Università degli Studi di Padova

Facoltà di Ingegneria

Corso di Laurea Triennale in Ingeneria Meccatronica

Transitori ed interferenze EMI: tecniche di immunità a livello software

Transients and EMI interference: software-based immunity techniques

Relatore: Alessandro Sona

Laureando: Andrea Mioni

Anno Accademico 2010 - 2011

Indice

Capitolo 1

Introduzione

Le interferenze elettromagnetiche EMI (ElectroMagnetic Interference) sono fenomeni di interazione ed accoppiamento tra dispositivi elettrici ed elettronici i cui effetti possono portare ad un degrado delle prestazioni di questi, mentre nel peggiore dei casi causano guasti e condizioni di pericolo nell'utilizzo di dispositivi, apparecchiature e macchinari. Conosciuti anche come fenomeni EMI, sono particolarmente pericolosi in ambiti quali il medicale, in quanto un loro effetto può comportare danni non solo ai dispositivi o alla strumentazione in uso, ma anche agli stessi pazienti ed operatori coinvolti. I fenomeni EMI e le varie tecniche utili alla elimininazione o alla mitigazione di questi sono studiati e affrontati nella disciplina conosciuta come compatibilità elettromagnetica (EMC)la quale tratta la generazione, la trasmissione e la ricezione non intenzionale di fenomeni elettromagnetici (EM) in relazione agli effetti indesiderati che possono comportare. L'obiettivo di questa è di garantire il corretto funzionamento, nel medesimo ambiente operativo, di apparecchiature elettriche, elettroniche e delle telecomunicazioni, secondo la loro funzione.

1.1 Modalità di propagazione delle interferenze EM

La sorgente di interferenza e l'elemento che viene colpito da questa sono ovviamente dispositivi o apparecchiature elettroniche, che utilizzano correnti e tensioni, che generano campi elettrici e magnetici e che risentono di questi ultimi quando provenienti dall'esterno.

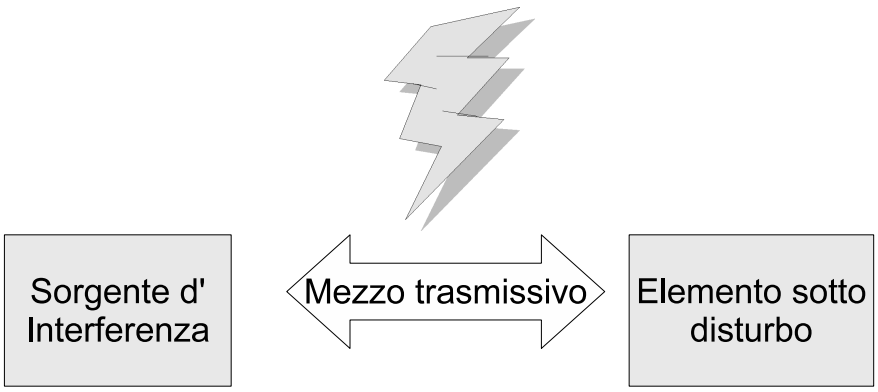


Figura 1.1: *Esempio di ambiente affetto da interferenze elettromagnetiche*

Il trasferimento dell'interferenza può avvenire :

1. Mediante correnti e tensioni in cui l'emissione e la propagazione dell'interferenza è di tipo condotto ed avviene per mezzo dei cavi conduttori di alimentazione, di comunicazione o del sistema di massa (grounding).

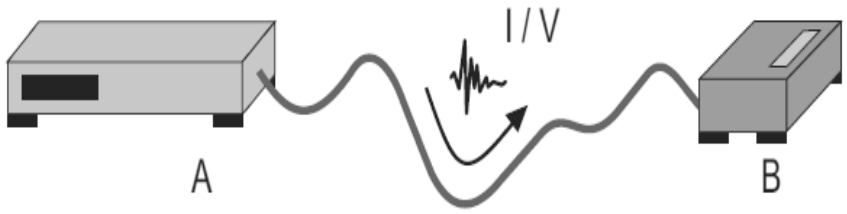


Figura 1.2: *Propagazione di tipo Condotto*

2. Attraverso i campi E e H in cui l'emissione e la propagazione dell'interferenza è di tipo irradiato ed avviene nell'aria attraverso le leggi fisiche dell'elettromagnetismo.

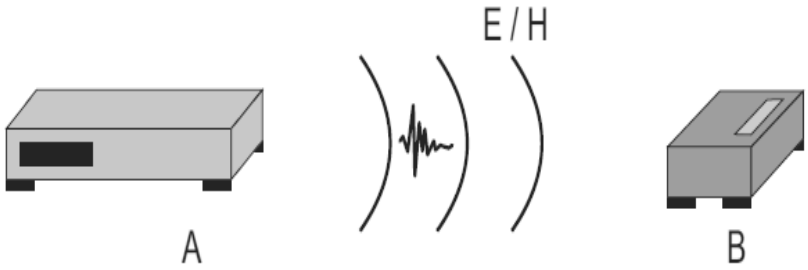


Figura 1.3: *Propagazione di tipo Irradiato*

1.2 Accorgimenti di tipo Hardware contro le interferenze

In ambito europeo, la compatibilità elettromagnetica è regolamentata principalmente dalla direttiva 2004/108/EEC, la quale nei suoi due requisiti essenziali, impone che un qualsiasi apparecchio elettrico o elettronico per poter essere immesso sul mercato comune europeo debba soddisfare ai criteri di compatibilità ed immunità.
Per criterio di compatibilità si intende che:
i disturbi elettomagnetici generati da un apparato durante il suo funzionamento devono essere di entità tale da non compromettere il funzionamento di altri apparati.
Per criterio di immunità, invece, si intende che:
l'apparato sia in grado di funzionare anche in presenza di disturbi elettromagnetici inferiori ai massimi livelli consentiti

I fenomeni EMI possono essere previsti, misurati e mitigati mediante opportune tecniche di simulazione, misurazione e progettazione che però

influiscono fortemente sul costo dei dispositivi o degli strumenti elettronici. Dal seguente grafico si nota come il costo degli accorgimenti usati per rientrare nei parametri stabiliti dalla direttiva vigente sia funzione della fase in cui il costruttore si occupa delle problematiche EMC e delle relative soluzioni.

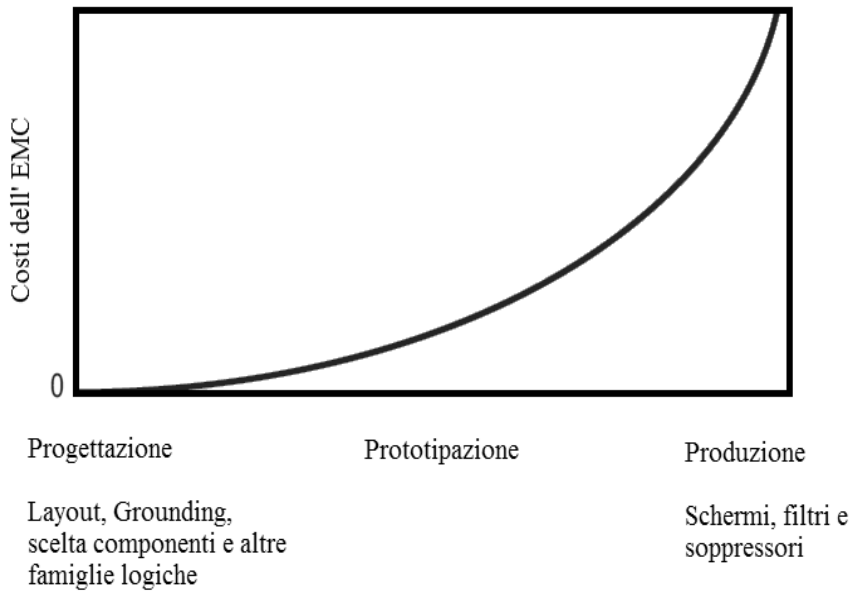


Figura 1.4: *Analisi del costo delle protezioni in funzione della fase progettuale*

Come descritto dal grafico, il costo è fortemente variabile. Parte da un valore minimo o quasi nullo nella prima fase, definita di progettazione, in cui semplici accorgimenti tecnici come ad esempio:

- la scelta di una determinata strategia di collegamenti a massa (grounding)
- la scelta della posizione e dimensione di alcuni cavi o dei collegamenti di una scheda elettronica
- l'uso di un tipo di componente elettronico o circuito integrato

- la scelta delle frequenze di temporizzazione (clock) in gioco con cui far lavorare un certo dispositivo.

possono ridurre drasticamente le interferenze EM a fronte di una spesa irrisoria e sono chiamate soluzioni di tipo preventivo. Se l'analisi EMC viene ritardata nella fase di prototipazione del prodotto, la libertà con cui il progettista può operare è notevolmente ridotta rispetto alla fase precedente. In alcuni casi, la risoluzione di eventuali problematiche EMC può richiedere la riprogettazione del progetto iniziale o di alcune sue parti, con ovvie conseguenze in termini economici e di tempo. La situazione può peggiorare ulteriormente posticipando l'analisi EMC alla fine del progetto, quando ormai alcune parti del prodotto (come ad esempio le schede elettroniche, i supporti o gli involucri) sono prossime alla produzione. Le soluzioni sono spesso ridotte ad alcune o poche possibilità, come ad esempio l'utilizzo di involucri schermanti, di cavi schermati, di filtri EMI esterni, di circuiti soppressori, di guarnizioni schermanti, o la modifica dell'involucro a causa dei nuovi componenti EMC da inserire, ecc. Si parla in questi casi di operazioni o soluzioni di tipo repressivo.

