



MINISTERO DELLO SVILUPPO ECONOMICO
DIREZIONE GENERALE PER LA LOTTA ALLA CONTRAFFAZIONE
UFFICIO ITALIANO BREVETTI E MARCHI

DOMANDA NUMERO	102001900967123
Data Deposito	31/10/2001
Data Pubblicazione	01/05/2003

Priorità	09/702.631
Nazione Priorità	US
Data Deposito Priorità	

Sezione	Classe	Sottoclasse	Gruppo	Sottogruppo
G	06	F		

Titolo

APPARECCHIATURA DI COLLAUDO DI MEMORIE CON POST-DECODIFICA INCREMENTATA.
--

RM 2001 A 000644

SIB BI3132R

10004438-5

DESCRIZIONE dell'invenzione industriale dal titolo:
"APPARECCHIATURA DI COLLAUDO DI MEMORIE CON POST-
DECODIFICA INCREMENTATA"

a nome di Agilent Technologies, Inc.

(una Società dello Stato del Delaware)

di Palo Alto, California (Stati Uniti d'America)

.....
DESCRIZIONE

Riferimenti a domande di brevetto correlate

L'argomento della presente domanda di brevetto è correlato con quanto illustrato in una domanda di brevetto statunitense intitolata MEMORY TESTER HAS MEMORY SETS CONFIGURABLE FOR USE AS ERROR CATCH RAM, TAG RAM's, BUFFER MEMORIES AND STIMULUS LOG RAM, n° di serie (sconosciuto) e depositata il 28 settembre 2000. Tale descrizione illustra aspetti di operazioni chiamate Address Classification e Data Classification che sono di interesse in questa sede. Per tale ragione la domanda di brevetto statunitense n° di serie (sconosciuto) è espressamente qui incorporata a titolo di riferimento.

La domanda di brevetto incorporata sopra

incorpora essa stessa la domanda di brevetto statunitense n° di serie 09/665.892 ed è intitolata ERROR CATCH RAM POR MEMORY TESTER HAS MEMORY SETS CONFIGURABLE FOR SIZE AND SPEED, depositata il 20 settembre 2000. Quest'ultima domanda ha una descrizione esauriente del sistema di memoria utilizzata nella forma di realizzazione preferita descritta oltre, che sebbene in principio non è essenziale, è tuttavia il sistema di scelta presente. I suoi attributi sarebbero difficili da sopprimere da una descrizione come quella che segue; conseguentemente, la domanda n° di serie 09/665.892 è anche qui espressamente incorporata a titolo di riferimento.

Base tecnica dell'invenzione

I dispositivi e le capacità elettroniche sono cresciute in modo eccezionale nella vita di ogni giorno.

Assieme ai personal computer a domicilio, molte persone porta più di un utensile di produttività per vari e disparati scopi. La maggior parte dei dispositivi elettronici di produttività personale includono una qualche forma di memoria non volatile i telefoni cellulari impiegano memoria non volatile al fine di immagazzinare e trattenere

numeri telefonici programmati dall'utilizzatore e configurazioni relative quando l'energia viene disinserita. Schede PCMCIA impiegano memoria non volatile per immagazzinare e mantenere informazione anche quando la scheda è rimossa dal suo slot nel calcolatore. Numerosi altri dispositivi elettronici comuni traggono vantaggio dalla capacità di memorizzazione a lungo termine di memoria non volatile in complessi non alimentati.

I fabbricanti di memorie non volatili che vendono ai fabbricanti di apparecchiature elettroniche richiedono apparecchiature di collaudo per sottoporre ad esercizio e verificare il corretto funzionamento delle memorie che producono. Per effetto del volume delle memorie non volatili che sono fabbricate e vendute a prezzi sempre più bassi, è estremamente importante minimizzare il tempo necessario per collaudare una singola parte. Gli acquirenti di memoria non volatili richiedono che i fabbricanti di memorie forniscano alte rese di spedizione per effetto dei risparmi di costi associati con la pratica di incorporare i dispositivi di memoria in complessi più costosi con collaudo minimo o assente. Conseguentemente, il processo di collaudo di memoria deve essere

sufficientemente efficiente per identificare una grande percentuale di parti non conformi e preferibilmente tutte le parti non conformi in un singolo processo di collaudo.

Con il fatto che le memorie non volatili diventano più grandi, più dense e più complesse, le apparecchiature di collaudo devono essere in grado di gestire la grandezza aumentata e la complessità aumentata senza aumentare significativamente il tempo necessario per collaudarle. Le apparecchiature di collaudo di memoria spesso marciano in continuo, ed il tempo di collaudo è considerato un fattore decisivo nel costo della parte finale. Con l'evolvere e migliorare delle memorie, la apparecchiatura di collaudo deve essere in grado di ricevere facilmente le variazioni fatte nel dispositivo. Un altro elemento specifico al collaudo di memorie non volatili consiste nel fatto che ripetute scritture delle celle delle memorie possono degradare le prestazioni di vita utile complessiva della parte. I fabbricanti di memorie non volatili hanno risposto a numerosi degli argomenti di collaudo costruendo speciali modi di collaudo nei dispositivi di memoria. Questi modi di collaudo non sono affatto utilizzati

dall'acquirente della memoria, ma a questi può accedere il fabbricante per il collaudo di tutta o di parte significative delle memorie non più breve tempo possibile e nel modo più efficiente possibile. Alcune memorie non volatili sono anche in grado di essere riparate durante il processo di collaudo. La apparecchiatura di collaudo, conseguentemente, deve essere in grado di identificare: una necessità di una riparazione; una collocazione della riparazione; il tipo di riparazione necessario; e, deve essere quindi in grado di effettuare la riparazione appropriata. Un tale processo di riparazione richiede una apparecchiatura di collaudo che è in grado di rivelare ed isolare una specifica parte non conforme della memoria. Al fine di trarre vantaggio completo dei modi speciali di collaudo così come delle funzioni di riparazione, è vantaggioso che una apparecchiatura di collaudo sia in grado di eseguire un programma di collaudo che supporta diramazioni condizionali basate su risposte previste del dispositivo.

Da una prospettiva concettuale, il processo di collaudo di memorie è un processo algoritmico. Ad esempio, collaudi tipici includono 'incrementare o

decrementare sequenzialmente gli indirizzi di memoria mentre si scrivono 0 e 1 nelle celle di memoria. E' usuale fare riferimento ad una raccolta di 1 e 0 che vengono scritti o letti durante un ciclo di memoria come un "vettore", mentre il termine "configurazione" si riferisce ad una sequenza di vettori. E' cosa convenzionale per i collaudi includere configurazioni di scrittura nello spazio di memoria quali scacchiere, 1 mobili e configurazioni a farfalla. Uno sviluppatore di collaudo può più facilmente ed efficientemente generare un programma per creare questa configurazione con l'aiuto di costruzioni algoritmiche. Una configurazione di collaudo che è algebricamente coerente è anche più facile da "debug" e impiegare metodi logici per isolare parti della configurazione che non si comportano nel modo previsto. Una configurazione di collaudo che è generata algebricamente impiegando istruzioni e comandi che sono ripetuti in loop di programma consumano meno spazio nella memoria della apparecchiatura di collaudo. Conseguentemente, è desiderabile avere una capacità di generazione di configurazioni di collaudo algoritmiche in una apparecchiatura di collaudo di memorie.

La collocazione e rivelazione precisa di fronti di segnale è anche una considerazione nella efficacia di una apparecchiatura di collaudo di memorie non volatili. Al fine di identificare parti che sono generalmente conformi ad una media mentre non conformi entro i margini specificati, una apparecchiatura di collaudo di memorie non volatili deve essere in grado di collocare con precisione ciascun fronte di segnale rispetto al tempo nei confronti di un altro fronte di segnale. E' anche importante essere in grado di misurare con precisione a quale punto nel tempo viene ricevuto un fronte di segnale. Conseguentemente, una apparecchiatura di collaudo di memorie non volatili dovrà avere sufficiente flessibilità e controllo della temporizzazione e collocazione di stimoli e risposte dal Device Under Test (memoria).

Le apparecchiature di collaudo di memoria sono dette generare vettori di trasmissione che sono applicati (stimolo) al DUT (Device Under Test), e ricevere vettori che sono previsti di ritorno (risposta). La logica algoritmica che genera questi vettori può generalmente effettuare ciò senza preoccuparsi di come un particolare bit in un vettore deve pervenire verso o da una particolare

piazzola di segnale nel DUT, mentre la apparecchiatura di collaudo di memoria contiene disposizioni di mappatura per instradare segnali da e verso i piedini che entrano in contatto con il DUT. La raccolta di generazioni di configurazioni algoritmiche, impostazioni di soglia, meccanismi di confronto e condizionamento di segnale, e sonde che collegano tutto ciò al DUT, viene chiamato sito di collaudo. Nel caso semplice in questa sede vi è un solo DUT per sito di collaudo.

Le apparecchiature di collaudo di memoria hanno una memoria interna di collaudo che viene impiegata per facilitare il processo di collaudo. Questa memoria interna di collaudo può essere impiegata per vari scopi, tra i quali vi sono la memorizzazione di vettori di trasmissione in anticipo nel tempo, in opposizione alla loro generazione in tempo reale, memorizzare vettori di ricezione previsti, e memorizzare una varietà di indicazioni di errori ed altre informazioni riguardanti il comportamento del DUT ottenuto durante il collaudo (vi sono anche altri scopi di gestione generale interna al funzionamento della apparecchiatura di collaudo che impiegano RAM e che possono risultare rientrare entro l'ambito della

frase "memoria interna". Questi sono riservati al funzionamento della apparecchiatura di collaudo, tendono a non essere visibili al livello algoritmico, e sono confrontabili a memorie di istruzioni eseguibili e a registri interni di controllo. Tale memoria è descritta come "memoria interna di controllo" ed è esclusa da ciò che viene qui inteso con il termine "memoria interna di collaudo" che noi impieghiamo per descrivere la memoria impiegata per memorizzare configurazioni di bit direttamente in correlazione con lo stimolo di, e risposta dal DUT). E' facile comprendere che questa memoria interna di collaudo deve funzionare almeno altrettanto velocemente quanto i collaudi che sono eseguiti; un paradigma molto comune è che la memoria interna di collaudo (o una qualche sua parte venga indirizzata con lo stesso indirizzo o qualche indirizzo derivato da questo) così come viene applicato al DUT. Cosa viene poi memorizzato nella collocazione indirizzata nella memoria interna di collaudo è in un certo senso indicativa del comportamento del DUT durante una operazione di collaudo eseguita sul DUT in quell'indirizzo. Considerazioni algoritmiche entro il programma di collaudo possono intendere che la sequenza di

indirizzi associata con vettori di trasmissione consecutivi possono essere arbitrarie. Conseguentemente, la memoria interna deve avere doppi attributi di alta velocità e indirizzabilità casuale. Le SRAM vengono in mente immediatamente come essere veloci, di facile controllo e tolleranti nell'indirizzamento completamente causale. In effetti, apparecchiature di collaudo di memoria convenzionali hanno impiegati SRAM come loro memoria interna di collaudo.

Sfortunatamente, le SRAM sono molto costose e ciò ha limitato la quantità di memoria interna di collaudo con la quale le apparecchiature di collaudo di memoria debbono lavorare. Il risultato consiste in limiti nella funzionalità dell'apparecchiatura di collaudo di memorie che vengono imposte da una quantità insufficiente di memoria. Le DRAM sono significativamente meno costose, ma non possono tollerare l'indirizzamento casuale e lavorare nel contempo ad alta velocità.

Le DRAM possono sostituire le SRAM come memoria interna di collaudo in una apparecchiatura di collaudo di memoria. Come descritto oltre brevemente, il problema di aumentare la velocità del funzionamento di DRAM per l'impiego come

memoria interna di collaudo può essere risolto aumentando la quantità di DRAM utilizzata, in luogo di aumentare la sua velocità. Numeri di banchi identici di DRAM sono trattati come Group. Una combinazione di segnali di interfogliatura per diversi Bank di memoria in un Group di queste e la moltiplicazione tra questi Group di Bank rallenta il traffico di memoria per un qualsiasi Bank fino ad una velocità che può essere gestita dal Bank.

Al livello di testa della organizzazione interna della memoria di collaudo vi sono quattro Memory Set, ciascuno avente il suo proprio ed indipendente e separato spazio di indirizzo e eseguire le richieste transazioni di memoria. Due sono di SDRAM come descritto precedentemente, e due sono di SRAM. Ciascun Memory Set ha il suo proprio dispositivo di controllo al quale sono indirizzate le transazioni di memoria. Per quanto riguarda le capacità operative visibili dall'esterno come memorie, tutti i quattro Memory Set sono sostanzialmente identici. Questi si riferiscono soltanto nella loro grandezza di spazio e di memoria e come sono realizzati all'interno: i Memory Set di SRAM non impiegano la moltiplicazione ed interfogliatura, dato che queste sono

sufficientemente veloci tanto per iniziare. Nonostante la loro indipendenza, i Memory Set dello stesso tipo (di SRAM o di DRAM) possono essere "impilati", cioè, tratti come un più grande spazio di indirizzo.

Conseguentemente, la memoria interna di collaudo della apparecchiatura di collaudo è divisa in quattro Memory Set, due dei quali sono SRAM "interne" e due dei quali sono DRAM "esterne". Per essere sicuri, tutta questa memoria è fisicamente all'interno della apparecchiatura di collaudo di memoria; i termini "interno" e "esterno" hanno più a che fare con un livello di integrazione. Le SRAM sono parti integrali di un circuito VLSI (Very Large Scale Integration) associato con la circuiteria funzionale centrale dell'apparecchiatura di collaudo, mentre le DRAM sono parti singolarmente confezionate montate in adiacenza al materiale VLSI. La quantità di SRAM è piuttosto piccola, ad esempio circa 1 megabit per Memory Set) mentre la quantità di DRAM è apprezzabile e selezionabile (ad esempio, nell'intervallo da 128 a 1024 megabit per Memory Set). I Memory Set del tipo SRAM sono sempre presenti, e possono essere impiegati per qualsiasi

scopo adatto, quali la memorizzazione del contenuto previsto di un DUT che è una ROM (Read Only Memory). I Memory Set DRAM, sebbene in effetti opzionali, sono tipicamente impiegati per creare una traccia per una analisi successiva che conduce alla riparazione, sebbene vi possano essere anche altri impieghi. La apparecchiatura di collaudo non deve necessariamente, in linea di principio imporre distinzioni tra Memory Set del tipo SRAM o del tipo DRAM, per quanto riguarda i diversi scopi per cui questi possono essere impiegati. Vi sono alcune distinzioni pratiche sorgono principalmente come questione di grandezza; i Memory Set in SRAM sono piccoli, mentre i Memory Set in DRAM sono potenzialmente grandi. La persona o persone che creano la programmazione di collaudo generalmente fanno decisioni riguardanti come devono essere impiegati i vari Memory Set. Vi sono, tuttavia, alcune distinzioni dove una particolare caratteristica operativa della apparecchiatura di collaudo di memorie richiede l'impiego di uno specifico Memory Set. Questi casi sorgono di solito da considerazioni di tipo economico o di prestazioni che richiedono un percorso hardware dedicato verso un Memory Set. Anche se questi

meccanismi potrebbero essere generalizzati, è conveniente semplicemente prenderne uno a piacere, e andare avanti in questo modo.

