implementazioni per la correzione degli errori innanzitutto esclusivamente elettroniche e, poi, piú affidabili e con meno difetti di quelle attuali. C'é da dire che ció che é stato descritto qui é soltanto la punta dell'iceberg di questa materia e che nel mondo industriale ci sono molte piú soluzioni delle quali, peró, si sa poco o nulla per ovvie ragioni di mercato.
Tutto questo é stato messo a disposizione del lettore prima di entrare nella parte finale dell'analisi dove si é stati introdotti al lato pratico dell'argomento in cui si sono studiati tutti i concetti visti precedentemente in vari test condotti da alcuni ricercatori di IBM per un prodotto reale che, al momento delle sperimentazioni, non era ancora stato lanciato nel mercato. Qui é stato visto come i circuiti moderni, se utilizzanti le giuste prevenzioni, siano davvero resistenti alle varie fonti di corruzione dei dati, e quale sia il loro comportamento in vari ambiti e situazioni. Vorrei ricordare, infine, che il sistema POWER6 é stato progettato per essere la nuova *flagship* di IBM nell'ambito server e che quindi difficilmente vedrá utilizzi in ambienti ostili come alcuni di quelli utilizzati negli esperimenti avvalorando ancora di piú la resistenza di una macchina con quest'architettura.

Bibliografia

[1] D. F. Heidel, K. P. Rodbell, E. H. Cannon, C. Cabral Jr., M. S. Gordon, P. Oldiges, and H. H. K. Tang, Alpha-particle-induced upsets in advanced CMOS circuits and technology. *IBM Journal of Research and Development* 52(3) pp. 225 - 232, May 2008

[2] H. H. K. Tang, SEMM-2: A new generation of single-event-effect modeling tools. *IBM Journal of Research and Development* 52(3) pp. 233 - 244, May 2008

[3] H. H. K. Tang, C. E. Murray, G. Fiorenza, K. P. Rodbell, M. S. Gordon, and D. F. Heidel, New simulation methodology for effects of radiation in semiconductor chip structures. *IBM Journal of Research and Development* 52(3) pp. 245 - 253, May 2008

[4] A. KleinOsowski, E. H. Cannon, P. Oldiges, and L . Wissel, Circuit design and modeling for soft error. *IBM Journal of Research and Development* 52(3) pp. 255 - 263, May 2008

[5] M. S. Gordon, K. P. Rodbell, D. F. Heidel, C. Cabral Jr., E. H. Cannon, and D. D. Reinhardt, Single-event-upset and alpha-particle emission rate measurement techniques. *IBM Journal of Research and Development* 52(3) pp. 265 - 273, May 2008

[6] P. N. Sanda, J. W. Kellington, P. Kudva, R. Kalla, R. B. McBeth, J. Ackaret, R. Lockwood, J. Schumann, and C. R. Jones, Soft-error resilience of the IBM POWER6 processor. *IBM Journal of Research and Development* 52(3) pp. 275 - 284, May 2008

[7] C. Bender, P. N. Sanda, P. Kudva, R. Mata, V. Pokala, R. Haraden, and M. Schallhorn, Soft-error resilience of the IBM POWER6 processor input/output subsystem. *IBM Journal of Research and Development* 52(3) pp. 285 - 292, May 2008

[8] J. A. Rivers, P. Bose, P. Kudva, J.-D. Wellman, P. N. Sanda, E. H. Cannon, and L. C. Alves, Phaser: Phased methodology for modeling the system-level effects of soft error. *IBM Journal of Research and Development* 52(3) pp. 293 - 306, May 2008

[9] T. J. Dell, System RAS implications of DRAM soft error. *IBM Journal of Research and Development* 52(3) pp. 307 - 314, May 2008

[10] David G. Mavis and Paul H. Eaton, Temporally redundant latch for preventing single event disruptions in sequential integrated circuits. *Technical Report*, Sept 1998

[11] David G. Mavis and Paul H. Eaton, Soft error rate mitigation techniques for modern microcircuits. *Reliability Physics Symposium Proceedings* pp. 216 - 225, Aug 2002. DOI: 10.1109/RELPHY.2002.996639

[12] VA: JEDEC Solid State Technology Association, Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft error in Semiconductor Devices. *JESD89A*, October 2006

[13] Shubu Mukherjee, Architecture design for soft error. *Morgan Kaufmann Publishers*, February 2008

[14] Jan M. Rabaey, Anantha Chandrakasan, and Borivoje Nikolic, Digital Integrated Circuits. *Prentice-Hall*, January 2003

[15] A. Eto, M. Hidaka, Y. Okuyaa, K. Kiimura, and M. Hosono, Impact of Neutron Flux on Soft error in MOS Memories. *IEEE Electron Devices Meeting, IEDM Technical Digest* p367 - 370, Dec 1998

[16] Remigiusz Wisniewski, Arkadiusz Bukowiec, and Marek Wegrzyn, Benefits of hardware accelerated simulation. *The International Workshop on Discrete-Event System Design*, June 2001

[17] J. F. Ziegler, H. W. Curtis, H. P. Muhlfeld, C. J. Montrose, B. Chin, M. Nicewicz, C. A. Russell, W. Y. Wang, L. B. Freeman, P. Hosier, L. E. LaFave, J. L. Walsh, J. M. Orro, G. J. Unger, J. M. Ross, T. J. O'Gorman, B. Messina, T. D. Sullivan, A. J. Sykes, H. Yourke, T. A. Enger, V. Tolat, T. S. Scott, A. H. Taber, R. J. Sussman, W. A. Klein, and C. W. Wahaus, IBM experiments in soft fails in computer electronics (1978-1994). *IBM Journal of Research and Development* 40(1) pp. 3 - 18, Jan 1996

[18] Yuhong Li, Suge Yue, Yuanfu Zhao, and Guozhen Liang, Low Power Dissipation SEU-hardened CMOS Latch. *Piers online* 3(7) pp. 1080 - 1084, 2007. DOI: 10.2529/PIERS060906031953

[19] Riaz Naseer and Jeff Draper, The DF-DICE Storage Element for Immunity to Soft error. *IEEE International Symposium on Circuits and Systems*, Aug 2005

[20] Riaz Naseer and Jeff Draper, DF-DICE: A Scalable Solution for Soft Error Tolerant Circuit Design. *IEEE International Symposium on Circuits and Systems*, May 2006

[21] Corey Toomey, Statistical fault injection and analysis at the register transfer level using the verilog procedural interface. May 2011

nohyphenation