1.3 Accorgimenti di tipo Software contro le interferenze

Spesso un fenomeno EMI può portare a conseguenze negative sulle apparecchiature che basano il loro funzionamento su una struttura a microprocessore, nelle quali, quindi, è prevista una parte software. L'entità di tali conseguenze può essere più o meno grave a seconda dei casi che si presentano. I fenomeni più pericolosi per un microprocessore sono:

1. Fenomeni di tipo Transitorio con i quali si intendono le sovratensioni impulsive dovute all'accoppiamento fra i vari circuiti elettrici e che si trovano sotto forma condotta nei cavi di alimentazione, negli ingressi di comando e di segnale degli apparecchi elettrici o elettronici. I tratti distintivi di questi disturbi sono:

   - La durata molto limitata del tempo di salita dell'impulso: 5 ns
   - La durata dell'impulso: 50 ns
   - La ripetitività del fenomeno: scariche d'impulsi per 15 ms
   - La frequenza di ripetizione: successione di scariche ogni 300 ms
   - La ridotta energia degli impulsi: 1 mJ
   - L'ampiezza elevata della sovratensione < 4 kV

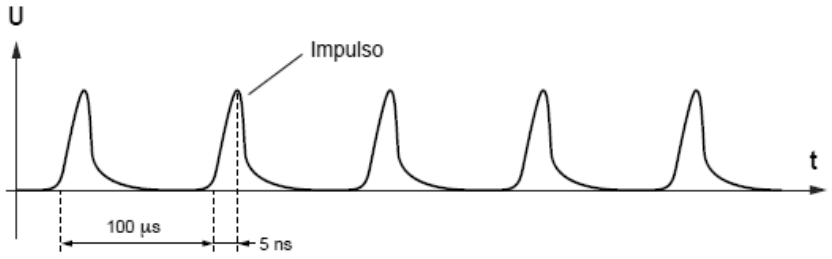


Figura 1.5: *Caratteristiche dei transitori normalizzati (tipo IEC 1000-4-4)*

2. Fenomeni dovuti alle scariche elettrostatiche con i quali si intendono gli impulsi di corrente che percorrono un oggetto qualsiasi, al momento del contatto (diretto o indiretto) di questo oggetto collegato alla massa, con un altro oggetto avente potenziale più elevato rispetto alla massa. I tratti caratteristici di questi disturbi sono:

   - La durata molto limitata del tempo di salita dell'impulso: 1 ns
   - La durata dell'impulso: 100 ns
   - Il carattere isolato del fenomeno: 1 scarica
   - La tensione molto elevata all'origine della scarica: dai 2 ai 15 KV

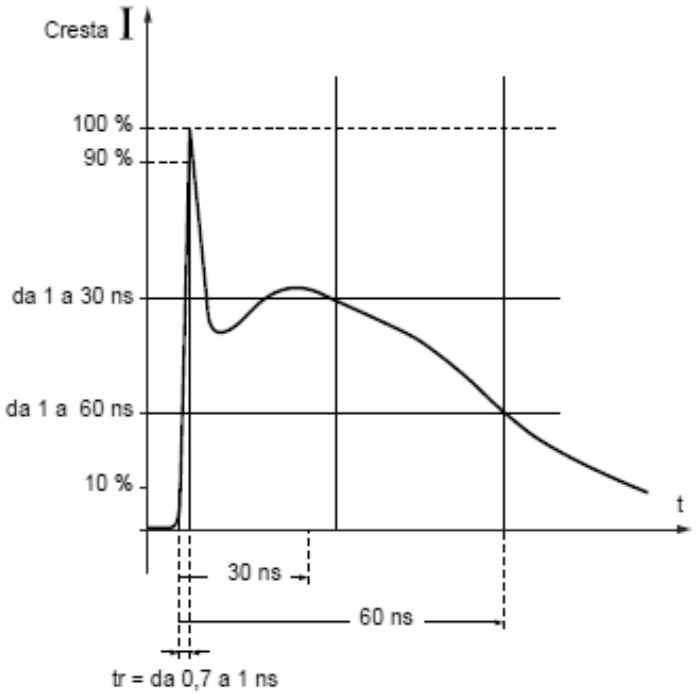


Figura 1.6: *Caratteristiche delle scariche elettrostatiche normalizzate (tipo IEC 1000-4-2)*

Queste interferenze di tipo impulsivo possono agire sul software di un sistema alterando qualche bit ed è quindi necessario organizzare il programma, ed eventualmente la struttura hardware, tenendo conto di queste possibilità. Questo tipo di soluzione richiede circuiteria aggiuntiva che aggiunge costi, peso ed aumenta le dimensioni del dispositivo e si concentra solo sulla prevenzione di guasti ed errori non essendo in grado di recuperarli, una volta avvenuti. Idealmente, al verificarsi di un'alterazione di qualche bit, il sistema dovrebbe innanzitutto accorgersi di questo problema e provvedere quanto prima al ripristino delle condizioni precedenti ma tale procedura non è interamente realizzabile tramite hardware aggiuntivo. Viene quindi utilizzato frequentemente un approccio di tipo software, il quale utilizza tecniche difensive di protezione contro le interferenze e permette di risolvere alcune particolari situazioni. L'aspetto più importante e molto apprezzato dai progettisti e soprattutto da chi commissiona la progettazione di un nuovo dispositivo è che l'approccio di tipo software risulta essere una soluzione economicamente molto più interessante di quello hardware. Le tecniche software hanno il vantaggio di poter essere utilizzate per una vasta gamma di applicazioni e differenti piattaforme, a differenza di quelle hardware che vanno adattate e dimensionate per ogni applicazione studiata . L'aggiunta di hardware in un dispositivo comporta costi extra che vengono sommati al costo dell'elemento mentre l'uso di software specifico viene sviluppato una sola volta e poi replicato facilmente.

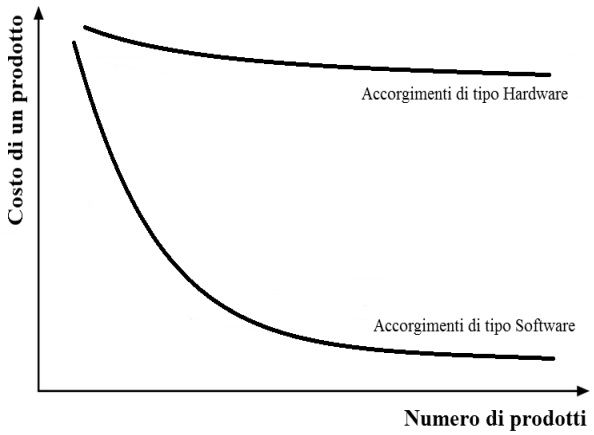


Figura 1.7: *Confronto tra i costi variabili delle due soluzione in funzione del numero di prodotti*

Verranno illustrate nelle pagine seguenti alcuni accorgimenti a livello software che permettono di avere un buon grado di affidabilità nell'uso di un qualsiasi dispositivo a microprocessore. In particolare verranno studiati le misure contro:

- errori che condizionano la lettura corretta degli ingressi.
- errori che colpiscono istruzioni di programma.
- errori che influenzano la trasmissione nella memoria dati.

Capitolo 2

Disturbi nella lettura degli ingressi

In un microprocessore le linee di I/O, utili per il collegamento dell'unità centrale con l'ambiente esterno, sono fortemente influenzate dai disturbi provenienti da esso (in molti casi sono soggette al verificarsi di scariche elettrostatiche che provocano disturbi di tipo impulsivo). Dalla figura seguente si nota come siano in posizione esterna rispetto all'involucro del microprocessore:

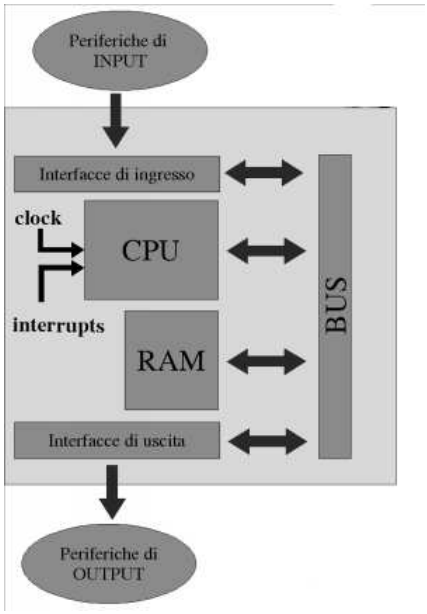


Figura 2.1: *Schema a blocchi di un comune microprocessore*

Tra queste linee vi sono quelle che trasportano i dati in ingresso e quelle che provvedono a mandare un segnale di interruzione al microprocessore dopo il verificarsi di situazioni che variano in base all'applicazione voluta (per esempio l'esaurimento di una risorsa durante il suo utilizzo). Va premesso che le tecniche qui illustrate permettono di attenuare e rendere quasi nulli i disturbi causati delle interferenze EM ma rendono le procedura di acquisizione dati molto più lenta del normale.