L'avvento di quantità imponenti di memoria interna di collaudo (sotto forma di Memory Set del tipo DRAM) solleva questioni di come questa quantità aggiuntiva di memoria possa essere impiegata per facilitare il funzionamento di caratteristiche desiderabili entro l'apparecchiatura di collaudo di memoria. Nella apparecchiatura di collaudo di interesse, il sottosistema della memoria interna di collaudo è estremamente flessibile, per il fatto che nonostante questo abbia una parola nativa di 32 bit, la effettiva larghezza di parola può essere una qualsiasi potenza di 2 (fino a $2^5 = 32$), con un corrispondente aumento nello spazio di indirizzo per parole più ristrette. Vi è una capacità estesa di mappatura di indirizzo, sia per l'indirizzamento dei DUT sia per l'indirizzamento della memoria interna di collaudo, notevoli meccanismi di classificazione di indirizzo e di classificazione di dati che facilitano le Tag RAM multiple e altri utensili di analisi di errore, tutti i quali sono resi più pratici dal fatto di avere una grande

quantità di memoria interna di collaudo. Inoltre, questi miglioramenti resi possibili da una quantità maggiore di memoria non esistono nel vuoto; questi sono molto preziosi nel collaudo di alcuni tipi di parti di memoria.

E' cosa convenzionale per una apparecchiatura di collaudo di memoria avere una capacità chiamata "post-decode" ("post-decodifica"). L'idea è che, dopo che sono state effettuati i collaudi su un DUT, la memoria interna di collaudo, che può essere divisa in svariate sezioni (organizzate in modo vario come tabelle, gli stati, tracce o "immagini" di eventi di interesse), conterrà dati pronti per la ispezione. Per la maggior parte, queste ispezioni saranno di scoprire indicazioni di avaria, e nel sistema di interesse una indicazione di una avaria e codificata da uno zero. E' un pò una semplificazione, ma ciò che è spesso necessario, e ciò che si prevede che faccia il meccanismo di post-decode, è di totalizzare il numero di zero in una qualche struttura nella memoria interna di collaudo. Apparecchiature di collaudo di memoria della tecnica anteriore dovevano finora fare in modo che i dati di collaudo da memorizzare in una qualche struttura di memoria

nella memoria interna di collaudo, quale una Error Catch RAM, prima di essere applicati al meccanismo di Post-Decode.

Apparecchiature di collaudo di memoria della tecnica anteriore con una capacità di Post-Decode erano limitati in modo in modi che attualmente sono ritenuti come provocare un tempo di collaudo aumentato. I fabbricanti di componenti di memoria in alti volumi richiedono che "il tempo sulla apparecchiatura di collaudo costituisce denaro" e sono a favore di strategie che riducono il tempo di collaudo. L'avvento di quantità aumentate di memoria interna di collaudo, in unione con tecniche di classificazione di dati e di indirizzi che consentono la produzione di molteplicità di "Tag RAM" (tabelle i cui ingressi hanno significati altamente raffinati, e che sono quindi piccoli in confronto, ad esempio, ad una semplice immagine di indirizzi rispetto ad errori che è sostanzialmente equivalente in grandezza al DUT stesso) hanno posto le tecniche convenzionali di Post-Decode nella posizione di essere un fattore limitante per il compito di ridurre il tempo di collaudo. Questa situazione semplicemente peggiora quando le memorie (DUT) divengono più grandi e più complesse al loro

interno. La complessità aumentata sta a significare che vi sono più strutture di memoria (Tag RAM, ecc) nella memoria interna di collaudo da applicare alla sezione di Post-Decode, ed una grandezza aumentata dei DUT significa che queste strutture di memoria diventano anche più grandi, il tutto si aggiunge ad aumentare il tempo di collaudo.

Cosa può essere fatto per rendere il funzionamento del Post-Decode più veloce e più efficiente? E, vi sono nuovi tipi di misure che un Post-Decode a capacità aumentate possa effettuare per aumentare la unità del dispositivo di collaudo e di memorie?

Sommario dell'invenzione

Si possono fare molte cose per aumentare le prestazioni del funzionamento del Post-Decode. Nella forma di realizzazione preferita descritta oltre, il percorso di dati nella sezione di Post-Decode viene modificata per consentire che il Post-Decode elabori dati adatti "al volo" come se tali dati fossero stati prima collocati in una struttura di memoria di destinazione nella memoria interna di collaudo. Altri dati continueranno ad essere collocati dapprima in una struttura di memoria nella memoria interna di collaudo prima di essere

applicati al meccanismo di Post-Decode. Una capacità estesa di mascheratura, accoppiata con copie di tabelle di errore consente una analisi di Post-Decode incrementale per un nuovo collaudo, ed evita il conteggio di errori in collocazioni di cui si sa che hanno già fallito durante uno o più collaudi precedenti. Si possono accumulare errori con ambiti diversi o diverse risoluzioni; si possono accumulare sia errori entro parole ed errori di bit. La architettura interna di un meccanismo di Post-Decode a prestazioni aumentate è spesso in grado di produrre più tipi di risultati da una singola passata attraverso i dati, sia che questi vengono applicati al volo e da una struttura nella memoria interna di collaudo. Il meccanismo di PostDecode possiede in esso numerosi contatori, e questi conteggiano a scendere da valori precaricati che rappresenta soglie per decidere qualcosa circa la attività di errore. Quando un contatore conteggia fino a zero, questo produce un flag di conteggio terminale. I valori dei vari flag di conteggio terminale sono disponibili in un qualsiasi momento come dati da memorizzare (i loro valori registrati) in una struttura della memoria interna di collaudo. Infine, è spesso il caso che

ciascuna contatore venga ricaricato con il suo conteggio iniziale alla conclusione di una fase di collaudo ed in preparazione ad una fase successiva. Si può risparmiare un certo lavoro a monte e si può aumentare utilità disponendo i contatori in modo che abbiano rispettivi registri di valore iniziale dai quali questo possono essere ricaricati in seguito alla ricezione di un singolo comando, in contrapposizione al dover inviare questi vari valori iniziali sempre di nuovo attraverso un bus. In aggiunta, è desiderabile predisporre la presenza di un flag di conteggio terminale per un contatore possa impedire il ricaricamento del contatore dal suo registro di valori iniziale.

Breve descrizione dei disegni

La figura 1 è uno schema a blocchi semplificato di una apparecchiatura di collaudo di memorie non volatili estensivamente riconfigurabile secondo l'invenzione;

la figura 2 è una espansione semplificata dello schema a blocchi della unità di collaudo 6 di DUT della figura 1;

la figura 3 è uno schema a blocchi semplificato funzionale del meccanismo della memoria interna di collaudo che si presenta nello

schema a blocchi di figura 2;

le figure 4A/B sono uno schema a blocchi semplificato del meccanismo 60 di Post-Decode della figura 2;

la figura 5 è uno schema a blocchi di un circuito di maschera impiegato nello schema a blocchi di figura 4;

la figura 6 è uno schema a blocchi di un circuito di maschera e di contatore a bit impiegato nello schema a blocchi di figura 4; e

la figura 7 è uno schema a blocchi semplificato di circuiteria di alcuni contatori e flag di conteggio terminale impiegato in figura 4B. Descrizione di una forma di realizzazione preferita

Si faccia riferimento ora alla figura 1, in cui viene mostrato uno schema a blocchi 1 semplificato di un sistema di collaudo di memorie non volatili costruito secondo i principi dell'invenzione. In particolare, il sistema può simultaneamente collaudare, con fino a sessantaquattro punti di prova ciascuno, fino a trentasei singoli DUT (Device Under Test) allo stesso momento, con predisposizione per riconfigurazione per consentire a elementi di una raccolta di sorgenti di collaudo di essere uniti

assieme per collaudare DUT aventi più di sessantaquattro punti di prova. Questi punti di prova possono essere collocazioni su una parte di un wafer di circuito integrato che non è stato ancora tagliato e confezionato, oppure potrebbero essere i piedini di una parte confezionata. Il termine "punto di prova" si riferisce ad una collocazione elettrica nella quale può essere applicato un segnale (ad esempio alimentazioni, clock, ingresso dati) o dove si può misurare un segnale (ad esempio una uscita dati). Sarà seguito l'uso industriale di far riferimento ai punti di prova come "channels". La "raccolta di risorse di collaudo da unire assieme" cui è stato fatto riferimento precedentemente si può intendere come essere fino a trentasei siti di collaudo, dove ciascun sito di collaudo include un Test Site Controller (4), un DUT Tester (a sessantaquattro canali) (6) è una raccolta (a sessantaquattro canali) di Pin Electronics (9) che effettua il collegamento elettrico effettivo verso un DUT (14). Nel caso in cui il collaudo del DUT richiede sessantaquattro o meno canali, un singolo Test Site è sufficiente per effettuare prove su quelle DUT, possiamo dire ad esempio che il Test Side #1 (così

come appare nella figura 1) forma o opera come una "stazione di prova a singolo sito". D'altro canto, quando una qualche forma della predetta riconfigurazione è in effetto, due (o più) Test Site sono "legati" assieme per funzionare come un Test Site equivalente più grande avente 128 canali. Conseguentemente, ed ancora in riferimento ad un esempio mostrato in figura 1, si può dire che il Test Site #35 e #36 formano una "two Site Test Station".

Per considerare brevemente un caso contrapposto, non si dovrebbe supporre che sia necessario un intero Test Site per collaudare un singolo DUT, oppure che un singolo Test Site possa collaudare soltanto un singolo DUT. SI supponga che un wafer abbia due, tre o quattro (probabilmente ma non necessariamente adiacenti) die, la somma dei quali requisiti di canale di prova sia di sessantaquattro canali o inferiore. Tali DUT (15a-d) possono essere collaudati simultaneamente da un singolo Test Site (ad esempio il Test Site #2 come mostrato in figura 2). Ciò che rende ciò possibile è la programmabilità per impieghi generali di ciascun Test Site, come incrementata da alcune caratteristiche in hardware che verranno descritte

in tempo debito. In linea di principio, un programma di collaudo eseguito dal Test Site potrebbe essere scritto in modo che una parte delle risorse del Test Site venga impiegata per collaudare uno dei DUT mentre un'altra parte viene impiegata per collaudare l'altro DUT. Dopo tutto, verrebbe supposto che se si avesse un terzo DUT che fosse la unione logica dei primi due, allora si sarebbe in grado di collaudare quel terzo DUT con un singolo Test Site in modo che si dovrebbe essere in grado di collaudare il suo "DUT componente" come se questo fosse così. Una differenza importante è, ovviamente, il mantenere singolarmente traccia di quale dei due DUT componenti "supera o fallisce il collaudo, in contrapposizione ad una semplice risposta unificata per il "terzo" DUT. Cioè, vi è un argomento riguardante quale parte del "terzo" DUT ha fallito. Vi sono altri argomenti egualmente, che includono la rimozione o limitazione dei segnali di pilotaggio ad un DUT difettoso, con diramazioni entro il programma di collaudo basate su quale DUT indica una avaria, mentre nello stesso tempo si impedisce che il programma di prova diventi diramato in modo intrattabile. Alcuni semplici aspetti di questa capacità di "Multi-DUT

Test Station" come singolo Test Site sono piuttosto semplici, mentre altri sono complessi. Il collaudo a Multi DUT non deve essere confuso con la nozione di legare assieme due o più Test Site assieme.

Se non fosse per questa nozione di riconfigurazione dei Test Site, non vi sarebbe alcuna differenza tra un Test Site e una Test Station, e si potrebbe fare a meno di uno di questi termini. Nella situazione in questione, tuttavia, si comprenderà facilmente che il numero di Test Station non richiede di essere eguale al numero dei Test Site. In passato, i numeri potevano essere diversi poichè i Test Site erano talvolta suddivisi per creare più Test Station per un semplice collaudo Multi DUT (DUT non complessi sufficientemente per consumare un intero Test Site). Ora, tuttavia, la differenza può essere anche dovuta al fatto che test Site sono stati uniti assieme per formare Test Station multi-sito (DUT troppo complessi per un singolo Test Site).

Per continuare, quindi, un Test System Controller 2 viene collegato con un System Bus 3 (o bus di sistema) per fino a trentasei Test Site Controller i cui nomi terminano con i suffissi #1 fino a #36 (4a-4z) (è vero che i pedici a-z vanno

soltanto da uno a ventisei, e non fino a trentasei. Ma questo piccolo inganno sembra preferibile su pedici numerici su caratteri di riferimento numerici, che sarebbero potenzialmente molto suscettibili di creare confusione. Il Test Site Controller 2 è un calcolatore, ad esempio un PC che marcia con NT) che esegue un adatto Test System Control Program che è relativo al compito di collaudare memorie non volatili. Il Test System Control Program rappresenta il livello più elevato di astrazione in una divisione gerarchica del lavoro (vedi complessità) per ottenere il collaudo desiderato. Il Test System Controller determina quali programmi debbano essere fatti marciare dai diversi Test Site, così come il sorvegliare un sistema robotizzato (non mostrato) che sposta le sonde di prova e i DUT a seconda della necessità. Il Test System Controller 2 può funzionare in modi che supportano la nozione che alcuni Test Site siano programmati per effettuare Test Station a singolo sito, o Test Station Multi-DUT, mentre altri sono uniti assieme per formare Test Station a più siti. Chiaramente, in tali circostanze vi sono diverse parti che sono sotto collaudo, è del tutto desiderabile che vengano usati diversi collaudi per

parti diverse. Similmente, non vi è alcun requisito che tutte le Test Station a singolo sito stiano collaudando lo stesso tipo di parti, neanche non vi è alcun requisito per le Test Station multi-sito. Conseguentemente, il Test System Controller 2 è programmato per emettere comandi per eseguire la necessaria unione di Test Site e quindi chiamare gli appropriati programmi di collaudo per le varie Test Station in uso. Il test System Controller 2 anche riceve informazioni circa i risultati ottenuti dai collaudi, in modo che questo possa intraprendere una azione appropriata per scartare le parti difettose e in modo che mantenga una registrazione per le varie analisi che possono essere impiegate ad esempio per controllare i processi di produzione nella impostazione di una fabbrica.