2.1 Segnale di Interrupt

L'innesco di un Interrupt invia una richiesta di interruzione (IRQ) nel controller. Se questa viene chiamata, la CPU, abilitata ad accettare l'interruzione, salva lo stato corrente della macchina e carica sul registro program counter l'indirizzo della routine di servizio (ISR) relativa alla richiesta effettuata. Se la richiesta di interrupt è partita da un collegamento fisicamente non utilizzato allora nella CPU viene caricato un valore qualsiasi, portando conseguenze non prevedibili e generalmente negative. Se invece l'interrupt parte da una linea realmente utilizzata, il puntatore all'istruzione corrente (IP) viene salvato sullo stack e il registro stack pointer (SP) viene incrementato. L'indirizzo della routine di servizio (ISR) viene letto e caricato dal registro IP da cui viene eseguito di conseguenza. Se, a causa delle interferenze, i segnali di interrupt sono generati troppo velocemente, lo stack cresce fino a quando tutta la memoria non viene utilizzata finendo poi per sovrascrivere i dati salvati precedentemente. Per eliminare i problemi derivanti dalle linee non utilizzate è conveniente abilitare solo quelle realmente utilizzate o in caso questo non sia possibile, predisporre una ISR a cui il programma accede ogni volta che una di queste linee è attivata. Questa routine di servizio speciale serve soltanto per tornare al programma principale e non va ad influire sul registro stack pointer, evitando il rischio di sovrascrivere dati importanti. Nel caso i disturbi interessino le linee realmente utilizzate è sufficiente adottare un piccolo accorgimento durante la lettura degli ingressi. Va tenuto conto che la scarica elettrostatica , essendo un fenomeno di tipo impulsivo, presenta durate difficilmente superiori ai 100 ns. È quindi opportuno regolare la letture degli ingressi in modo che questa siano sensibili ai livelli e non ai fronti dei segnali, in modo da eliminare totalmente la possibilità di interpretare come dato, o come comando, un impulso provocato da una scarica.

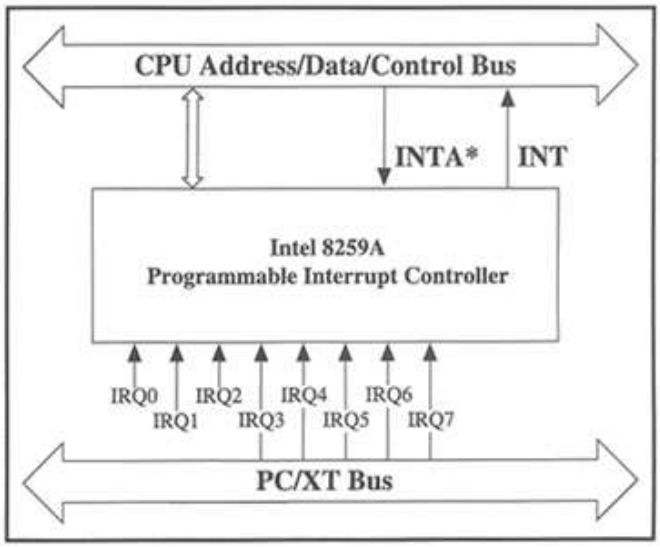


Figura 2.2: *Schema a blocchi per la gestione degli Interrupt e delle relative IRQ*

2.2 Segnale di Ingresso

Le interferenze EM possono condizionare la corretta lettura dei dati in ingresso. Per evitare questo problema vengono usate varie tecniche tra cui:

- L'elaborazione della routine di ingresso in modo che effettui due letture, con una separazione temporale di almeno 100 ns (durata media di un disturbo impulsivo), e con un confronto di quanto ricevuto. Se le due operazioni di lettura danno lo stesso risultato allora si è in presenza di un segnale non impulsivo e la lettura viene convalidata.
- Lo studio preliminare delle caratteristiche del segnale di ingresso grazie alle quali si accetta un campione solo se coerente con alcuni campioni precedenti, eliminando quindi l'influenza di disturbi impulsivi esterni. Si potrebbe, per esempio, porre che la variazione fra due campioni adiacenti non possa superare un valore massimo in modo da eliminare i segnali di tipo impulsivo che hanno un tempo si salita molto veloce.

- La determinazione di un input range per i valori in ingresso. Se un dato risulta fuori da tale range, è molto probabile che sia stato alterato da un disturbo come una scarica elettrostatica e quindi non debba esserne abilitata la lettura.

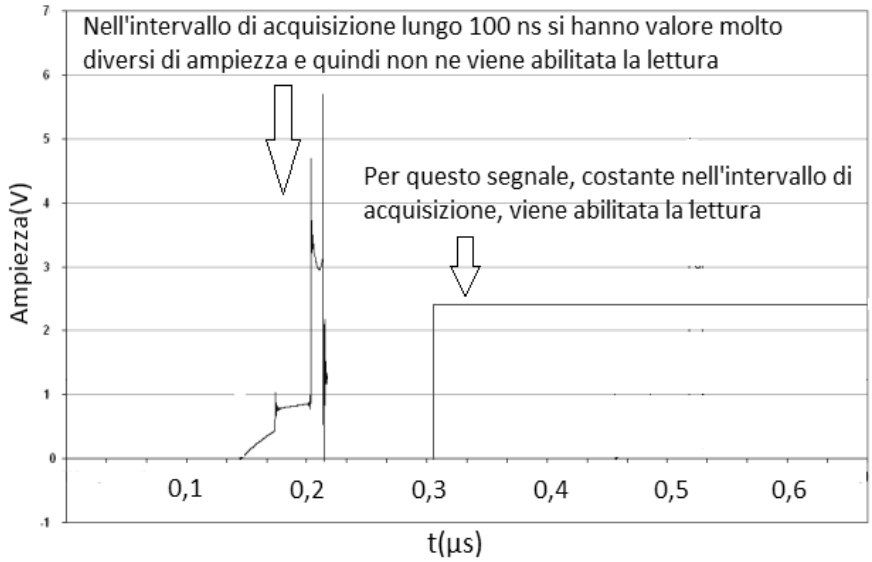


Figura 2.3: *Analisi temporale dei segnali in ingresso*

Capitolo 3

Disturbi nelle istruzioni di programma

I disturbi di tipo impulsivo possono provocare la variazione di alcuni bit nei registri della CPU o nelle istruzioni di un programma. Questo fenomeno, di difficile individuazione in quanto può causare qualsiasi tipo di errore o di rallentamento nel programma, può andare ad incidere sia sulla memoria utilizzata sia su quella non usata dal dispositivo colpito dall'interferenza.

3.1 Il Watchdog

Nel caso venga alterata la memoria utilizzata può succedere che il programma interpreti dei dati in ingresso come istruzioni da svolgere causando grossi rallentamenti nel funzionamento del dispositivo. Questo tipo di errore interrompe la corretta successione di operazioni definite inizialmente nella programmazione del microcontrollore, portandolo inevitabilmente ad entrare in uno stato di loop infinito. In molti casi questo si traduce nell'attesa di qualche istruzione finale che non però non arriva mai, oppure nell'esecuzione di un'istruzione o di elaborazione di un comando senza fine. Una situazione di loop infinito sussiste anche quando la condizione finale di un programma è irraggiungibile, in questo caso le istruzioni possono finire, ma non si avrà mai il risultato finale voluto. Per evitare questo tipo di situazione molti progettisti inseriscono un software watchdog all'interno del programma principale. Questo consiste in un contatore che viene inizializzato con una certo valore e procede a ritroso fino allo zero. Durante l'esecuzione del programma diverse routine lanciate in seguito alla riuscita delle funzioni programmate provvedono a fermare il contatore nel watchdog ed a farlo ripartire dal valore iniziale. Se il programma continua a

comunicare al watchdog dei reset ad intervalli regolari significa che sta lavorando correttamente, nel caso contrario se il watchdog non riceve segnali di reset significa che il programma non funziona correttamente ed è quindi entrato in un loop infinito. In questo caso il watchdog genera automaticamente una richiesta di interruzione ad elevata priorità, con l'attivazione di una particolare routine di servizio. Generalmente tale routine segnala l'inconveniente e permette al programma di uscire dalla situazione senza fine in cui era entrato. Se si vuole ottimizzare il funzionamento di questa tecnica allora il ciclo di aggiornamento del timer deve essere impostato della lunghezza più breve possibile e chiamato dalla funzione principale, in modo che il segnale di reset possa essere eseguito prima che possano verificarsi gravi problemi al microcontrollore.

Nella figura seguente viene introdotto lo schema realizzativo della funzione Watchdog:

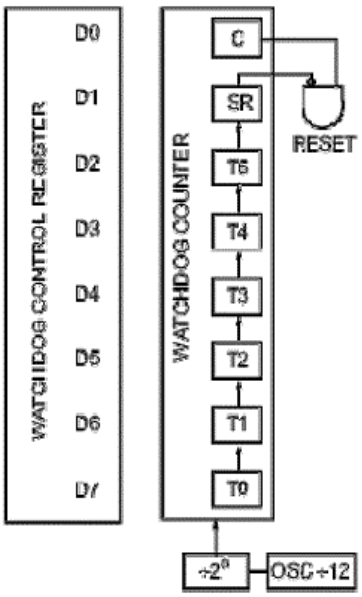


Figura 3.1: *Realizzazione della funzione Watchdog*

Il comando di attivazione e disattivazione avviene mediante il bit denominato C, settando il quale (con valore uguale a 1) si attiva il watchdog; ovviamente portando a zero questo bit la funzione viene disattivata.

Questa operazione può essere effettuata solamente in alcuni micro, precisamente in quelli la cui sigla termina con l'estensione: /SWD. Nei micro con estensione /HWD, invece tale bit è settato automaticamente dall'hardware e non può essere modificato dal software.

La richiesta di reset è implementata nel secondo bit denominato SR (Software Reset),quando questo bit viene resettato ed il bit C è allo stato logico 1, il watchdog genera un impulso di reset. Durante il normale funzionamento il registro DWDR viene decrementato con una frequenza legata al clock del micro. Tale frequenza è uguale a quella del quarzo divisa per 12 e per 2^8 (12 x 256 = 3072). Per il decremento tale registro utilizza 6 bit (D7, D6, D5, D4, D3, D2), corrispondenti ai Flip-Flop T: T0, T1, T2, T3, T4, T5). Se utilizziamo, ad esempio, un quarzo esterno da 8 MHz il contatore DWDR viene decrementato ogni 384 microsecondi infatti:

$$F_{OSC} = 8MHz \qquad F_{DWDR} = \frac{F_{OSC}}{3072} = 2,604kHZ$$

$$Quindi \quad T_{DWDR} = \frac{1}{F_{DWDR}} = 384\mu s$$

A questo punto agendo sui bit di conteggio del registro DWDR (D7, D6, D5, D4, D3, D2; 6bit; 26=64 combinazioni) possiamo generare un reset con tempo variabile da 384 microsecondi a 24,576 ms. Se, ad esempio, scriviamo nel registro del watchdog il valore esadecimale FF, abilitiamo questa periferica a generare un reset dopo 24,576 ms.

Valore nel DWDR (hex)	VD_{10} =Valore nel contatore (base 10)	Ritardo reset 384 µs*(VD_{10}+1)
00	0	384 µs
A4	41	16,128 ms
FF	63	24,576 ms

Figura 3.2: *Esempio di calcolo del tempo di ritardo del reset generato dal Watchdog.*

Per evitare che il circuito generi l'impulso di reset dovremo, prima dello scadere del tempo impostato, ricaricare il registro con il valore FF o con un altro dato.

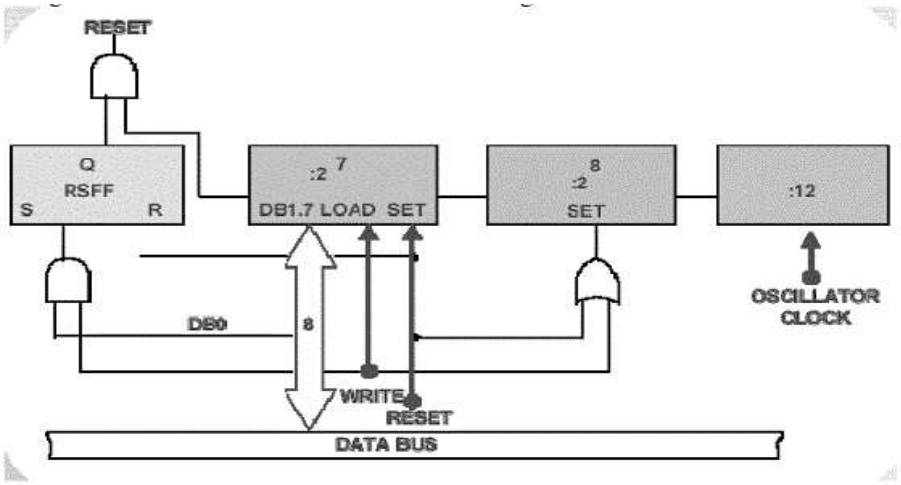


Figura 3.3: *Schema a blocchi del circuito watchdog*

Se il microcontrollore non dispone di un watchdog interno allora la sua funzione può essere replicata con l'uso di un timer interrupt o di un dispositivo esterno.

3.2 Il metodo Token Passing

Un'ulteriore barriera ai disturbi può essere ottenuta mediante il controllo del flusso di esecuzione noto come token passing . L'implementazione di questa tecnica è semplice ed efficiente in quato ogni funzione viene etichettata con un valore ID unico. Quando la funzione viene chiamata, l'ID desiderato viene salvato in una variabile globale. La funzione viene eseguita solo se l'ID funzione nella variabile globale e l'ID della funzione descritta dalla parte di codice puntata solo gli stessi. Se questi non corrispondono, allora si è verificato un errore di puntatore all'istruzione e possono essere prese misure cautelative prima dell'esecuzione della funzione sbagliata. L'implementazione del metodo token passing aumenta la dimensione del codice programma e ne rallenta quindi le prestazioni.

3.3 Le istruzioni NOP

A seguito di un disturbo può capitare che le istruzioni puntino verso porzioni di memoria non utilizzata, quindi oltre al controllo dell'indirizzo dove si effettua il fetch delle istruzioni devono essere previsti ulteriori accorgimenti adeguati. Le soluzioni più utilizzate sono:

1. Completare l'area di memoria non utilizzata con istruzioni NOP (not operations), con alla fine un salto ad una routine di servizio per la gestione degli errori. Il programma passerà in sequenza le istruzioni NOP, senza far nulla, e alla fine attiverà la routine di servizio. Molto spesso questa è un semplice reset.

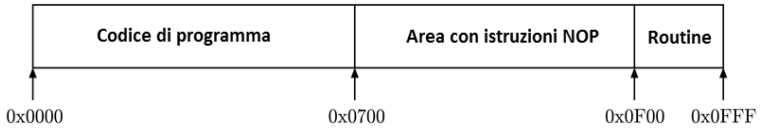


Figura 3.4: *Riempimento area inutilizzata con istruzioni NOP e routine di servizio finale*

2. Riempire l'area di memoria programma non utilizzata con istruzioni di salto alla routine di errore, la quale nella prima soluzione era posizionata solo alla fine della memoria, rendendo così molto più veloce l'individuazione e l'uscita dall'area di memoria non utilizzata.

Quest'ultima soluzione, apparentemente di ovvie e convenienti prestazioni rispetto alla precedente, non può essere attuata per ogni microcontrollore. Solitamente un' istruzione di salto richiede 3 byte, uno per il codice operativo e due per l'indirizzo. Se il microcontrollore in questione lavora ad 8 bit non è detto che l'attività di fetch errata avvenga esattamente in coincidenza del byte relativo al codice operativo dell'istruzione di salto. Se questa avviene in un indirizzo adiacente si acquisisce una istruzione errata con ben note conseguenze. Un' istruzione NOP richiede invece un solo byte, e quindi in qualsiasi posizione dell'area di memoria si ha sempre una istruzione valida ed eseguibile, ad eccezione delle ultime tre

posizioni, che contengono l'unica istruzione di salto. La soluzione con le istruzioni di tipo NOP è molto più lenta ma permette di avere un margine di errore molto più piccolo della soluzione con le istruzioni di salto.

Capitolo 4

Disturbi nella trasmissione dei dati

L'effetto di un disturbo EM può portare ad alterazioni di alcuni bit durante la trasmissione nella memoria dati RAM. Gli errori che possono verificarsi durante questa sono di tre tipi:

1. Errori single-bit (a bit singolo): coinvolgono un solo bit della sequenza trasmessa il cui valore viene trasformato da 0 a 1 o viceversa.

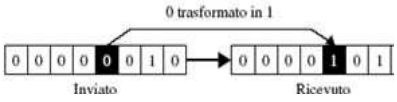


Figura 4.1: *Errori single-bit.*

2. Errori multiple-bit (a bit multiplo): coinvolgono due o più bit non consecutivi dell'unità dati, il cui valore viene trasformato da 0 a 1 o viceversa.

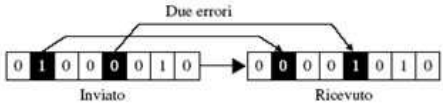


Figura 4.2: *Errori multiple-bit*

3. Errori burst (a raffica): coinvolgono due o più bit consecutivi dell'unità dati, il cui valore viene trasformato da 0 a 1 o viceversa.