Il Test System di per sè è un sistema piuttosto grande e complesso, ed è comune impiegare un sottosistema robotizzato per caricare wafer su un tavolo traslatore che posiziona in sequenza uno o più futuri die sotto sonde collegare alla Pin Electronics 9, in seguito alla quale cosa questi futuri die (il Wafer non è stato ancora tagliato) sono collaudati. Il Test System può anche essere

impiegato per collaudare parti confezionate che sono state caricate su un adatto supporto. Vi sarà (come verrà spiegato oltre) almeno un Test Site Controller associato con ciascuna Test Station in uso, quale che sia il numero dei test Site che sono impiegati per formare quella Test Station, o quante Test Station vi siano su un Test Site. Un Test Site Controller è un sistema ad elettronica incorporato che può essere un elaboratore i960 della Intel con trentasei fino sessantaquattro MB di memoria combinata di programmi e di dati che marciano su un sistema operativo riservato chiamato VOS (VersaTest O/S), che è stato anche impiegato in prodotti più antichi per collaudare memorie non volatili (ad esempio la Agilent V1300 oppure V3300). Per il momento verrà considerata soltanto la situazione per Test Station a singolo sito. Per i fini di un esempio definito si supponga che il Test Site #1 stia funzionando come Test Station #1 e che si debba collaudare il componente WHIZCO n° 0013. Il regime di collaudo comporta un centinaio o circa tipi diversi di collaudi (variare e monitorare i livelli di tensione, larghezze di impulso, posizioni dei fronti, ritardi, così come una grande quantità di semplicemente memorizzare e quindi

recuperare configurazioni selezionate di informazioni) e ciascun tipo di collaudo comporta numerosi milioni di singoli cicli di memoria per il DUT. Al livello più elevato, gli operatori del Test System istruiscono il Test System Controller 2 ad usare la Test Station #1 per iniziare il collaudo dei WHIZCO 0013. A tempo debito il Test System Controller 2 dice al Test Site Controller #1 (4a) (che è un sistema a [calcolatore] incorporato) per far marciare il programma di collaudo associato, ad esempio, TEST_WHIZ_13. Se tale programma è già disponibile entro l'ambiente del Test Site Controller #1, allora questo viene semplicemente eseguito. Nel caso contrario, allora questo è fornito dal Test System Controller 2.

Ora, in linea di principio, il programma TEST_WHITZ_13 potrebbe essere interamente autonomo. Ma se così fosse, allora questo sarebbe sicuramente piuttosto grosso, e potrebbe essere difficile per il processore del sistema incorporato entro il Test Site Controller 4a marciare sufficientemente rapidamente per produrre i collaudi alla velocità desiderata, o anche ad una cadenza che sia uniforme da un ciclo di memoria di DUT fino al prossimo. Conseguentemente, vengono generate attività del

tipo di subroutine a basso livello che generano sequenze di indirizzi e dati associati che debbono essere scritti o che sono previsti da una operazione di lettura, a seconda delle necessità mediante un meccanismo algoritmico programmabile collocato nel Tester 6 del DUT, ma che opera in sincronismo con il programma che viene eseguito dal sistema incorporato nel Test Site Controller . Si deve pensare di ciò come esportare una certa attività simile a subroutine di basso livello e il compito di avviare i siti di memoria del DUT verso un meccanismo (il dispositivo di collaudo del DUT) che è più prossimo all'ambiente Hardware del DUT 14. Parlando in termini generici, allora, ogni volta che il Test System Controller 2 equipaggia un Test Site Controller con un programma di collaudo questo anche fornisce il DUT Tester associato con le appropriate routine di implementazione a basso livello (eventualmente specifiche per la memoria che viene collaudata) necessarie per svolgere la completa attività descritta e necessaria mediante programmazione per il Test Site Controller. Le routine di implementazione a basso livello sono chiamate "configurazioni", e queste sono generalmente nominate (esattamente come funzioni e

variabili in linguaggi di programmazione ad alto livello hanno nomi).

Ciascun test Site Controller #n (4) viene accoppiato al suo DUT Tester #n (6) mediante un Test Site Bus #n (5). Il test Site Controller impiega il Site Test Bus 5 sia per controllare il funzionamento del DUT Tester e sia per ricevere da questo informazioni circa i risultati dei collaudi. Il DUT Tester 6 è in grado di generare ad alta velocità i vari cicli di memoria del DUT che sono coinvolti nel regime di collaudo, e se decide che i risultati di un ciclo di memoria di Read sono come previsti. In essenza, questo risponde a comandi o codici operativi ("chiamati configurazioni") inviati dal Test Site Controller avviando sequenze utili corrispondenti di Read and Write di cicli di memoria del DUT (cioè, questo esegue le configurazioni corrispondenti. Concettualmente, la uscita del DUT Tester 6 è una informazione di stimolo che deve essere applicata al DUT, ed anche accetta informazione di risposta da questo. Questa informazione di stimolo/risposta 7a passa tra il DUT Tester 6a ed un complesso 9a di Pin Electronics #1. Il complesso 9a di Pin Electronics supporta fino a sessantaquattro sonde che possono essere

applicate al DUT 14.

La informazione di stimolo sopra menzionata è soltanto una sequenza di configurazioni di bit in parallelo (cioè una sequenza di "vettori di trasmissione" e "vettori di ricezione" previsti) espressi secondo i livelli di tensione di una qualche famiglia di dispositivi logici impiegati nel DUT Tester. Vi è una mappatura configurabile tra le posizioni di bit entro uno stimolo/risposta e le sonde che vanno al die, e questa mappatura è compresa dal DUT Tester 6. I singoli bit sono corretti per quanto riguarda la loro temporizzazione e collocazione dei fronti, ma in aggiunta alla mappatura questi possono anche richiedere spostamenti di livelli di tensione prima che questi possano essere applicati al DUT. Similmente, una risposta che si origina nel DUT successivamente ad uno stimolo può richiedere la necessità di un wafer e uno spostamento di livello (inverso) prima che questo possa essere considerato adatto per essere ricondotto indietro al DUT Tester. Questi compiti di spostamento di livello sono nell'ambito della Pin Electronics 9a. La configurazione della Pin Electronics necessaria per collaudare un WHIZCO 0013 probabilmente non

funzionerà per collaudare una parte dalla ACME Co., ed eventualmente neanche con un altro componente WHITZE Co. Così, si comprenderà che il complesso di Pin Electronics richiede di essere anche esso configurabile; tale configurabilità è funzione delle linee PE Config 8a.

Quanto sopra conclude una breve panoramica architettonica di come un singolo Test Site sia strutturato per collaudare un DUT. Si passa ora ad argomenti che sorgono quando vi sono numerosi Test Site con i quali lavorare. In via preliminare, verrà descritta una forma di realizzazione preferita per costruire un Test System avente multipli Test Site. Sotto numerosi aspetti, una certa quantità dell'informazione che stiamo per descrivere è una questione di scelta basata su studi di mercato delle preferenze dei clienti ed analisi costi-benefici. Quale che sia la situazione, per costruire una di queste apparecchiature uno deve fare delle scelte definite, e una volta che ciò è fatto vi sono particolari conseguenze che sono visibili in tutto il sistema nel suo complesso. Si ritiene che sia utile descrivere, almeno in termini generali a larghi contorni le proprietà di hardware del Test

System. Anche se alcune di queste proprietà sono contingenti, una loro conoscenza sarà tuttavia di assistenza nella valutazione di vari esempi impiegati per illustrare l'invenzione.

Per iniziare allora si considerino quattro cestelli per schede piuttosto grandi. Ciascun cestello per schede possiede, oltre le alimentazioni di energia elettrica e raffreddamento ad acqua (i ventilatori possono essere una sorgente di contaminazione in un ambiente "clean room", e l'acqua refrigerata è più economica del condizionamento ad aria per rimuovere le svariate decine di KW di calore dissipato per un sistema interamente caricato), una scheda madre, un "front plane" e un "back plane". In ciascun cestello per schede possono essere collocati fino a nove complessi. Ciascun complesso comprende un Test Site Controller, un DUT Tester e Pin Electronics. Verrà descritto il complesso generico di come i Test Site controller sono uniti assieme, il che comporta alcuni bus impiegati per creare le daisy chain.

E' probabilmente utile una breve digressione riguardante il termine "daisy chain". Si considerino gli elementi di sistema A, B, C e D. Si supponga che questi debbono essere collegati in

daisy in questo ordine. Si potrebbe dire che vi è una informazione o percorso di controllo che lascia A e va entro B, che B può selettivamente far passare traffico e quindi lascia B ed entra in C, e che C possa far passare selettivamente traffico che quindi passa in D. Queste stesse disposizioni possono esistere per il traffico anche nell'altra direzione. Le daisy chain sono spesso impiegate per creare schemi di priorità; verranno usate per creare relazioni master/slave tra vari dei Test Site Controller. Verranno indicate queste disposizioni di comunicazione del tipo daisy chain con il suffisso "DSY" invece di "BUS". Quindi si potrà fare riferimento ad un Command/DataDSY invece di un Command/Data Bus. Ora, la nozione che informazione "entra in B e viene fatta passare selettivamente in avanti" può suggerire che il traffico viene replicato su un separato insieme di conduttori prima di essere fatto passare oltre. Potrebbe essere realizzato in tal modo, ma per ragioni di prestazioni ciò è più simile ad un convenzionale bus avente entità indirizzabili. Per mezzo di una disposizione di mappatura di indirizzo programmabile e la capacità di collocare parti dei Test Site Controller a valle in situazione "di

attesa", il singolo bus può essere reso apparire logicamente (cioè funzionare) come una molteplicità di daisy chain. Infine, si comprenderà che le daisy chain sono percorsi ad alta prestazione per informazione di comando e controllo, e che se non fosse così, allora non ci si potrebbe aspettare una combinazione master/slave (Test Station a più siti) per operare con la stessa velocità di quanto fa un singolo Test Site. Per i vantaggi delle prestazioni daisy chain, i vari DSY non lasciano i rispettivi cestelli di scheda. Effetto di questa decisione è di collocare alcuni limiti su quali Test Site (e quindi anche quanti) possono uniti assieme. In linea di principio, non vi è alcuna necessità fondamentale per questa limitazione, ne vi è una effettiva mancanza di praticità tecnica coinvolta (questa potrebbe essere fatta); è semplicemente ritenuto che, dato che vi sono già nove Test Site in un cestello di schede, la estensione della DSY aggiunge costi significativi con vantaggi aggiuntivi relativamente limitati.

Per riprendere la discussione della figura 1, quindi, si considerino i vari Test Site Controller 4a-4z che possono essere disposti nei quattro cestelli per schede, ciascun con nove test Site

Controller. Questi vengono indicati come 4a-4f, 4g-4m, 4n-4t e 4u-4z. (Non si faccia caso, come spiegato precedentemente, che questi sono nominalmente soltanto 26 pedici - il lettore è invitato ad immaginare che vi siano altri dieci simboli di pedice in una qualche posizione). Un CMD/DAT DSY 17a (Command & Data Daisy Chain) intercollega il Test Site Controller 4a-4f che è disposto in un cestello di schede, mentre un diverso CMD/DAT DSY 17b intercollega i Test Site Controller 4g-4m in un altro cestello per schede. La medesima disposizione esiste per i rimanenti cestelli di schede, e i Test Site Controller 4n-4t e 4u-4z, rispettivamente. E' stato detto precedentemente che DSY non lascia i cestelli delle schede, in quanto la "tail end" di un bus che costituisce in effetti il DSY non lascia un cestello di schede e diviene la testa di un segmento successivo in un altro cestello di schede. Invece, il System Bus 3 dal Test System Controller 2 va a tutti i Test Site Controller, e ciascuno è in grado di divenire un Master alla testa di un segmento DSY che non lascia il cestello delle schede.

Il CMD/DAT DSY 17a-d che è stato discusso

esiste tra vari Test Site Controller 4a-4z. Vi è una simile disposizione per il SYNC/ERR DSY 18a-18d e i DUT Tester 6a-6z. La informazione di sincronizzazione e di errore trasportata dal SYNC/ERR DSY 18 consente ai DUT di funzionare in sincronismo. Questi due daisy chain (17 e 18) possono portare tipi di informazione leggermente diversi, ma questi esistono come parte dello stesso meccanismo generale per unire uno o più Test Site assieme in una Test Station.

Si passa ora ad una discussione della figura 2, che è una espansione semplificata a schema a blocchi del DUT Tester 6 di figura 1, dei quali possono ne esistere fino a trentasei. E' sufficiente al momento descrivere soltanto un caso di questi. Uno sguardo alla figura 2 mostrerà che questa è piuttosto popolata di materiale, in particolare per quello che è uno schema a blocchi "semplificato". Una parte di ciò che si trova nel DUT Tester 6 è rappresentato nello schema a blocchi è funzionalmente molto complicato, e non è disponibile sottoforma "disponibile da magazzino". E' quindi appropriato in questa sede puntualizzare due cose. Dapprima, lo scopo primario di includere la figura 2 è di descrivere le proprietà

fondamentali di un ambiente operativo importante con il Test System 1 di memoria non volatile nel suo complesso. L'invenzione che è interamente descritta in riferimento alla figura 3, e figure successive sarà una espansione di meccanismi esposti nella seguente descrizione di figura 1, oppure saranno nuovi meccanismi la cui premessa di motivazione si trova in figura 2. Nell'uno e l'altro modo, così come descritto non è esattamente noto quali di questi sia davanti al lettore. La meta attualmente è di fornire un punto di partenza semplificato e tuttavia informativo per numerose diverse descrizioni particolareggiate di varie forme di realizzazione preferite, in modo che ciascuna di queste può essere concisa quanto appropriato (in contrapposizione ad una descrizione "monumentale" che illustra tutto a proposito di ciascuna diversa invenzione. Il secondo punto consiste nel fatto che materiale espanso o esteso, che è in accordo generico complessivo con la figura 2, può contenere informazione che non si "accoppia" esattamente con la versione semplificata. Ciò non significa che vi è stato un errore, o che le cose sono fatalmente non congruenti; ciò sorge per il fatto che talvolta è difficile o impossibile

semplificare qualcosa in modo che si ottenga la sua immagine esatta in miniatura. La situazione è piuttosto simile a quella delle carte geografiche. Una carta stradale di grandezza unificata del Colorado mostrerà che quando si va ad est sulla I-70, voi potete andare a nord sulla I-25 a Denver. Ciò sembra una svolta a sinistra. E anche se questa è di solito una effettiva svolta a sinistra, non è in effetti così, ed una mappa dettagliata degli incroci mostrerà una sequenza di svolte e tratti di strada interposti. Ma nessuno direbbe che la carta stradale di grandezza standard è erronea; questa è corretta per il suo livello di astrazione. Similmente, e nonostante il suo aspetto abbastanza ingarbugliato, la figura 2 è in effetti una semplificazione operante a livello medio di astrazione, ma ciò che sembra essere svolte a sinistra non sono affatto semplici svolte a sinistra. Così come è mostrato in figura 1, l'ingresso principale al DUT Tester 6 è un caso del Test Site Bus 5, che si origina da un Test Site Controller 4 che è associato con il caso del DUT Tester 6 che è di interesse. Il test Site Bus 5 è accoppiato ad un Multi-Bus Controller 88 che converte traffico sul Test Site Bus a traffico su

un Ring Bus 85 o un VT Bus 89. Il traffico del Ring Bus può anche convertirsi in traffico BT Bus, e viceversa. Pressochè ogni cosa nella figura 2 è parte di un qualche circuito integrato su larga scala, il circuito Timing/Formatting & Comparison 52 (descritto oltre) è in effetti costituito da otto di tali circuiti integrati, sebbene sia stato mostrato come una singola entità per concisione. Ad eccezione del vari Ext DRAM (alcune delle quali sono anche parte della Interior Test Memory 87 - vedere figura 3), la maggior parte del materiale di figura 2 fa parte di un altro grosso circuito integrato chiamato APG (Automatic Pattern Generator). Il Ring Bus 85 è un percorso di comunicazione tra meccanismi di scopi generali per configurare gli elementi principali entro la parte APG del DUT Tester 6, e per impostare modi di funzionamento, ecc. Vi sono anche altri percorsi ad alta velocità e molti ampi tra vari elementi dello APG. Il VT Bus 89 è un bus da circuito integrato a circuito integrato per l'impiego entro il DUT Tester stesso.