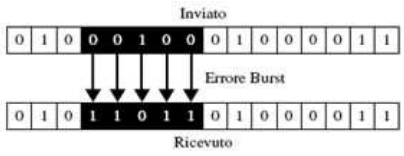


Figura 4.3: *Errori burst*

Per illustrare le varie tecniche di prevenzione e soluzione degli errori precedentemente analizzati verrà seguito questo schema:

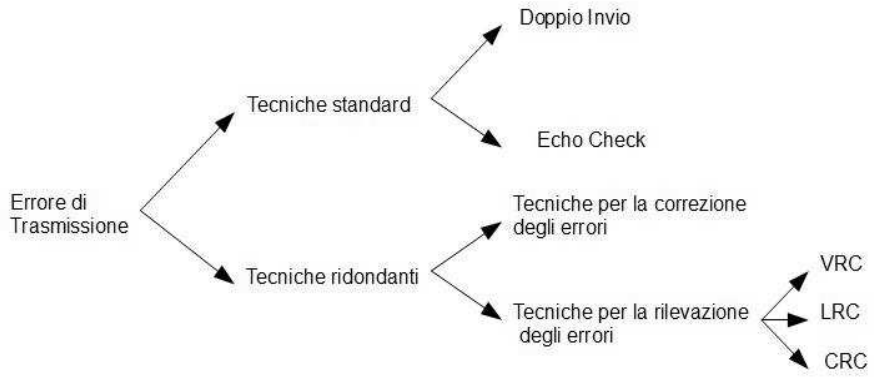


Figura 4.4: *Possibili soluzioni di un errore di trasmissione*

4.1 Tecniche standard

Le tecniche standard più semplici a livello concettuale per individuare gli errori nel processo di trasmissione sono:

- il doppio invio dei dati, nel quale il mittente invia per due volte la sequenza da trasmettere al ricevente, il quale provvederà a confrontare bit per bit le due sequenze ricevute.

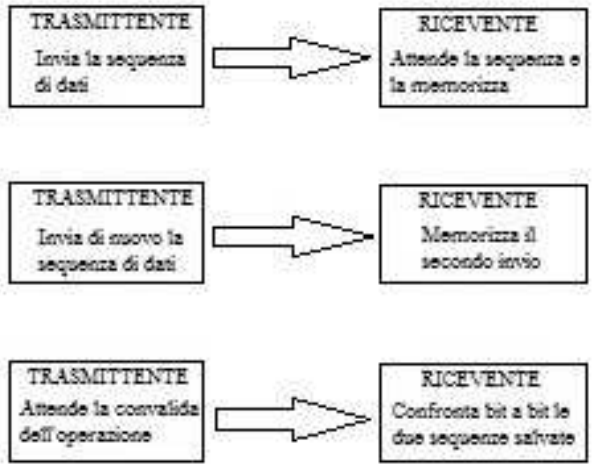


Figura 4.5: *Procedimento del Doppio Invio*

- L'Echo Check, nel quale vi è un singolo invio della sequenza di dati dal trasmittente al ricevente. Questa viene memorizzata dal ricevente e viene reinviata al trasmittente che provvede a confrontarla con il dato originario.

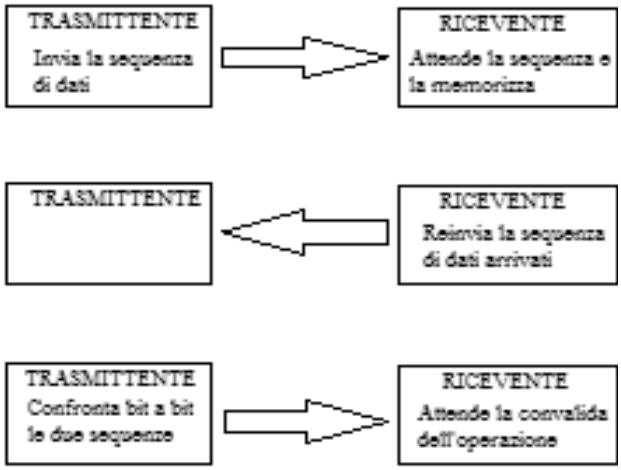


Figura 4.6: *Procedimento dell'Echo Check*

Questi metodi rendono la trasmissione affidabile con una percentuale di correttezza altissima, essendo infinitesima la probabilità di due errori sullo stesso bit, ma allo stesso tempo peggiorano l'operazione di trasmissione rendendola molto lenta. Il tempo necessario a compierla viene più che duplicato, infatti oltre alla durata dell'invio delle due sequenze deve essere aggiunto il tempo necessario per la verifica bit a bit.

4.2 Tecniche Ridondanti

Dato l'importante aspetto della velocità nella maggioranza delle comunicazioni attuali si preferisce adottare tecniche di controllo ridondanti in cui vengono aggiunti alla sequenza da trasmettere pochi bit indicanti una determinata proprietà o un aspetto del dato trasmesso. Questo insieme di bit viene comunemente definito checksum, letteralmente somma di controllo. L'aggettivo ridondante indica come una volta effettuata l'operazione di verifica e convalida del dato ricevuto questi bit vengano definitivamente eliminati dalla sequenza.

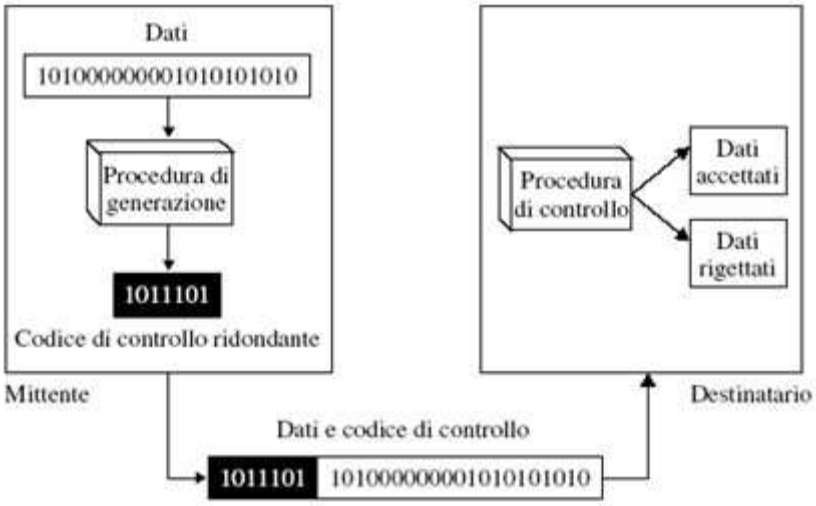


Figura 4.7: *Rilevamento degli errori con tecnica di ridondanza*

Le tecniche ridondanti possono essere riassunte in due grandi famiglie:

- tecniche di correzione degli errori: permettono di scovare la posizione dell'errore avvenuto in modo da poter effettuare la correzione.
- tecniche di rilevazione degli errori: permettono di verificare la correttezza della trasmissione ma non danno ulteriori informazioni. In pratica convalidano o bloccano l'operazione di trasmissione.

I codici per la correzione degli errori vengono usati alcune volte per la trasmissione di dati quando il canale è simplex e quindi non è possibile richiedere la ritrasmissione dei dati. L'utilizzo di queste tecniche è gravoso in termini di tempo e risorse realizzative e va quindi valutato in base all'applicazione che il dispositivo utilizzato dovrà effettuare . Quindi, più spesso, si preferisce la rilevazione degli errori con conseguente ritrasmissione in quanto essa è molto più efficiente e veloce, e comporta pure un minor peso in termini di bit di controllo.

4.2.1 Il controllo di Parità Semplice

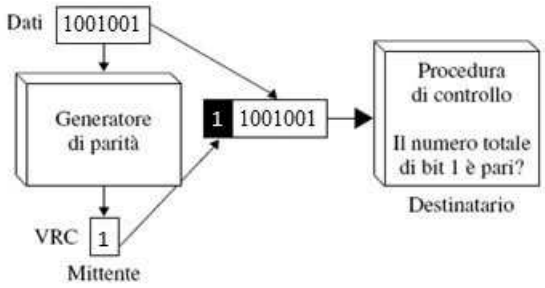


Figura 4.8: *Procedimento del Controllo di Parità*

Il controllo di parità (detto talvolta VRC, per Vertical Redundancy Check o Vertical Redundancy Checking) è uno dei sistemi di rilevazione d'errore più semplici. Il suo nome deriva dal fatto che era inizialmente utilizzato sulla codifica dei caratteri su nastro perforato, caratteri che erano rappresentati verticalmente rispetto al nastro stesso. Esso agisce nei protocolli orientati al carattere e consiste nell'aggiungere un bit, detto appunto bit di parità, in testa ad un certo numero di bit di dati, definito sequenza, (generalmente 7 bit utili a formare un byte con il bit di parità) il cui valore serve ad avere un numero di 1 pari nella sequenza. In caso di numero di 1 pari il bit di parità vale 0 e non influenza il conteggio dei bit, mentre nel caso di bit a 1 dispari il bit di parità vale 1 incrementando così il numero di 1 e rendendolo pari. Vengono qui di seguito illustrati alcuni esempi utilizzanti il controllo di Parità.

Mittente:

- Vuole spedire la sequenza 1001001
- Viene calcolato il bit di parità:

$$1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 0 \oplus 1 \equiv 1$$

- Viene aggiunto il bit di parità in coda e viene spedita la sequenza: 10010011

Destinatario:

- Dopo la trasmissione riceve la sequenza: 10010011

- Viene calcolata la parità della sequenza:

$$1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 0 \oplus 1 \oplus 1 \equiv 0$$

- Convalida l'operazione di trasmissione dopo aver osservato la parità della sequenza.

Nella trasmissione si possono verificare errori nella sequenza e nel bit di parità. Ecco come si comporta il metodo.
Mittente:

- Vuole spedire la sequenza 1001001

- Viene calcolato il bit di parità:

$$1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 0 \oplus 1 \equiv 1$$

- Viene aggiunto il bit di parità in coda e viene spedita la sequenza: 10010011

Errore nella Trasmissione: errore nella sequenza!
Destinatario:

- Dopo la trasmissione riceve la sequenza: 10**1**10011

- Viene calcolata la parità della sequenza:

$$1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 \oplus 0 \oplus 1 \oplus 1 \equiv 1$$

- L'operazione di trasmissione non viene convalidata dopo aver osservato la disparità della sequenza.

Mittente:

- Vuole spedire la sequenza 1001001

- Viene calcolato il bit di parità:

$$1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 0 \oplus 1 \equiv 1$$

- Viene aggiunto il bit di parità in coda e viene spedita la sequenza: 10010011

Errore nella trasmissione: errore nel bit di parità!
Destinatario:

- Dopo la trasmissione riceve la sequenza: 1001001**0**
- Viene calcolata la parità della sequenza:

$$1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \equiv 1$$

- L'operazione di trasmissione non viene convalidata dopo aver osservato la disparità della sequenza.

Questo sistema presenta però una lacuna evidente:
Mittente

- Vuole spedire la sequenza 1001001
- Viene calcolato il bit di parità:

$$1 \oplus 1 \oplus 0 \oplus 0 \oplus 0 \oplus 1 \equiv 1$$

- Viene aggiunto il bit di parità in coda e viene spedita la sequenza: 11001001

Errore doppio nella trasmissione: errore nel bit di parità e nella sequenza!
Destinatario:

- Dopo la trasmissione riceve la sequenza: 1**0**001100**0**
- Viene calcolata la parità della sequenza:

$$1 \oplus 0 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 0 \oplus 0 \equiv 0$$

- L'operazione di trasmissione viene convalidata dopo aver osservato la parità della sequenza.

Qui viene rilevata la vera carenza del sistema di controllo di parità in quanto esso rileva solo gli errori in numero dispari, pari quindi solamente al 50% degli errori totali (percentuale troppo bassa). In caso di numero pari di errori il sistema convalida l'operazione di trasmissione lasciando dunque passare informazione corrotta senza interrompere immediatamente il processo in corso. Questi problemi, uniti alla sua semplice realizzazione (bastano infatti poche porte logiche Exor per realizzare il controllo), fanno

si che questo sistema sia abbastanza utilizzato in applicazioni in cui la probabilità di errore sia molto bassa e nei casi in cui avendo a disposizione un canale bidirezionale sia possibile richiedere la ritrasmissione del messaggio. Si utilizza principalmente nella prevenzione degli errori durante lettura dei dati memorizzati su nastri magnetici e per la memorizzazione dei byte nelle memoria principale.

Per utilizzare il controllo di parità di tipo pari sul carattere, il mittente combina i sette bit del codice con una cascata di porte EXOR, che generano il bit di parità pari; il bit P verrà inviato sul canale insieme agli altri sette bit.

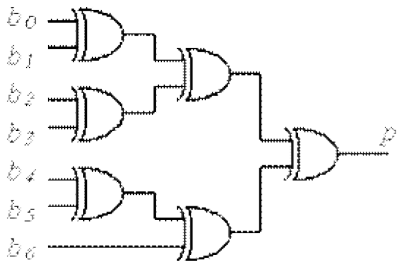


Figura 4.9: *Circuito per il calcolo del bit di parità p*

Il ricevitore verifica la correttezza del messaggio con un analogo circuito, applicato agli otto bit ricevuti.

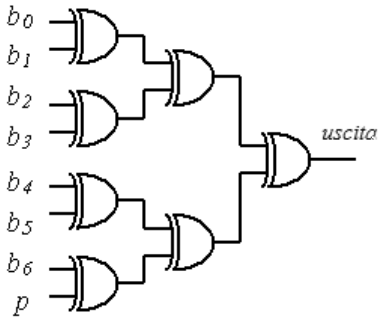


Figura 4.10: *Circuito che verifica la correttezza del messaggio arrivato*

In caso di trasmissione corretta l'uscita sarà uguale a 0 infatti questo indica che la sequenza ha un numero pari di bit mentre se l' uscita assume il valore 1 si è verificato un errore.

4.2.2 Il controllo di Parità Incrociato

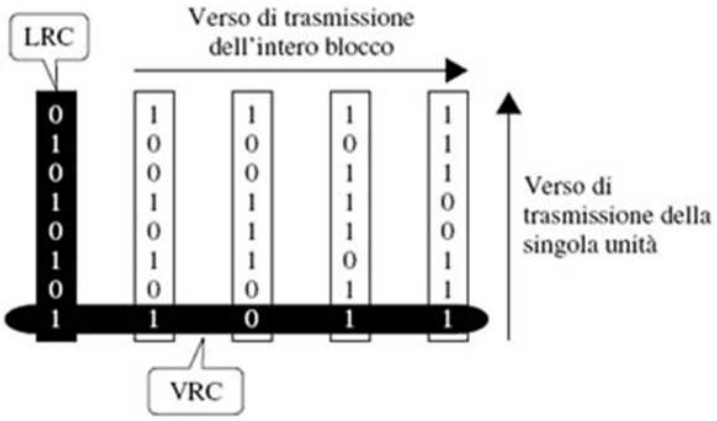


Figura 4.11: *Procedimento del Controllo di Parità Incrociato*

Il controllo di parità incrociato LRC (Longitudinal Redundancy Check) detto anche controllo di parità longitudinale, a differenza del VRC opera a livello del messaggio e non del singolo carattere. In questo tipo di controllo i dati sono raggruppati in modo tale da formare una matrice nella quale ogni riga rappresenta un carattere, mentre il numero delle colonne è uguale al numero di bit utilizzati per la codifica dei caratteri stessi. Si effettuano quindi due operazioni:

- il controllo della parità su ogni riga e quindi su ogni carattere.
- il controllo della parità su ogni colonna.

Il metodo di rilevazione LRC può essere dunque visto come una variante del VRC di tipo bidimensionale. Come nel VRC si ha infatti l'aggiunta del bit di parità ad ogni unità dati. Ad ogni blocco viene però aggiunta una unità supplementare che contiene i bit di parità associati alle sequenze di bit corrispondenti del blocco.

Per il metodo LRC si potrà dunque utilizzare la stessa configurazione hardware usata per il VRC integrandola con il controllo sui bit che occupano la stessa posizione nella sequenza di dati:

Ecco un esempio di trasmissione della stringa ANDREA avvenuta correttamente:

Lettera	Codice ASCII su 7 bit	LRC
A	1000001	0
N	1001110	0
D	1000100	0
R	1010010	1
E	1000101	1
A	1000001	0
VRC	0011101	0

Ecco invece un esempio di trasmissione della stringa ANDREA avvenuta con il verificarsi di due errori nella ricezione della lettera N che il controllo di parità VRC non avrebbe notato:

Lettera	Codice ASCII su 7 bit	LRC
A	1000001	0
N	1**1**01**0**10	0
D	1000100	0
R	1010010	1
E	1000101	1
A	1000001	0
VRC	0**0**111**1**01	0

Si può notare come i bit di parità dei singolo carattere siano relativamente corretti mentre il VRC sia sbagliato proprio nelle colonne dove si è verificata la commutazione del valore originario. Se per la trasmissione di queste sequenze di caratteri si avesse utilizzato il controllo di Parità semplice non sarebbe stato possibile rilevare l'errore in quanto quel controllo agisce a livello carattere e non riesce a scovare gli errori doppi. Questa tecnica molto più efficace si basa sull'aggiunta di informazione calcolata non sui singoli codici delle parole che costituiscono il messaggio ma su di un blocco di dati. Il controllo di Parità incrociato assicura maggiore affidabilità nell'individuazione degli errori di tipo multiple-bit e burst rispetto al controllo di Parità semplice, ma ha ancora dei limiti perché non rileva

gli errori verificatisi su un numero pari di bit nella stessa posizione. Questo metodo risulta inoltre insensibile all' ordine delle parole del messaggio, infatti se l'ordine viene modificato, il valore del campo checksum rimane lo stesso e la trasmissione viene convalidata nonostante l'errore verificatosi. La ridondanza di questo controllo non dipende dalla dimensione del messaggio, ma dal numero di bit che compongono ogni sequenza trasmessa infatti se ogni sequenza è composta da n bit, anche la somma di controllo sarà composta da n bit, indipendentemente dalla lunghezza del messaggio. L'obiettivo è quello di ottenere la più alta possibilità di rilevazione di errori con la minor ridondanza introdotta possibile.