Il Ring Bus 85 è il meccanismo mediante il quale il Test Site Controller comunica con la parte APG del DUT Tester 6. Il Ring Bus 85 è accoppiato

ad un Micro-Controller Sequencer 19, che può essere considerato come un microprocessore per scopi speciali. Impiegando un indirizzo creato da un Next Address Calculator 102, questo ricerca istruzioni da un programma memorizzato in una memoria di programma, che può essere o interna al Micro-Controller Sequencer 19 (PGM SRAM 20) o esterna a questo (EXT.DRAM 21). Sebbene queste due memorie appaiono essere indirizzate da ciò che è essenzialmente un indirizzo logico comune 63 che serve come contatore di programma (o indirizzo di ricerca di istruzione), e può essere una sorgente di programmazione da eseguire, si deve notare che:

(1) soltanto una delle memorie esegue cicli di memoria di ricerca di istruzione durante un qualsiasi periodo di tempo; e (2) in effetti queste sono indirizzate da segnali elettricamente diversi. La SRAM è veloce e consente un vero accesso casuale, ma consuma spazio prezioso entro il Micro-Sequencer 19 (che fa parte del grosso circuito integrato APG IC), per cui la sua grandezza è limitata. La DRAM esterna può essere fornita in quantità aggiustabili di considerevole grandezza, ma è veloce soltanto quando a questa si accede in gruppi sequenziali che comportano una esecuzione

lineare e nessuna diramazione. La programmazione nella SRAM 20 è più spesso ciò che è intensamente algoritmico, mentre la EXT.DRAM 21 è meglio adatta per materiale non facilmente generato con processi algoritmici, quali le routine di inizializzazione e dati casuali o irregolari.

Il Next Address Calculator 102 può realizzare diramazioni nel programma di collaudo che viene eseguito, in risposta ad un salto non condizionato o ad istruzioni o ad un salto condizionato o istruzioni condizionate di subroutine condizionate su vari PROGRAM CONTROL FLAGS (25), OTHER FLAGS (55), e alcuni altri segnali che, per chiarezza, sono mostrati separatamente (DFE 0:3 104 e DPE 0:3 104) e che sono previsti per il funzionamento su più DUT.

La parola di istruzione prelevata ed eseguita dal Micro-Controller Sequencer 19 è piuttosto ampia: duecentotto bit. Questa è costituita da tredici campi a sedici bit. Questi campi spesso rappresentano informazione di istruzione prelevata per meccanismi che si trovano fuori del Micro-Controller Sequencer vero e proprio. Tali campi sono dedicati ai loro meccanismi associati. Un insieme di ALU INSTRUCTIONS 22 viene applicato ad

una raccolta di otto ALU 24 a sedici bit, mentre altri sono erogati a vari altri meccanismi distribuiti per tutto il DUT tester. Quest'ultima situazione è rappresentata da linee e leggende "VARIOUS CONTROL VALUES & INSTRUCTIONS" 42.

Le otto ALU (24) a sedici bit hanno ciascuna un repertorio convenzionale di istruzioni aritmetiche, costruite attorno a registri di risultato associati a sedici bit (ciascuna ALU possiede anche altri registri). Tre di questi registri di risultato e le loro ALU associate sono previste per generare componenti di indirizzo 27 X, Y e Z che sono combinati in vario modo in un indirizzo completo applicato al DUT. Altri due degli otto registri delle ALU (DH & DL) sono previsti come ausilio nella creazione algoritmica di configurazioni 28 di dati a trentadue bit che sono divise tra una parte più significativa (DH) ed una parte meno significativa (DL). Tre ALU/registri finali (A, B, C) sono impiegati come contatori che contribuiscono alla produzione di vari PROGRAM CONTROL FLAGS 25 che assistono con controllo e diramazioni di programma al completamento di un qualche numero specificato a programma di iterazioni o altre condizioni numeriche. Questi

PROGRAM CONTROL FLAGS 25 sono ricondotti indietro al Micro-Controller Sequencer 19, dove questi influenzano il valore dell'indirizzo di ricerca di istruzione (creato dal Next Address Calculator 102) in modi familiari a coloro che comprendono i meccanismi di esecuzione microprogrammati. Vi sono anche vari OTHER FLAGS 55 che anche possono essere impiegati per effettuare diramazioni di programma. Questi si originano con vari degli altri meccanismi entro il DUT tester 6 che sono controllati da campi diversi della parola di istruzione prelevata. Uno specifico flag aggiuntivo viene mostrato espressamente come elemento separato: PD_ERR 90. Questo è applicato alla PGM SRAM 20, si origina con il meccanismo 60 di Post Decode, e indica che il meccanismo 60 di Post Decode ha scoperto un errore. Un'altra di queste flag aggiuntive è VEC_FIFO_FULL 26. In un altro disegno avente dettagli un poco inferiori ciò potrebbe essere raggruppato assieme a OTHER FLAGS 55. Questo è stato separato per aiuto nell'applicazione di un aspetto del funzionamento del Micro-Controller Sequencer 19.

Ciò che fa VEC_FIFO_FULL è di arrestare (temporaneamente) la ulteriore esecuzione del programma da parte del Micro-Controller Sequencer

19. Vi sono numerosi stadi di pipeline tra le istruzioni prelevate dal Micro-Controller Sequencer 19 ed il meccanismo che alla fine produce vettori di collaudo che debbono essere applicati alla DUT. In aggiunta, parte del bagaglio che accompagna un vettore mentre questo si sposta verso l'essere applicato alla DUT è informazione riguardante la cadenza di applicazione finale di vettore, o, ciascuna durata di vettore. Conseguentemente, la cadenza di applicazione di vettore alla DUT non richiede di essere costante, ed in particolare un gruppo di vettori può impiegare più tempo per essere applicato di quanto sia necessario per generarli. Il Micro-Controller Sequencer semplicemente esegue programmazione alla sua velocità massima. Ma chiaramente, sulla media, la cadenza di "consumo di vettori" deve essere eguale alla cadenza di "produzione di vettori" mentre il pipeline deve essere elasticamente quasi senza limite. Vi è un Vector FIFO 45 sull'uscita dello Address Mapper 29 discusso oltre, e questo serve come capacità elastica nel pipeline. Il segnale VEC_FIFO_FULL viene impiegato per evitare il trabocco del numero limitato di stadi nel pipeline, provocando una cessazione temporanea della

produzione di nuovi vettori sull'estremità di testa del pipe.

Per continuare, le componenti di indirizzo 27 di X, Y e Z (tre moltiplicato sedici uguale quarantotto bit) sono applicati ad un Address Mapper 29, la cui uscita è una ridisposizione quasi arbitraria selezionata in anticipo dei valori di indirizzo dello spazio ordinato di indirizzo a quarantotto bit. Come punto di partenza per comprendere ciò, si supponga per un momento che il Address Mapper 29 sia una memoria che ha popolato interamente lo spazio di indirizzi a quarantotto bit, e che trattiene un valore di quarantotto bit in ciascun indirizzo. (Per il momento non si faccia caso al fatto che tale memoria, potrebbe essere di una grandezza - oggi in ogni modo - di un grosso frigorifero). Data una tale memoria, si potrebbe realizzare una tabella di ricerca che potrebbe mappare qualsiasi indirizzo in un altro valore a quarantotto bit selezionato arbitrariamente che potrebbe essere poi usato come indirizzo di sostituzione. La ragione di tale mappatura di indirizzo è desiderabile per il fatto che le componenti di indirizzo X Y e Z generalmente hanno un significato utile nel contesto di una

particolare architettura interna di DUT che molto probabilmente è non implementata con un grosso decodificatore lineare. Le nozioni di righe, colonne e strati, blocchi o pagine può essere molto utile per un ingegnere di collaudo, e le avarie che si verificano in collocazioni che sono fisicamente vicine l'una all'altra possono comportare una corrispondente vicinanza nei loro indirizzi X, Y e Z. Tali configurazioni nei risultati di collaudo possono essere preziose nel valutare ciò che è errato e nel cercare di ripararlo, sia che ciò sia a livello di progetto o a livello di produzione di riprogrammare una parte per sciuntare una operazione di sezione difettosa con una sezione di ricambio. Sorgono due considerazioni da tale modo di pensare. La prima è quella di confrontare i quarantotto bit fino al numero effettivo di bit (ad esempio trentadue o eventualmente sedici) perché devono essere applicati alla DUT. In breve menzioneremo come tale appaiamento venga effettuato, ed è ampiamente una questione di prendere questo numero di bit da X altrettanti da Y ed il resto da Z. Però non interamente, e questo è il secondo elemento, poiché alcuni indirizzi potrebbero ritrovarsi entro circuiteria che è

un'immagine speculare destra contro sinistra (o sinistra contro destra o alto contro basso) di un'altra parte della circuiteria. Ciò ha l'effetto di riarrangiare ciò che significano i bit, in quanto i valori di indirizzo sequenziali si trovano in ordine fisico entro tale circuiteria. Questa proprietà di layout del chip può verificarsi numerose volte, e può benissimo essere il caso che come un gruppo di bit per, ad esempio Y, sono interpretati, può dipendere dal valore associato di altri, ad esempio, Z bit. Il mappatore 29 di indirizzi è predisposto per consentire che gli indirizzi grezzi X Y e Z vengano "reimpaccati" come dovrebbero essere, per riflettere questo genere di cose a vantaggio di coloro che effettuerebbero collaudi di memoria aventi una tale disposizione architettonica interna. Per come ciò viene effettivamente effettuato, il Address Mapper 29 è costituito da un numero piuttosto grande di multiplatori intercollegati. Questo non può implementare il comportamento di una tabella di ricerca completamente arbitraria di uno schema di decodifica di memoria interamente popolata come era stato supposto temporaneamente precedentemente per scopi di illustrazione. Questo può tuttavia,

riarrangiare sotto-campi delle componenti di indirizzo X Y e Z a seconda delle necessità, in particolare dato che vi è ancora un altro meccanismo che effettuerà l'appaiamento in basso da quarantotto bit al numero effettivo necessario. Il Address Mapper 29 contiene anche tre tabella di ricerca (indirizzo) a sedici bit che consentono di effettuare una mappatura arbitraria limitata entro intervalli locali.

L'uscita 30 di indirizzo mappato dello Address Mapper 29 viene applicata come indirizzo a varie Mapper Memories e/o Tag RAM 31A-B e alla Error Catch RAM 1/2 (32A/B), che, anche se hanno funzioni separate, possono ciò nondimeno essere implementate con partizioni selezionabili dei quattro Memory Set che sono collettivamente la Interior Test Memory 87. L'uscita 30 di indirizzo mappato viene anche applicata come un ingresso ad un circuito 37 di Addr. Bit Select, la cui funzione di moltiplicazione viene descritta a tempo debito. La Interior Test Memory può essere configurata per contenere numerosi elementi di varie strutture di memoria a base RAM usate per diverse funzioni. Ciò si ottiene mediante dichiarazione che alcune parti dei diversi Memory Set debbono essere usate per gli scopi

associati. Ciò che è mostrato in figura 2 è una disposizione di questo genere; le disposizioni possono essere cambiate come procede il collaudo, e questa intera questione di impiego del Memory Set deve essere considerata molto dinamica. Niente di ciò che è all'interno della Interior Test Memory (ad esempio la Error Catch RAM 32A/B) sono strutture di hardware permanenti. Ciò che è permanente sono i quattro Memory Set. Ma quale parte di quale Memory Set è una Error Catch RAM ad un certo istante (anche se poi ne esiste uno definito) dipende da quale configurazione è stata stabilita.

Si considerino le Buffer Memories 31A e 31B. Le loro funzioni sono di trattenere configurazioni di dati 33 ed indirizzi 34 che possono essere applicati al DUT. Queste sono uscite effettive separate dalle loro Buffer Memories associate, sebbene queste Buffer Memories non siano memorie a doppia porta, ma siano preferibilmente costituite da parti di due diversi Memory Set. In linea con ciò, si preferisce che gli Stored Data 33 siano mantenuti in un Memory Set, mentre gli Stored Address 34 sono tenuti in un'altra. Inoltre, non è stato mostrato un meccanismo esplicito per scrivere

una Buffer Memory. Un modo che può essere effettuato è mediante una operazione a bus indirizzato avviata da un Test Site Controller 4 all'inizio del programma che viene eseguito. Vi è un bus "sotto il pavimento", come se fosse un "sistema di utenze" chiamato Ring Bus 85 che va a quasi tutto ciò che è mostrato in figura 2 (la maggior parte delle connessioni di questo non sono mostrate in quanto queste complicherebbero in modo spropositato il disegno). Un altro modo più veloce di scrivere informazione nei Memory Set è descritto in riferimento alla figura 3.

Le Error Catch RAM 32A-B sono indirizzate con lo stesso indirizzo che è applicato alle Buffer Memories, e queste immagazzinano o recuperano informazioni circa errori, le quali operazioni sono effettuate in unione con un Post Decode Circuit, che verrà discusso oltre. Come per i percorsi 33 e 34 dalle Buffer Memories 31A-B, i percorsi 62A-D (dalla Error Catch RAM1 32A) sono preferibilmente uscite multiplate da una parte di un Memory Set (configurato per operare come Error Catch RAM), secondo l'informazione di configurazione distribuita dal Ring Bus (non mostrato).

Si deve notare che il Data MUX 35 possiede

come ingressi la uscita 33 di stored data dalla Buffer Memory 31A così come i dati 28 dai registri DH e DL nella raccolta 24 di ALU. Il data MUX 35 effettua una selezione iniziale secondo i valori 36 immagazzinati nella PGM SRAM 20, di cui di questi ingressi (28, 32) per presentarli come uscita 38, che, a meno che non venga modificata come descritto oltre, viene poi applicata come uno di due componenti di vettore ad un Transmit Vector Mapper/Serializer/Receive Vector Compare Data Circuit 40 (l'altra componente è l'uscita 39 del circuito 37 di Addr. Bit Select).