Esempio: durate la trasmissione si è verificato un errore che ha commutato i primi due bit della prima lettera A e della N.

Lettera	Codice ASCII su 7 bit	LRC
A	0100001	0
N	0101110	0
D	1000100	0
R	1010010	1
E	1000101	1
A	1000001	0
VRC	0011101	0

L'algoritmo convalida erroneamente la trasmissione del blocco di dati dopo aver verificato la correttezza del VRC e del LRC

4.2.3 Il controllo CRC

Il metodo di rilevazione degli errori chiamato codice di ridondanza ciclica CRC (conosciuto anche come codice polinomiale) è quello più utilizzato nelle trasmissioni di dati in quanto rappresenta l'algoritmo più affidabile tra quelli studiati. La trattazione seguente sarà affrontata con un taglio molto più teorico di quello visto nei precedenti metodi; l'apparente difficoltà dei concetti sarà poi giustificata pienamente dall'efficacia di questo algoritmo. Questo metodo considera le sequenze di bit come rappresentazioni di polinomi con coefficienti 0 e 1. Un messaggio di k bit viene visto come una lista di coefficienti di un polinomio a k termini compresi tra x^{k-1} e x^0. Questo polinomio è detto di grado k-1. Il bit di livello più alto (il più a sinistra) è quindi il coefficiente di x^{k-1}; il bit successivo è coefficiente di x^{k-2} e così via.

Esempio: considero il byte 00110001, esso rappresenta un polinomio ad otto termini con coefficienti 0, 0, 1, 1, 0, 0, 0 e 1 e cioè del tipo:

$$x^4 + x^5 + x^0$$

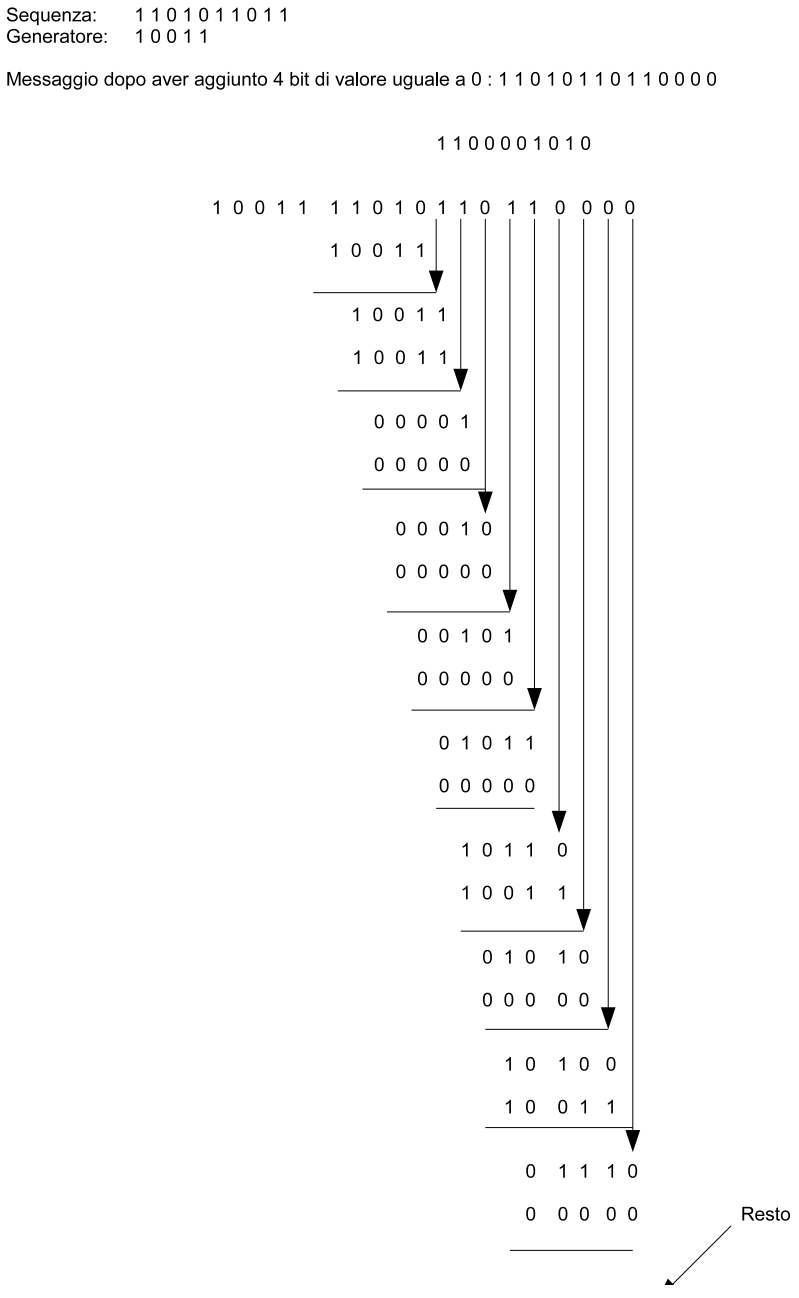
Nell'impiego di questo metodo, il mittente e il ricevente devono accordarsi in anticipo su un generatore polinomiale, G(x). Entrambi i bit di più alto e più basso ordine di questo devono essere a uno. Per calcolare il blocco checksum, ossia la parte ridondante che verrà aggiunta in coda al messaggio da trasmettere, di alcuni blocchi di dati di m bit, corrispondenti al polinomio M(x), il blocco di dati deve essere più lungo del generatore polinomiale. L'idea di fondo è quella di completare il blocco di dati da trasmettere con la somma di controllo calcolata in modo che il polinomio rappresentato dalla componente messaggio-checksum sia divisibile per G(x). Quando il ricevente ottiene il messaggio verificato, esso prova a dividerlo per G(x). Se questa operazione da un resto allora significa che deve esserci stato un errore di trasmissione. Il procedimento per il calcolo del blocco checksum è questo:

1. Sia r il gradi di G(x). Si appendano r bit 0 dopo il bit di ordine più basso del pacchetto in modo che esso arrivi a contenere m+r bit e corrisponda al polinomio $x^rM(x)$.

2. Si divida la sequenza di bit corrispondete a G(x) per la stringa di bit corrispondente a $x^rM(x)$ usando la divisione modulo 2. Essa viene svolta come se fosse binaria ad eccezione del fatto che la sottrazione è eseguita in modulo 2. Un divisore è contenuto nel dividendo se il dividendo ha tanti bit quanti il divisore.

3. Si sottragga il resto (che è sempre composto di r bit o di meno) dalla stringa i bit a x(r)M(x) usando la sottrazione modulo 2. Il risultato rappresenta il blocco di dati verificati da trasmettere e verrà chiamato in seguito T(x).

Si riporta un esempio di calcolo di checksum per il pacchetto 1101011011. Viene utilizzato il generatore polinomiale 10011 che determina G(x) = x^4+x+1 .



T(x) è quindi divisibile (in modulo 2) per G(x) in quanto in ogni pro-

blema di divisione se si sottrae il resto dal dividendo, quello che ne rimane è divisibile per il divisore. Questo algoritmo ha un grado di rilevazione degli errori molto alto che può essere così dimostrato: si immagini che sia avvenuto un errore di trasmissione in cui invece di ricevere la sequnza T(x), si riceve la stringa T(x) + E(x). Ogni bit uguale a 1 in E(x) corrisponde ad un bit invertito, nel caso ci siano k bit in E(x) allora sono avvenuti k errori di un singolo bit. Dopo aver ricevuto la sequenza verificata, il ricevente la divide per G(x); esegue quindi l'operazione : [T(x) + E(x)] / G(x).
T(x) / G(x) è uguale a 0 quindi il risultato della computazione è E(x)/G(x). Gli errori che corrispondo a polinomi contenenti il fattore G(x) non saranno rilevati mentre tutti gli altri si.

- Nel caso di errori di singolo bit: $E(x) = x^i$ dove i determina il bit errato. Se G(x) contiene due o più termini, non sarà mai in grado di dividere E(x) e quindi tutti gli altri errori di singolo bit saranno riconosciuti.

- Nel caso di errori isolati di singoli bit: $E(x) = x^i+x^j$ dove $i > j$ il quale può anche essere scritto come $E(x) = x^i(x^{i-j} + 1)$. Se si assume che G(x) non sia divisibile per x, una condizione sufficiente affinché tutti gli errori doppi siano riconosciuti è che G(x) non divida x^k+1 per ogni k fino al valore massimo di i-j (cioè fino alla lunghezza massima della sequenza). Vengono usati solitamente polinomi semplici e di basso grado che danno protezione a lunghe sequenze: per esempio $x^{15} + x^{14} + 1$ non divide $x^k + 1$ per ogni k minore di 32768.