Il circuito 40 può eseguire tre funzioni correlate con vettori: assemblare componenti di vettori (38, 39) in una rappresentazione logica ordinata di un intero vettore che deve essere applicato (trasmesso) al DUT; applicare una corrispondenza dinamica arbitraria (mappatura) tra i bit ordinati della rappresentazione logica del vettore di trasmissione e il numero di canale fisico effettivo della Pin Electronics (cioè, quale punta di sonda) entrerà in contatto con il DUT per conto di quel segnale (cioè, quel bit nel vettore); e, coopera con il compilatore nella divisione di un intero vettore logico in parti da applicare

separatamente ed in ordine (serializzazione) per i DUT che ammettono una tale procedura. Quale di queste funzioni viene eseguita è cosa determinata da segnali di controllo da una SRAM 41 che è anche indirizzata nella istruzione a duecentotto bit prelevata dal Micro-Controller Sequencer 19.

Inoltre, entro il circuito 40 è contenuta una parte della DUT Disable Logic 90. Il suo scopo è quello di rispondere a varie condizioni, alcune statiche, alcune contingenti sulla base dei risultati di collaudo, ma tutte definite programmaticamente, che indicano quale tra uno o più DUT tra fino a quattro di questi, debbono essere disabilitati. Queste indicazioni sono portate da quattro segnali DD 0:3 44b (DUT Disable per DUT zero, per DUT Uno, ecc.). Ciò è in supporto del collaudo a più DUT su un Test Site, ed è ulteriormente spiegato nella domanda di brevetto associata qui incorporata. L'uscita del circuito 40 è un vettore 44a fino a 64 bit che, assieme ai segnali 44b di DUT Disable, viene applicato ad un Vector FIFO 45 che quando è pieno genera il segnale VEC_FIFO_FULL 26, il cui significato ed impiego è stato discusso precedentemente. Il vettore alla testa del Vector FIFO 45 viene rimosso da questo in

seguito alla ricezione di un segnale VEC_FIFO_UNLOAD 47 che si origina in un Period Generator 49 (che verrà discusso tra breve). Tali vettori rimossi (46) sono applicati ad un circuito 52 di Timing/Formatting&Comparison che è collegato al DUT con il caso associato di Pin Electronics 9. Cioè, ciascun caso (tra i vari Test Site) della Pin Electronics 9 riceve Transmitted and Received Vectors 7 e informazione 8 di configurazione di Pin Electronics dal suo circuito 52 associato di Timing/Formatting&Comparison.

Il circuito 52 di Timing/Formatting&Comparison è accoppiato al VT Bus 89 per ricevere informazione di configurazione e controllo. Si ricorderà che il circuito 52 di Timing/Formatting&Comparison è in effetti costituito da otto circuiti integrati, che per i presenti scopi vengono trattati come una singola entità.

Il circuito 52 di Timing/Formatting & Comparison, possiede una Internal SRAM 54 indirizzata dallo stesso Instruction Address ("A" nel piccolo cerchio) come lo è la Programma SRAM 20 del Micro-Controller Sequencer 19. (Una External DRAM 53 può essere impiegata in luogo della Internal SRAM 54, ma è indirizzata localmente da un

contatore incrementato che non è mostrato). La Internal SRAM 54 (o DRAM esterna 53) assiste la produzione dei cicli di Drive e Comparison, che hanno formati associati. I cicli di Drive applicano un vettore di trasmissione al DUT impiegando un formato preselezionato fornito da una delle RAM 54 o 53. I cicli di Comparison ricevono un vettore presentato dalla DUT e lo esaminano, anche secondo un formato preselezionato fornito dalla RAM, per determinare se questo coincide con dati di confronto forniti in precedenza. Ambedue i cicli di Drive e di Comparison sono aggiustabili per quanto riguarda la loro durata, e appropriatamente aggiustabili per quanto riguarda se e quando viene applicato un carico, quando i dati sono agganciati o sottoposti a Strobe, se un segnale è un Return-to-Zero oppure no, se associano un segnale pilotato con il suo complemento, ecc. (queste opzioni sono vari formati menzionati precedentemente).

Il confronto prodotto dal circuito 52 di Timing/Formatting&Comparison comprende informazione, su una base per canale, sul fatto che un canale che ha presentato avaria per il fatto che un valore logico era erroneo (un errore funzionale) e/o per il fatto che le sue proprietà elettriche

sono fuori di limiti accettabili (un errore parametrico). Inoltre, è come è spiegato in una domanda di brevetto qui incorporata, quando viene effettuato un collaudo su più DUT è noto quali canali sono associati con quale DUT. Ciò consente la produzione dei quattro segnali DFE 0:3 (DUT # Functional Error) 103 e i quattro segnali DPE 0:3 (DUT # Parametric Error) 104.

Il confronto effettuato dal circuito 52 di Timing/Formatting&Comparison produce anche un valore 56 a sessantaquattro bit che viene applicato al Receive Vector Reverse Mapper/Deserializer 57, la cui funzione può essere considerata l'inverso logico del circuito 40. (Il funzionamento del circuito 57 è controllato da una SRAM 58 che corrisponde al controllo del circuito 40 dalla SRAM 41). A sua volta l'uscita 59 del circuito 57 è applicata al circuito 60 di Post Decode, ed anche alla Error Catch RAM1 32A. Al momento, è sufficiente dire che il circuito 60 di Post Decode può essere ispezionato attraverso criteri programmatici sia informazione 59 di errore in arrivo ed informazione di errore precedentemente memorizzata nella Error Catch RAM1 32A per produrre informazione di errore condensata e più facilmente

interpretabile che può successivamente essere immagazzinata di nuovo nella Error Catch RAM2 32B attraverso il percorso 61. Un esempio sarebbe creare un conteggio di quante volte si è presentato un errore entro un particolare campo di indirizzi informazione che può essere utile nel decidere quando tentare di impegnarsi in una riparazione sul chip con l'abilitazione di circuiti sostitutivi.

Si passa ora al Period Generator 49 e alla associata Timing SRAM 51. Questi rispondono ad un segnale ad otto bit T_SEL 43 che, per una istruzione a duecentotto bit prelevata dal Micro-Controller Sequencer 19, determina una durata per il funzionamento associato del circuito 52 di Timing/Formatting&Comparison. T_SEL 43 è un elemento dei Various Control Values & Instructions 42 che sono rappresentate da diversi campi entro l'istruzione prelevata. Come valore ad otto bit questo può rappresentare o codificare duecentocinquantasei diverse cose. In questo caso, queste "cose" sono valori a ventotto bit immagazzinati nella Timing SRAM 51 e che sono indirizzati da T_SEL. Ciascun valore a ventotto bit (23) specifica una durata desiderata con una risoluzione di 19,5 picosecondi. La sequenza dei

valori (23) di durata a ventotto bit cui è stato fatto accesso è immagazzinata in un Period FIFO 50 in modo che i singoli elementi di quella sequenza verranno ritrovati ed applicati in sincronismo con il recupero del loro vettore corrispondente inteso, che è immagazzinato nel Vector FIFO 45.

Un campo di valore di temporizzazione grossolano nella più vecchia immissione nel FIFO 50 trasporta informazioni di durata con una risoluzione di 5 nsec, e produce da questi un segnale VEC_FIFO_UNLOAD 47 che trasferisce il prossimo vettore di trasmissione dal Vector FIFO 45 al circuito 52 di Timing/Formatting&Comparison. Un segnale associato TIMING REMAINDER 48 viene anche applicato ad un circuito 52. È a questo punto che viene ottenuta la risoluzione finale a 19,5 picosecondi.

Si faccia riferimento ora alla figura 3, che è uno schema a blocchi semplificato 64 della Interior Test Memory 87 nello schema a blocchi di figura 2. Questa riceve un indirizzo 30 mappato a 48 bit dallo Address Mapper 29, che viene applicato ai vari Address Classifier 77, 78 e 79. Gli Address Classifier sono associati con i Memory Set 73-76, che sono ciascuno un meccanismo completo di memoria

che può effettuare individualmente varie funzioni, come essere un ECR 32. Due di questi Memory Set (73, 74) sono di DRAM esterna, mentre due sono di SRAM interna. I due DRAM Memory Set esterni avranno sempre la medesima funzione di Address Classifier in effetti, e quindi condividono un Address Classifier 77 in comune. I SRAM Memory Set 75, 76 hanno rispettivamente i loro propri Address Classifier associati 78 e 79. Questi Address Classifier possono o far passare un indirizzo attraverso non modificato o modificarlo in modi che verranno descritti in un qualche dettaglio in una delle domande di brevetto incorporate.

Ciascun Memory Set comprende un Memory Set Controller; i DRAM Memory Set esterni 73 e 74 possiedono rispettivamente DRAM Memory Set Controller 65 e 66, mentre i SRAM Memory Set interni 75 e 76 hanno rispettivi SRAM Memory Set Controller 67 e 68. Durante il collaudo di una DUT l'indirizzo per le transazioni di memoria indirizzate a ciascuno di questi Memory Set perviene al Memory Set Controller associato dallo Address Classifier rispettivamente associato. Tutti i quattro Memory Set possono ricevere dati direttamente sul percorso 59, che è essenzialmente

il percorso di tecnica anteriore entro la memoria interna di collaudo (che aggira il Post Decode 60). Durante il collaudo di una DUT Error Data 61 che perviene dal circuito 60 di Posto Decode e che deve essere scritto nella Error Catch RAM2 32B viene dapprima applicato ai Data Classifier 82-83, che sono rispettivamente associati con i Memory Set 2 e 3. I Data Classifier 80-83 possono o no cambiare i dati applicati ad essi, a seconda di come questi sono configurati e delle funzioni che debbono eseguire. Gli Address and Data Classifier rappresentano percorsi ad alta velocità per indirizzi e dati, rispettivamente, che sono intesi operare alle più elevate velocità necessarie. In breve verrà visto che il Ring Bus (non ancora mostrato) fornisce un altro modo per convogliare indirizzi e dati ai Memory Set.

A questo punto si hanno quattro Memory Set Controller (65-68) che hanno ciascuno indirizzi in arrivo (classificati) e dati (classificati). Ciascuno di questi Memory Set Controller è accoppiato ad una memoria associata: DRAM Memory Set Controller 73 e 74 sono rispettivamente accoppiati a DRAM esterne 69 e 70, mentre i SRAM Memory Set Controller 75 e 76 sono rispettivamente

accoppiati a SRAM interne 71 e 72. Queste disposizioni costituiscono i quattro Memory Set 73-76, due dei quali (75, 76) hanno piccole quantità di SRAM ad alta velocità, e due dei quali (73, 74) hanno grandi quantità di DRAM più lente. Ciò che è di interesse a questo momento è di come i DRAM Memory Set possono essere resi altrettanto veloci quanto gli SRAM Memory Set, così come incorporare alternative riguardanti la configurazione della DRAM, a seconda della preferenza dell'utilizzatore e dalla strategia di programma di collaudo. Quindi, si presenta il fatto che i DRAM Memory Set Controller 65 e 66 sono configurabili, effettuano diversi tipi di transazioni di memoria, e non sono tuttavia gli stessi dei più semplici SRAM Memory Set Controller 67 e 68. Per brevità, la figura 3 non mostra la struttura che fornisce questa flessibilità; per il momento si può dire soltanto che ciascun Memory Set Controller è collegato al Ring Bus (non ancora mostrato), dal quale viene istruito nel modo particolare di funzionamento e configurazione che sono desiderati. Alcuni di questi modi comportano come i dati vengono memorizzati, e alcuni hanno a che fare con il recuperarli indietro di nuovo. Per concludere,

quindi, si deve notare che ciascun Memory Set possiede un Data Out (62A-D) associato che viene inviato al Post Decode Mechanism 60 per ulteriore elaborazione. Si deve notare anche che le uscite di dati dai Memory Set zero e due sono applicate a MUX 84 l'uscita del quale diviene STORED DATA 33 che sono inviati a data MUX 35. Similmente, le uscite dei dati dai Memory Set uno e tre sono applicate ad un MUX 127 la cui uscita diviene STORED ADDRESSES che sono inviati allo Addr. Bit Select MUX 37. La ragione per avere i MUX 84 e 86, ed i dettagli di come questi sono controllati non sono di particolare interesse in questa sede; questi sono discussi nelle domande di brevetto incorporate.

Ora si considerino le figure 4A-B, che sono uno schema a blocchi semplificato 91 del circuito 60 di Post Decode di figura 2. Iniziando con la figura 4A, ed in modo congruente con quanto mostrato in figura 2, vi sono due percorsi di dati che servono come ingressi nel meccanismo 60 di Post Decode. Questi sono la singola sorgente 59, dal Receive Vector Reverse Mapper/Deserializer (57), e le quattro sorgenti 62A-D dalla memoria interna di collaudo 87 per un Memory Set configurato per funzionare come ECR (che sono stati illustrati come

Error Catch RAM1 32A, oppure ECR1. Di queste quattro sorgenti 62A-D, soltanto una (attiva) ECR sarà presente in un qualsiasi singolo Memory Set ad un dato momento. Tutte le cinque sorgenti sono applicate al MUX 94, che seleziona una di queste come ingresso da applicare a varie Input Mask (96, 97, 105, 107) e ad un MUX 108 che seleziona tra l'uscita del MUX 94 e le uscite di queste varie Input Mask.

Si impiega il termine "mask" (maschera) nel suo modo convenzionale, per indicare una configurazione ed un meccanismo associato per rimuovere o includere bit selezionati in un campo di dati sulla base di un qualche tipo di considerazione. La configurazione di maschera ha lo stesso numero di bit del campo di dati di interesse, ed è considerata essere in corrispondenza uno a uno con questo. Nel nostro caso impieghiamo una maschera che ha uni in collocazioni che sono da ignorare o da "mascherare fuori" dalla considerazione. Il campo dati seguirà la usuale convenzione di rappresentare le avarie con zeri. Le "considerazioni" di interesse generalmente saranno la presenza di un qualche, o il numero di zeri nel campo dati rimanente dopo che

è stata applicata la maschera. La circuiteria che effettua l'effettivo mascheramento verrà descritta oltre. Per il momento, verrà trattata semplicemente l'operazione di mascheratura sul livello di schema a blocchi mentre consideriamo la natura generica delle figure 4A-B. La proprietà di schema a blocchi di una Input Mask consiste nel fatto che posizionerà una posizione di bit mascherato a uno, in modo che questo non contribuirà a rappresentare una qualche avaria.

Per continuare quindi, si consideri il Jam Mask Register 93, e si noti che questo è accoppiato con il Ring Bus 85 allo scopo di ricevere un valore di maschera a 32 bit. Tale valore di maschera viene applicato alla Input Mask 96, mentre il suo complemento bit per bit è applicato alla Input Mask 97 con l'azione di invertitore 98. Queste maschere sono costanti per la durata di una data configurazione. Cioè, non ci aspetta che queste cambino in modo di routine da un valore di dati al successivo. Le Input Mask esse stesse ricevono come dati di ingresso la sorgente selezionata dal MUX 94, secondo il segnale ERR_DATA_SEL, che a sua volta si origina con un PD Control Register 92 anche esso collegato al Ring Bus 85. Le uscite

delle Input Mask 96 e 97 sono poi applicate come ingressi aggiuntivi a MUX 108.