- Nel caso di un numero dispari di bit errati, E(x) contiene un numero dispari di termini. Considerando il fatto che non esiste un polinomio con un numero dispari di termini che abbia x + 1 come fattore nel sistema modulo a 2 è possibile, rendendo x + 1 fattore di G(x), determinare tutti gli errori consistenti in un numero dispari di bit invertiti. Per verificare questo fatto assumiamo che: E(x) abbia un numero dispari di termini e che sia divisibile per x + 1. Fattorizziamo E(x) come (x + 1)* Q(x). Calcoliamo ora E(1) = (1 +1)*Q(1). Poiché 1 + 1 = 0 (modulo 2), E(1) deve essere quindi 0. Se E(x) ha un numero dispari di termini sostituendo x con 1 otterremo 1 come risultato e quindi nessun polinomio con un numero dispari di termini è divisibile per x + 1.

- Nel caso di errori a raffica: Un codice polinomiale con r bit di controllo riconosce tutte le raffiche di errori di lunghezza <= r Una raffica di errori di lunghezza k può essere rappresentata da $x^i*(x^k\text{-}1+\ldots+1)$

dove i indica la distanza della raffica di errori dall'estremità destra della sequenza ricevuta. Se G(x) contiene un termine x^0 non avrà x^i come fattore, quindi se il grado dell'espressione parametrizzata è minore del grado di G(x), il resto non potrà mai essere zero. Se la lunghezza della raffica è di r+1, il resto della divisione per G(x) sarà nullo se e solo se la raffica è identica a G(x). Per definizione di raffica il primo e l'ultimo bit devono essere a 1 e di conseguenza l'identità dipende dagli r- 1 bit intermedi. Se tutte le combinazioni vengono considerate equiprobabili la probabilità che un tale pacchetto errato venga accettato come valido è di $(1/2)^{r-1}$. Si può anche dimostrare che quando si ha una raffica di errori di lunghezza maggiore a r + 1, oppure una serie di raffiche più corte, la probabilità che una sequenza passi inosservata è di $(1/2)^r$, assumendo che tutte le combinazioni di bit siano equiprobabili.

Questi polinomi sono diventati lo standard internazionale:

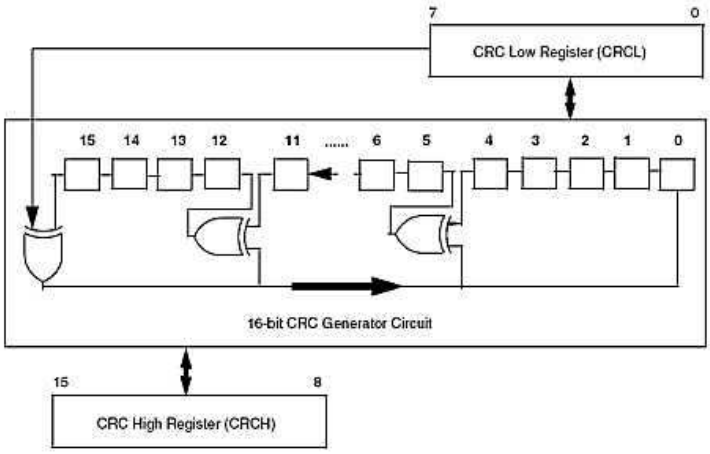
Tipo di CRC	Polinomio e rispettivo valore in bit
CRC-12	$x^{12} + x^{11} + x^3 + x^2 + x^1 + 1$ 1100000001111
CRC-16	$x^{16} + x^{15} + x^2 + 1$ 11000000000000101
CRC-CCITT	$x^{16} + x^{12} + x^5 + 1$ 10001000000100001
CRC32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$ 100000100110000010001110110110111

Tutti e tre contengono x +1 come fattore primo per il vantaggio illustrato precedentemente. Il CRC-12 viene utilizzato per sequenze di 6 bit mentre gli altri due per sequenze di 8 bit. Un checksum di 16 bit come CRC-16 oppure CRC-CCITT riconosce:

- errori singoli e doppi
- errori con un numero dispari di bit

- tutte le raffiche di lunghezza minore o uguale a 16
- il 99,997% delle raffiche di lunghezza 17 bit
- il 99,987% delle raffiche d 18 bit o più lunghe

Nonostante la computazione richiesta per il calcolo della checksum sembri alquanto complicata, in realtà può essere realizzata mediante un semplice circuito con registri di shift per calcolare e verificare tramite hardware la somma di controllo. Tale strumento è utilizzato quasi sempre:



Il procedimento per il calcolo del CRC non è complicato in quanto fondamentalmente consiste in una serie di operazioni di scorrimento e di OR esclusivo. L'aritmetica modulo due aiuta notevolmente la procedura di calcolo: una divisione viene eseguita tramite operazioni di scorrimento (shift), mentre la sottrazione è eseguita tramite l'operazione di OR esclusivo (XOR). L'implementazione hardware richiederà un registro a scorrimento con un numero di bit uguale al grado del polinomio generatore, più una porta XOR associata ad ogni bit avente un '1' come coefficente nel polinomio generatore.

Il modulo CRC utilizza solamente due registri, CRCH e CRCL. Quando il calcolo del CRC inizia, un valore di iniziale programmabile viene caricato nel registro CRCH:L; lo standard previsto da CRC16-CCITT, per esempio, utilizza un valore pari a 0xFFFF. Una volta caricato il valore iniziale, l'algoritmo può proseguire nel modo seguente: Ciascun byte del messaggio viene caricato nel registro CRCL; ciò sincronizza il modulo CRC

che carica ciascun byte (partendo da quello più significativo) nel registro a scorrimento, e la procedura deve essere ripetuta fino a quando tutti i byte sono stati processati. A questo punto, il registro CRCH:L contiene il CRC calcolato. Un nuovo calcolo può essere eseguito scrivendo un nuovo valore nel registro CRCH:L. Possono essere utilizzati altri valori iniziali; il valore 0xFFFF è quello conforme allo standard CRC16-CCITT, ma altri valori comunemente utilizzati sono 0x102D e 0x0000.

Metodo	Percentuale di individuazione degli errori	Velocità computazionale	Peso in bit
VRC	50%	Alta	Elevato, ha le stesse dimensioni della sequenza da trasmettere
LRC	50-80%	Alta	Elevato, ha le stesse dimensioni delle sequenze di cui è composto il messaggio da trasmettere
CRC	99,998%	Medio-Alta	Modesto, ha le dimensioni di un resto

Dalla tabella si nota come il CRC sia molto migliore degli altri due metodi, esso presenta una percentuale di individuazione degli errori molto più alta, una buona veloce di computazione e soprattutto è molto meno ingombrante degli altri due e permette quindi, a fronte dell'aggiunta di pochi bit, di scovare efficacemente errori in un gran numero di bit di dati. Nella realtà tutte le reti locali utilizzano tecniche CRC (nelle LAN Ethernet viene adottato il CRC-32). Internet utilizza un CRC a 16 bit e la maggior parte dei protocolli di reti geografiche dispone di una somma di controllo ciclica.

Capitolo 5

Conclusioni

Vi sono molti libri che trattano di Compatibilità Elettromagnetica e delle relative soluzioni tramite hardware aggiuntivo nei dispositivi utilizzati.
In questa Tesi si è voluto soffermarsi sulle strutture a microprocessore, le quali non permettono grosse modifiche hardware, per analizzare le tecniche a livello software contro le interferenze EMI. In queste strutture i segnali di interferenza più pericolosi sono rappresentati da quelli di tipo impulsivo, come la scarica elettrostatica, e possono comportare l'alterazione di uno o più bit nel microprocessore. Si è analizzato quali tipi di dati possono essere alterati, portando l'attenzione sulla lettura degli ingressi, sulle istruzioni contenute nel IR (istruction register) e sulla trasmissione di sequenze di dati.
Tutte le tecniche illustrate, sebbene riguardanti applicazioni molto diverse, hanno presentato due aspetti comuni: la velocità e la semplice realizzazione. La velocità è una caratteristica molto importante nei sistemi a microprocessore, i quali lavorano a frequenze molto alte, dell'ordine dei Ghz, e quindi si è cercato di individuare tecniche che non rallentassero troppo l'esecuzione delle operazioni programmate. La semplice realizzazione è invece una caratteristica universale che qualunque progettista mira ad avere nel suo prodotto, infatti questa quasi sempre si traduce in un significativo risparmio nel costo della produzione finale.
L'attenzione verso questi due aspetti ha portato ad escludere applicazioni come le tecniche per la correzione degli errori, le quali vengono utilizzate in alcuni casi ma comportano una lentezza e un ingombro realizzativo più elevato rispetto alle tecniche di rilevazione degli errori.

Bibliografia

[1] P. Clayton, *Compatibilit Elettromagnetica*. Biblioteca Scientifica Hoeplil, 4th ed., 1999. ISBN: 88-203-2210-2.

[2] A. Tanenbaum, *Reti di Computer*. UTET Libreria s.r.l., 9th ed., 2001. ISBN: 88-7750-453-6.

[3] C. White, *Reti di comunicazione per l'azienda*. Apogeo, 2001. ISBN:88-87303-861-0.

[4] "Il controllo degli errori –." Online available http://it.kioskea.net/contents/base/control.php3, 2009.

[5] "Immunity-aware programming –." Online available http://en.wikipedia.org/wiki/Immunity-aware-programming, 2011.