MUX 108 riceve due ingressi mascherati in più. Queste sono le uscite di Input Mask 105 e 107, i cui ingressi sono anche uscite del MUX 94, e le cui maschere sono anche complementari per effetto degli invertitori 106. In questo caso, tuttavia, la sorgente del valore di maschera è prevista essere una buffer Memory, come selezionata da MUX 95 e il segnale di controllo di MUX BM_SEL. A questo scopo, si deve notare che MUX 95 ha come ingressi le quattro sorgenti 62A-D. Questa disposizione consente che le maschere impiegate dalle Input Mask 105 e 107 cambino su base ciclo per ciclo mentre marcia una configurazione.

Si comprenderà ora che ciò che è presentato come ingressi a MUX 108 sono quattro versioni variamente mascherate dei dati indirizzati o una versione non mascherata di quei dati. Quali di queste versioni sia selezionata da MUX 108 ed inviata come dati 114 mascherati viene determinato dal segnale di controllo di MUX PD_MASK_MODE.

I tre segnali di controllo di MUX menzionati finora (ERROR_DAT_SEL, BM_SEL, PD_MASK_MODE), così come uno aggiuntivo da introdurre successivamente

(PD_ERR_SEL), sono tutti prodotti come uscite del PD Control Register 92, a seconda di come il suo contenuto è impostato dal traffico indirizzato ad esso sul Ring Bus 85.

Si consideri ora la figura 4B. Questo disegno piuttosto affollato non è terribile come sembra. Si consideri il quadrante superiore di sinistra della figura. In quel punto sono rappresentati trentadue casi di Analysis Mask e meccanismi di Analysis Counter. Si consideri lo scopo di tutte queste parti.

Per iniziare, le Analysis Mask (ANYL. MASK_0 ... ANYL. MASK_31, 110a-z) hanno a livello di schema a blocchi la proprietà che questi producono una uscita a singolo bit che indica con uno zero che uno più dei bit di dati non mascherati era uno zero, e un uno se non vi erano indicazioni di avaria (zeri) tra i bit non mascherati. La maschera impiegata per ciascun circuito di maschera 110a-z viene fornita da un MASK_0 ... MASK_31 Register (109a-z) che viene accoppiato al Ring Bus per ricevere il suo valore. Ora si devono notare due cose. Dapprima, l'uscita dei circuiti di Analysis Mask 110a-z sono inviati a corrispondenti Analysis Counter (ANYL. CNTR_0 ... ANYL. CNTR_31, 112a-z).

Questi contatori contano gli zeri inviati ad essi, e possono essere usati per un totale fino al numero di volte (ad esempio durante una configurazione) in cui viene rivelato un errore dalla maschera associata. In secondo luogo, tutte le trentadue uscite mascherate sono raccolte in un insieme 113 di queste, in modo che queste possono essere o memorizzate in un ECR (attraverso il percorso 61) o ulteriormente analizzati da un PD Error Mask 120.

Per continuare, gli Analysis Counter 112a-z hanno le seguenti proprietà a livello di schema a blocchi. Questi conteggiano a scendere da un valore pre-caricato inserito in essi mediante ciò che è memorizzato in corrispondenti LOAD REGISTERS (CNTR_0 LOAD REG. ... CNTR._31 LOAD REG., 111a-z), che a loro volta sono tutti impostati separatamente da traffico individualmente indirizzato ad essi sul Ring Bus. Ciò consente ai contatori di essere pre-impostati con un valore di soglia, dal quale questi conteggiano a scendere. Quando questi raggiungono zero un corrispondente Terminal Count Flag (TCF_0 ... TCF_31, 115a-z) esplicita uno zero, e viene inibito un ulteriore conteggio a scendere (i contatori rimangono bloccati su zero). Questi rimarranno a zero fin quando questi sono caricati

con un nuovo valore di soglia per l'azione di un segnale LOAD_COUNTERS, che forzerà un ricarico ogni volta che questo viene attivato. Vi è anche un segnale QUALIFIED_LOAD che ricaricherà ciascuno dei contatori 112a-z dai rispettivi registri 111a-z, posto che un contatore non sia già zero, nel quale caso viene inibito un ricarico individuale.

Si deve notare che i vari Terminal Count Flag 115a-z sono raccolti come 116 e applicati come ingressi alternativi a MUX 118, assieme alla raccolta 113 dalle Analysis Mask 110a-z. Le raccolte 116 e 113 sono anche applicate ad un MUX 117 dal quale quella prescelta di queste può essere immagazzinata in una Buffer Memory attraverso il percorso 61. La selezione per sia il MUX 117 sia il MUX 118 viene effettuata dal segnale di controllo PD_ERR_SELL, che si origina dal PD Control Register 92.

Per continuare con il resto di figura 4B, si comprenderà che la Analysis Mask e il meccanismo Analysis Counter appena descritti consentono che possa essere rivelata e contata un'ampia varietà di condizioni di errore, con indicazioni prodotte circa i conteggi che superano soglie prestabilite. Lo stato di cose che si verifica sulle uscite 113 e

116 può essere immagazzinato in memoria attraverso MUX 117 e percorso 61. Lo stesso stato di cose è anche sottoposto ad ulteriore analisi attraverso MUX 118 e PD Error Mask 120, che è una maschera di tipo Input la cui maschera è contenuta nel Error Mask Register 119, come impostato dal Ring Bus. L'uscita 121 a trentadue bit della PD Error Mask 120 è applicata alla porta NAND 122, dove è essenzialmente posta in OR con altri due segnali "zero è errore": le uscite delle porte 129 e 136. Questi ultimi segnali verranno discussi a tempo debito. Il segnale di OR così prodotto viene reso come uno significa esplicitato di segnale PD_ERR 90 che viene applicato al Micro-Controller Sequencer 119.

I dati mascherati 114 sono anche applicati ad altre due strutture maschera/contatore mostrate nella parte inferiore di figura 4B. Si consideri il ACCUM. ANYL. MASK 123 (la cui maschera è contenuta da ACCUM. MASK REG. 124, come impostato dal Ring Bus). Questo funziona nello stesso modo come qualsiasi delle maschere 110a-z. La sua uscita è applicata ad un ACCUM.CNTR. 126 che anche conteggia a scendere quando questo viene alimentato con uno zero dalla maschera 123, e viene pre-impostato ad

un valore di soglia da ACCUM. LOAD REG. 125 quando viene attivato il segnale LOAD_ACCUM_CNTR. Il valore di soglia nel registro 125 è fornito dal Ring Bus. L'uscita del contatore 126 può contare a scendere fino a zero, in seguito della quale cosa fa in modo che il Terminal Count Flag TCF_ACCUM espliciti uno zero ed il contatore stesso rimanga bloccato a zero fino al ripristino. A questo punto la struttura degli elementi 123-127 è essenzialmente identica a quanto è stato già descritto in riferimento ad Analysis Mask e Analysis Counter. La differenza consiste in come i risultati possono essere posti in uscita, e in cosa si intende che siano le maschere. Gli elementi 123-127 sono intesi per accumulare il numero di parole intere che possiedono errori, e confrontarli contro la soglia desiderata. Per continuare, il segnale TCF_ACCUM da 127 viene applicato come maschera ad un bit costituita da Latch ad un bit A_LATCH 128 e porta OR 129. L'uscita della porta OR 129 indica, con una logica di zero-significa-errore rivelato, se la soglia è stata incontrata, ed è un altro ingresso nella produzione di PD_ERR 90.

Vi è più di una maschera e meccanismo di accumulazione in figura 4B. BITWISE ANYL. MASK 130,

BITWISE MASK REG. 131 e BITWISE LOAD REG. 132 sono identici alle loro controparti precedentemente spiegate (ad esempio 123-125). La sola differenza a questo punto consiste nel fatto che il segnale di carico per il registro 132 è il suo proprio segnale LOAD_BITWISE_ACCUM. Sorge ora una differenza nelle proprietà dello schema a blocchi, in quanto il Latch BITWISE ACCUM. 133 non è un semplice contatore come gli altri. Questo conteggia il numero di zeri che lo raggiunge entro ciascun risultato di maschera dopo l'operazione di mascheramento, e accumula questi come quantità a decrementare contro una soglia iniziale nel registro 132. Quando la soglia è ridotta a zero il flag TCF_BITW del Latch 134 viene attivato basso. La similarità viene ancora una volta ripresa con una maschera ad un bit costituita da B_LATCH 135 e porta OR 136, la cui uscita è il contributore finale alla produzione di PD_ERR 90 da parte della porta NAND 122.

Si può concludere la presente panoramica del meccanismo 60 di Post Decode con una breve discussione della funzione del 4-bit Write Only Register 137, collocato nell'angolo superiore di destra della figura 4B. Questo viene posizionato

dal Ring Bus, e quando uno dei suoi bit di uscita viene controllato a clock come TRUE (un uno) questo ovviamente attiva se stesso ed il segnale di controllo con il nome associato. Questi sono stati già incontrati, e questi sono i vari segnali di LOAD impiegati dai contatori che abbiamo descritto precedentemente. L'idea qui è di fissarli in modo che questi segnali di LOAD facciano il loro lavoro e quindi scompaiano, in quanto sarebbe estremamente dannoso per il funzionamento che questi persistano. A tale scopo, la porta OR 130 rivela che uno o più di questi segnali di LOAD è vero, e fornisce questo ad una porta AND 139 che, quando si presenta il successivo ciclo di clock, ripristina i contenuti del registro 137 a tutti zeri. È un meccanismo del tipo "monostabile".

Infine, e sebbene ciò non sia stato ancora mostrato (questo si trova in figura 7), i valori nei vari contatori possono essere letti in estrazione sul Ring Bus.

Si faccia ora riferimento alla figura 5, in cui viene mostrato uno schema a blocchi 165 che descrive i modi in cui lavorano le varie maschere. Una Input Mask è costituita da un valore 166 di dati di ingresso a trentadue bit ed un valore 167

di maschera a trentadue bit. Il valore di maschera 167 potrebbe provenire da un registro (eventualmente statico) oppure potrebbe essere più transitorio, quali la lettura di dati di maschera dalla memoria. In ogni caso, vi è una operazione di OR bit per bit da parte delle porte 168a-z tra i due elementi 166 e 167. La quantità 169 risultante a trentadue bit è l'uscita da una Input Mask. Per una Analysis Mask, la quantità 169 è applicata alla porta AND 170 che produce uno zero sulla sua uscita se un qualsiasi bit della quantità 169 è uno zero.

La natura del meccanismo 133 BITWISE ACCUMULATION può essere compresa facendo riferimento alla figura 6, che è uno schema a blocchi 140 di tale circuiteria. Questo contiene un registro di maschera 141 il cui valore 142 a trentadue bit è posto in NOR bit per bit con un valore 143 di dati di ingresso a trentadue bit. La differenza finora con la disposizione di maschera di figura 5 consiste nel fatto che le porte 144a-z sono porte NOR invece di porte OR. L'effetto di questo cambiamento è di produrre uni invece di zeri per avarie che sopravvivono al mascheramento. Ciò è per fare in modo che il verificarsi di indicazioni di avaria possa essere sommato più facilmente in un

totale. Queste sono sommate raggruppando porte NOR in coppie e applicando le uscite di queste coppie di porte NOR a sommatore 145a-o. I sommatore 145a-o sono essi stessi appaiati e le loro uscite sono applicate ad ancora altri addizionatori 146a-h. In questo modo il numero complessivo di zeri nel valore 143 di ingresso viene accumulato pezzo per pezzo, fin quando questo è disponibile sull'uscita dell'addizionatore 147. Ora questo viene usato per decrementare un valore che è iniziato come valore di soglia in BITWISE LOAD REGISTER 132, che per un'attivazione iniziale del segnale di controllo di MUX LOAD_BITWISE_ACCUM sul MUX 150, è diventato BITWISE ACCUM. REGISTER 151. Ora il tutto viene disposto in modo che il valore nel registro 151 venga decrementato dall'uscita dell'addizionatore 147. Ciò è ottenuto dal circuito 138 di sottrazione, la cui uscita è quindi indirizzata attraverso MUX 150 per diventare il valore memorizzato successivo nel registro 151. L'effetto è di far contare a scendere il registro 151 con le somme prodotte dall'addizionatore 147. Quando il registro 151 passa per lo zero il meccanismo 134 di Terminal Count Flag esplicita uno zero.

La figura 7 mostra uno schema a blocchi 154

che illustra la natura degli Analysis and Accumulation Counters 156 descritti precedentemente. Gli Analysis Counters e i loro Load Registers (157) sono da ventiquattro bit mentre quegli elementi per la Accumulation impiegano trentadue bit. In ogni caso, uno zero che deve essere contato dalla maschera viene applicato alla porta NOR 158, il cui altro ingresso è un segnale YES_ZERO 164. YES_ZERO viene prodotto da una porta NOR 159 pilotata dalle uscite (vero) del contatore 156 e questo compito deve produrre YES_ZERO = ONE ogni volta che il contenuto del contatore 156 è zero. YES_ZERO svolge due compiti. In primo luogo, questo inibisce un ulteriore conteggio forzando l'uscita della porta NOR 150 a zero quale che sia l'ingresso dalla maschera. Il contatore 156 in effetti conteggia uni, per cui non conteggia più. In secondo luogo, YES_ZERO posiziona TCF Latch 160, la cui uscita NOT-Q va quindi a zero.

Per un Analysis Counter, il segnale TCF che va a zero opera attraverso la porta AND 161 per inibire l'azione del segnale QUALIFIED_LOAD. Ciò previene il ricaricamento del registro 156 dal verificarsi, come menzionato precedentemente. Se

QUALIFIED_LOAD non viene inibito, questo passa attraverso la porta OR 162 dove QUALIFIED_LOAD oppure LOAD_COUNTERS causeranno un LOAD del contatore 156 con il valore nel registro 157 ed anche azzererà il PCF Latch 160. Ciò è indicato dalla circuiteria entro la linea a tratteggio 171.

Per lo Accumulation Counter, i componenti all'interno della linea tratteggiata 171 sono sostituiti dalla semplice disposizione mostrata con la linea tratteggiata 172. In tale caso il segnale LOAD_ACCUM_CNTR viene applicato direttamente al LOAD per il contatore 156 e RESET per il TCF Latch 160.

Si deve notare che l'uscita del contatore 56 è applicata ad un Bus Interface 155, dalla quale il suo valore può essere inviato sopra il Ring Bus.

Varie caratteristiche desiderabili per un meccanismo di Post Decode sono state esposte nel sommario. In questa sede ora, vi è una breve spiegazione di come l'argomento delle figure 4A-B, 5, 6 e 7 possa essere impiegato per ottenere questi scopi desiderabili. La caratteristica di "analisi al volo" del meccanismo incrementato di Post Decode viene effettuata dal percorso 59 che è collegato al MUX 94. Ciò consente che l'analisi proceda senza

dover precedentemente memorizzare risultati di collaudo nella memoria di collaudo interna. La capacità di effettuare un'analisi al volo, accoppiata al fatto di essere in grado di utilizzare il contenuto di una buffer Memory come maschera che può cambiare da indirizzo ad indirizzo applicati, si combina per facilitare una caratteristica che consente che soltanto nuovi errori vengono riconosciuti. Nel sommario ciò era stato definito come "Incremental Post Decode Analysis" (analisi di Post Decode incrementale). L'idea è quella di far crescere dati di errore in un ECR durante una prova iniziale, copiare tali dati di errore in una buffer Memory, e quindi usare tale buffer Memory come maschera per oscurare vecchi errori durante un nuovo collaudo, consentendo che soltanto nuovi errori vengano riconosciuti ed ulteriormente elaborati. Questa caratteristica è supportata dall'accoppiamento dei percorsi 62A-D fino a MUX 95. Questa è anche supportata dagli invertitori 106 accoppiati a Input Mask 105, in quanto questi invertitori consentono che il formato per cui uno zero che indicano un'avaria vengano invertiti in uno logico che indicheranno posizioni di bit che debbono essere

oscurate dalla maschera. Durante una prima prova iniziale, tuttavia, è desiderabile impiegare una maschera di tutti zeri nel Jam Mask Register 93 in unione con una Input Mask 96 e con MUX 94 al fine di consentire la cattura non mascherata per la prima passata della prova. Dopo ciò, può iniziare il modo di funzionamento incrementale impiegando MUX 95 in luogo di MUX 94.

Il meccanismo di analisi incrementale descritto precedentemente si basa sul fatto di essere in grado di copiare da una ECR ad una Buffer Memory. Il sistema di memoria che è la memoria interna di collaudo non fornisce, di per sé, mezzi per effettuare una tale operazione di copia. Questa può essere fatta, tuttavia, con l'aiuto del meccanismo incrementato di Post Decode descritto precedentemente. In particolare, vi è un percorso per la trasmissione di dati non alterati che inizia con uno qualsiasi dei percorsi 62A-D e termina con il percorso 61. Tale percorso non alterato inizia con MUX 94, quindi con MUX, procede attraverso una corretta configurazione di Analysis Mask 110a-z, e termina lungo il percorso 113 verso MUX 117. La corretta configurazione della Analysis Mask 110a-z è per ciascuno avere una maschera di tutti uni, ad

eccezione della posizione di bit corrispondente al numero di maschera, che sarà uno zero. (Questa condizione apparentemente strana segue dal fatto che ciascuna Analysis Mask pone in uscita soltanto un singolo bit, e predispone che la molteplicità di queste maschere faccia passare i dati senza modificarli).

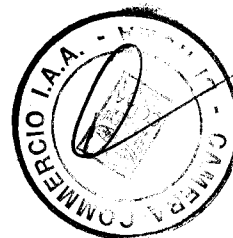
Il meccanismo 60 di Post Decode incrementato, in alcuni casi effettua analisi simultanee. Ad esempio, la raccolta di Analysis Mask 110a-z può effettuare il suo compito o compiti allo stesso tempo in cui ciascun meccanismo di Accumulation Analysis (123-129) ed il meccanismo Bitwise Analysis (139-136) effettuano indipendentemente i loro compiti. Ciò che rende ciò possibile è che ciascuno di questi meccanismi è indipendentemente e simultaneamente esposto ai dati 114.

Si comprenderà che il meccanismo Accumulation Analysis (123-129) opera sulla presenza o assenza indifferenziata di errori entro una parola di dati che si presenta sul percorso 114. Cioè, l'esistenza di uno o più errori entro una parola di dati viene trattata allo stesso modo di un singolo errore. In contrasto, il meccanismo Bitwise Analysis (130-136) risponde al numero effettivo di errori entro una

parola di dati che si presenta sul percorso 114.

Infine, si deve notare che i valori raccolti per i Terminal Count Flag (116) sono applicati al MUX 117, dai quali questi possono essere registrati verso una Buffer Memory attraverso il percorso 61. In questo è un esempio di perché questa caratteristica può essere utile. Si supponga che un ECR sia stato configurato per operare come uno Z (cioè blocco) TAG RAM. È desiderabile avere soglie singole per errori associati con ciascun blocco. Per effettuare ciò, si registrerebbe il Terminal Count Flag sulla ECR per ciascun indirizzo di blocco e quindi si azzerano gli Analysis Counter tra blocchi. L'informazione registrata indica i blocchi che superano le loro soglie.

(Iscr. Albo n. 83 BM)



RM 2001 A 000644

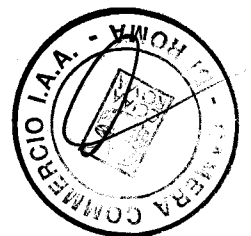
RIVENDICAZIONI

1. Metodo per analizzare risultati di prova per una memoria sotto collaudo (14) su un'apparecchiatura di collaudo di memoria (1), il metodo comprendendo i passi di:

- (a) effettuare un'operazione di collaudo su una memoria sotto collaudo per produrre un risultato di dati (56);
- (b) in assenza di una precedente memorizzazione di tale risultato di dati in una collocazione di lavoro, applicare una maschera (96, 97, 105, 110, 120, 167, 170) al risultato di dati, e quindi;
- (c) attivare un contatore (112) secondo indicazioni di errore rimanenti nel risultato di dati dopo l'applicazione della maschera.

2. Metodo come nella rivendicazione 1, ulteriormente comprendente i passi di confrontare (148/156) il contenuto del contatore contro un valore di soglia (151/157) e generare un segnale di errore quando la soglia viene incontrata.

p.p. Agilent Technologies, Inc.



Stampa n. 00 0001

A handwritten signature in black ink.

1/8
RM 2001 A 000644

FINO A TRENTADUE SITI DI COLLAUDO (TEST)

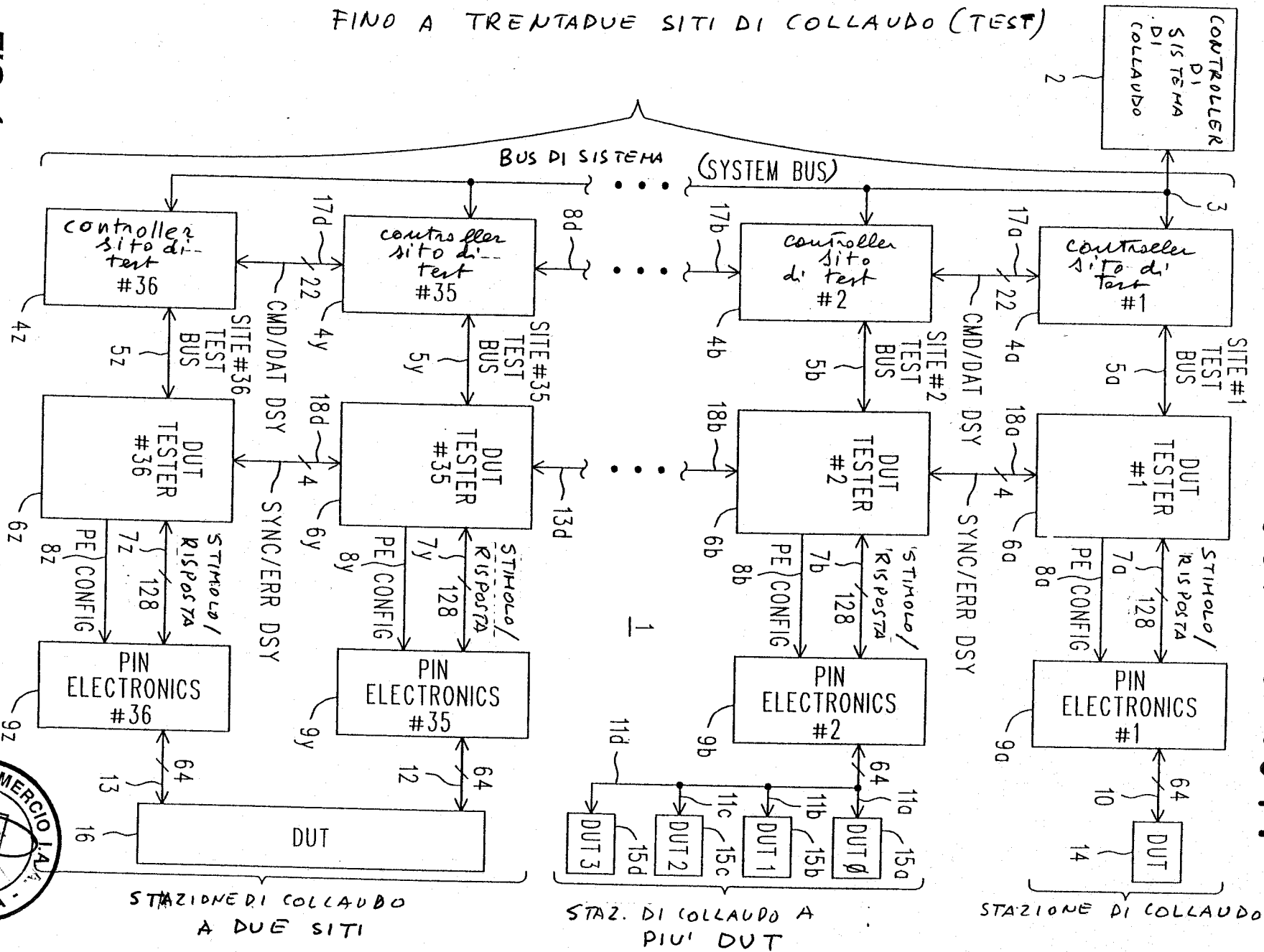


FIG. 1

Monza



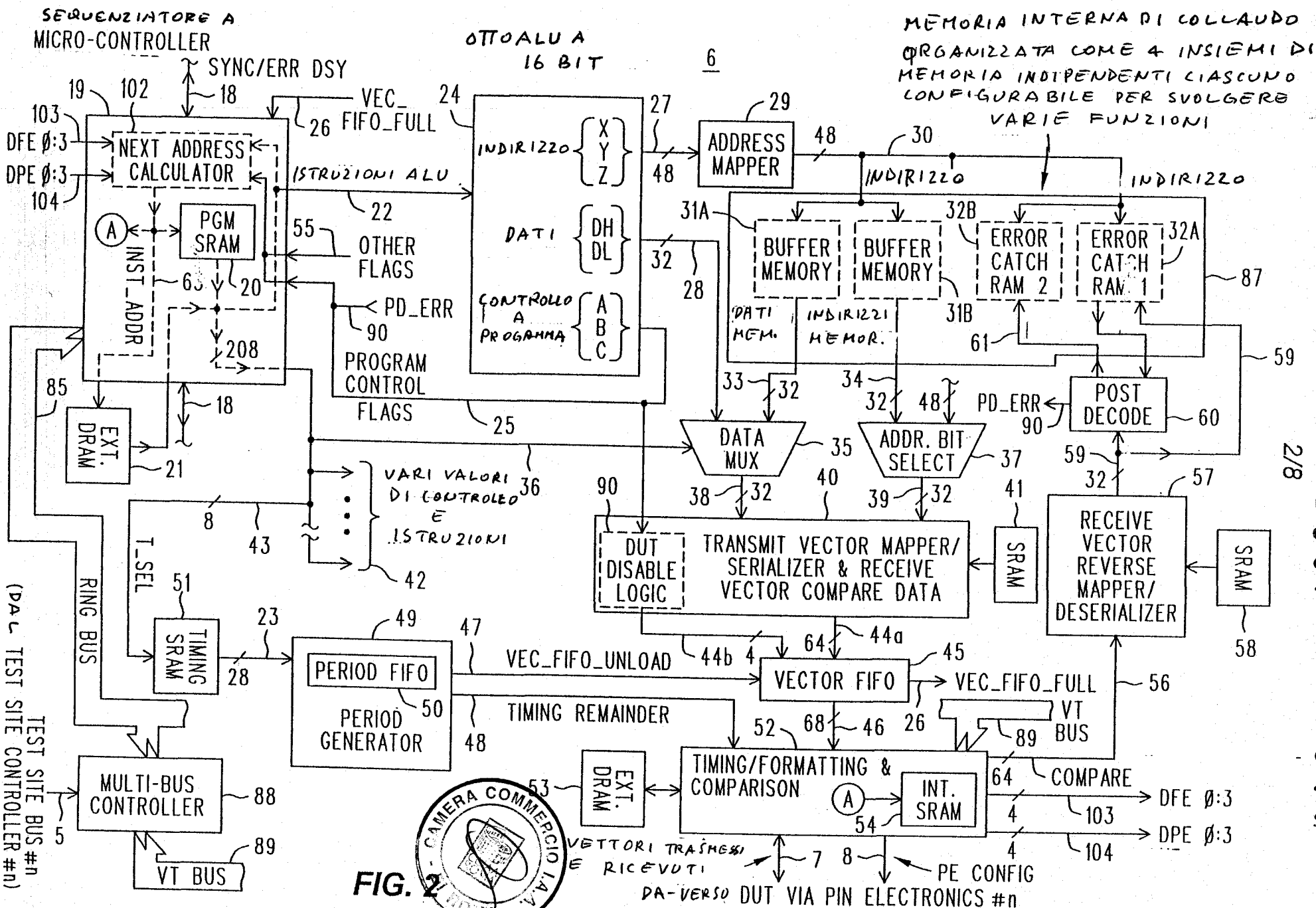
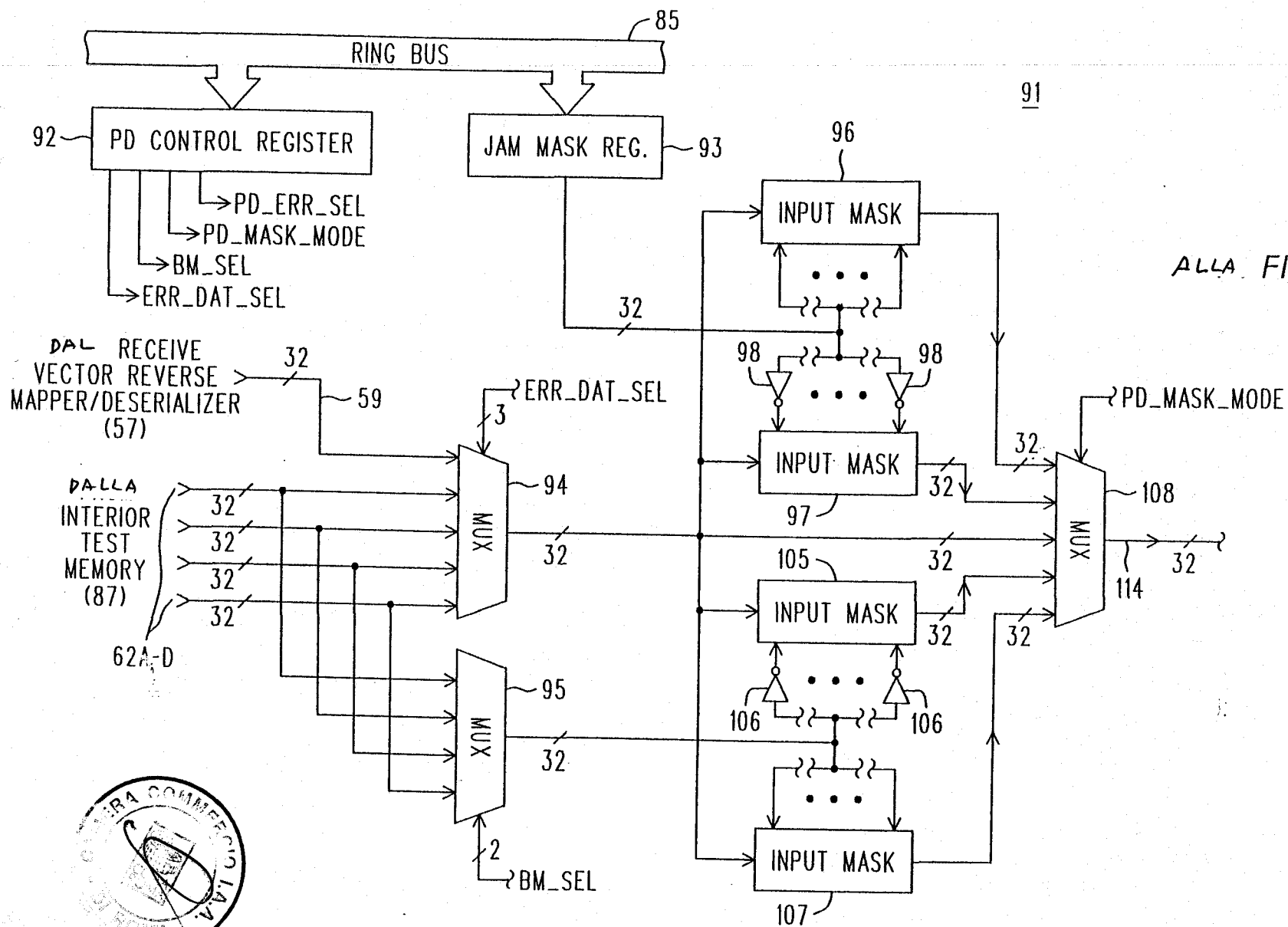
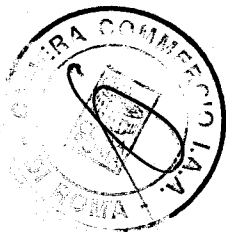




FIG. 3

[Handwritten signature]



ALLA FIG. 4B
→

FIG. 4A

Adam



4B

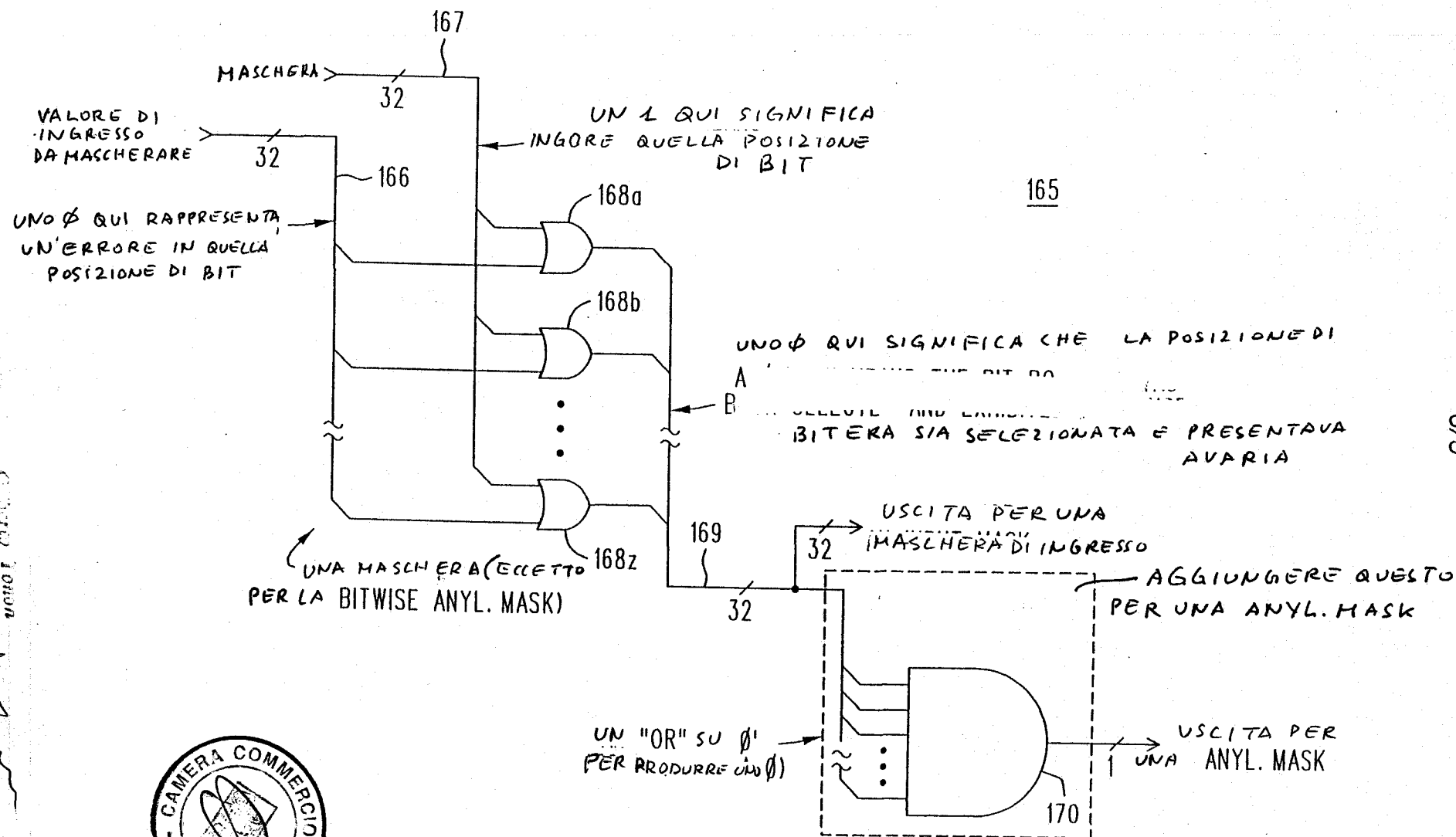
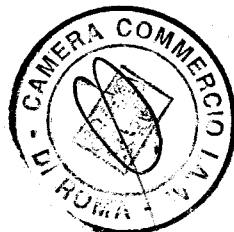


FIG. 5

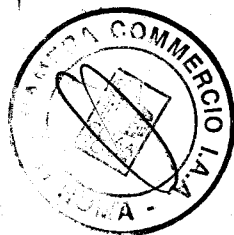
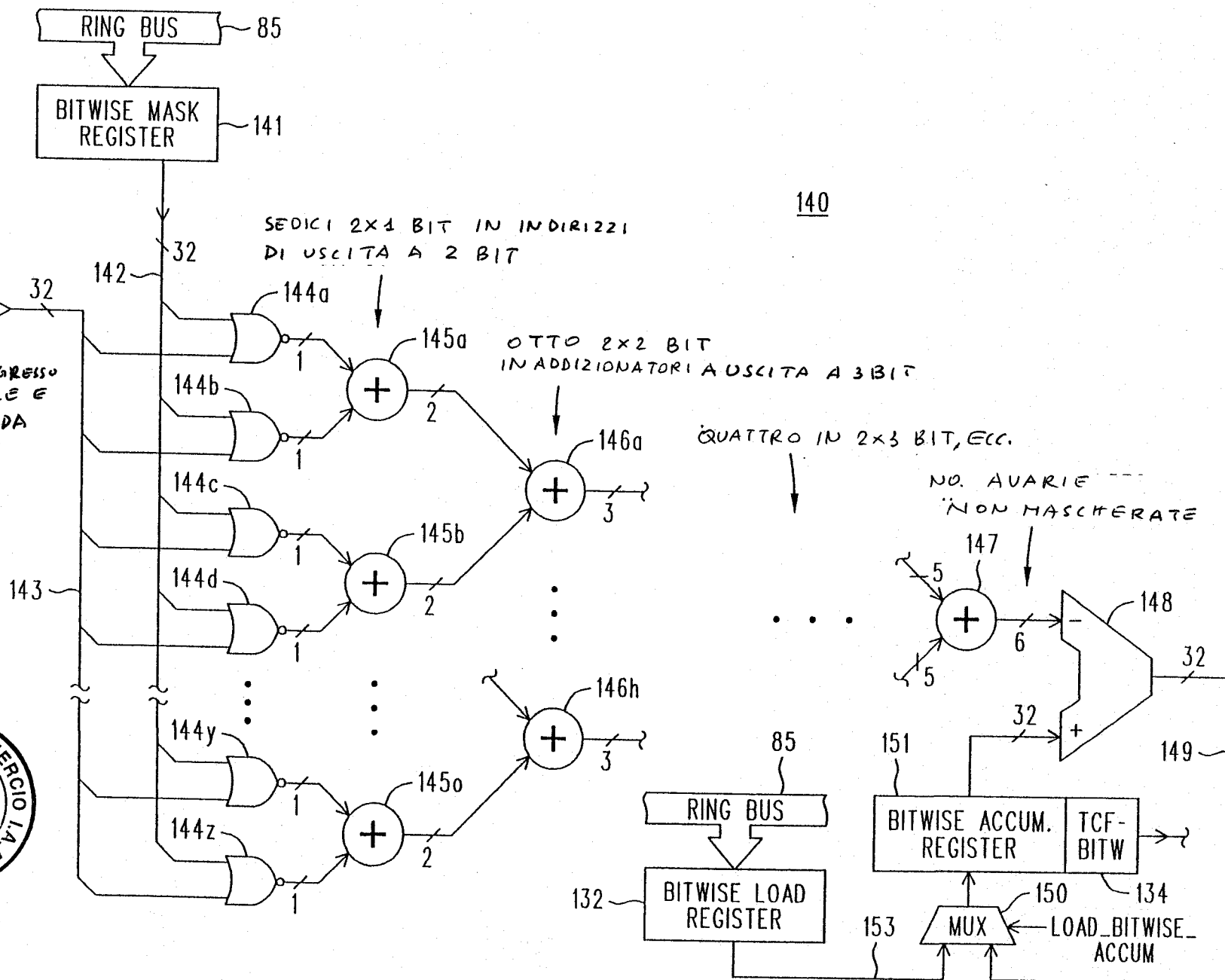


FIG. 6



7/8 PM 2001 A 000644

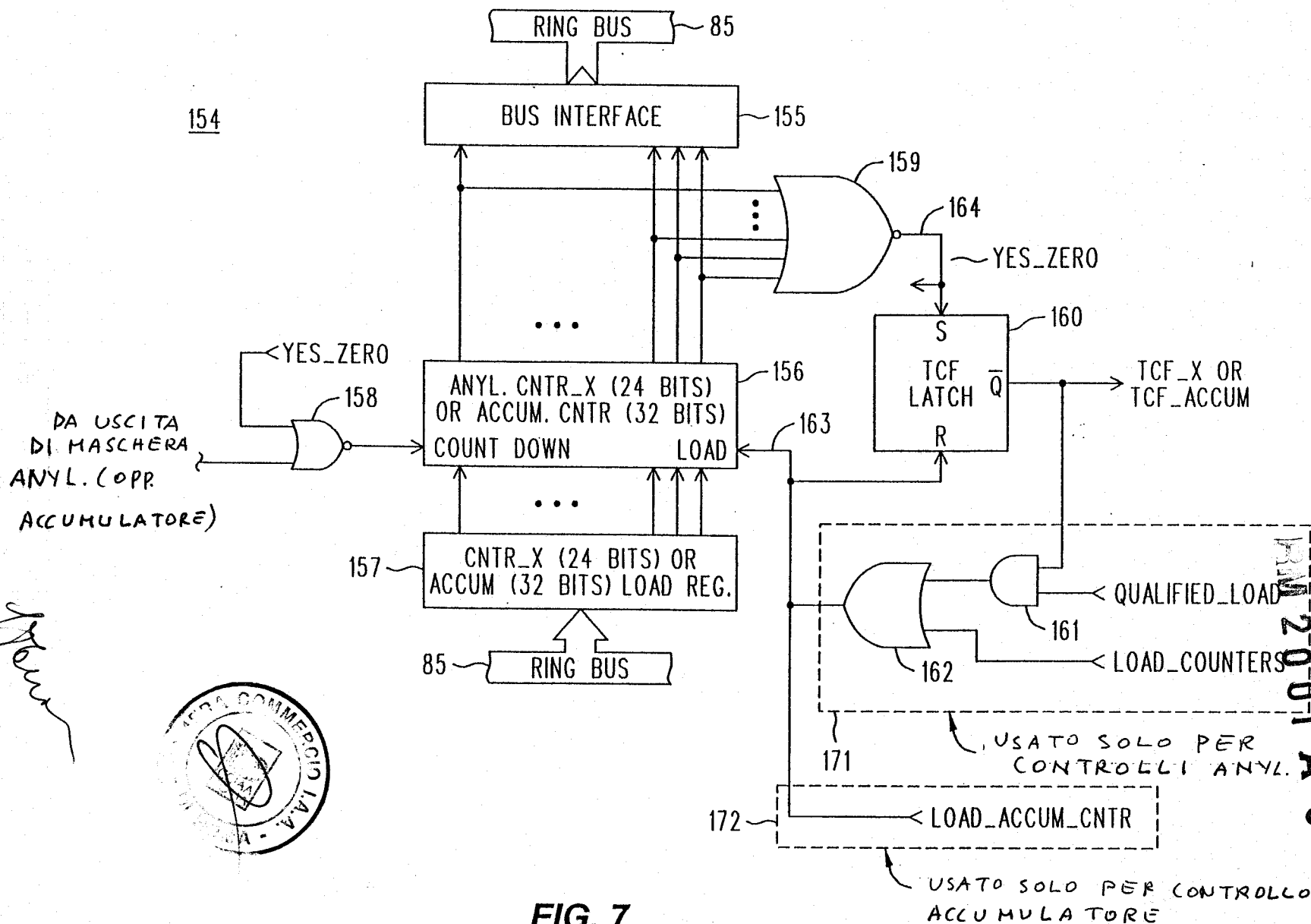
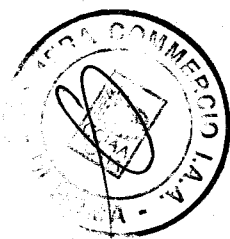


FIG. 7



RM 2001 A 000646