Sicurezza e affidabilità nel progetto di sistemi embedded

In questa seconda parte dell'articolo vengono delineate le caratteristiche hardware del microcontrollore necessarie per il supporto dei meccanismi di affidabilità e le metodologie da adottare per rilevare eventuali difetti software

Abdul Aleaf
Applications manager
National Semiconductor

II parte

Alcune caratteristiche dell'hardware del microcontrollore rivestono un'importanza particolare per il supporto dei meccanismi di affidabilità richiesti dallo standard UL 1998. I progettisti devono essere a conoscenza di tali caratteristiche in modo da evitare una scelta errata del microcontrollore. Questo è un esempio delle difficoltà - o addirittura dell'impossibilità - che si incontrano quando si vogliono eseguire modifiche per conferire doti di affidabilità a un prodotto che non è stato concepito fin dall'inizio del progetto per garantire tali doti. Il costo legato alla scelta del processore più idoneo è irrilevante prima della progettazione dell'hardware e della scrittura del software, ma può diventare proibitivo nel caso si apportino variazioni in una fase avanzata del progetto stesso.
Le caratteristiche hardware necessarie per garantire l'affidabilità si possono così riassumere:
- l'architettura del dispositivo deve assicurare che tutte le istruzioni vengano eseguite al di fuori della memoria di programma, che contiene il software applicativo e il software relativo alla diagnostica e alla sicurezza;
- poiché il software nella memoria di programma del microcontrollore riveste un ruolo di primaria importanza nel funzionamento del dispositivo, il set di istruzioni deve contenere un'istruzione atta a semplificare l'esecuzione delle operazioni di checksum (somma di controllo) e di CRC (Cyclic Redundancy Check - controllo a ridondanza ciclica) in fase di esecuzione;
- le caratteristiche di supervisione implementate via hardware presenti a bordo, come ad esempio sul watchdog timer e il circuito per la rilevazione dei brown out (calo della tensione di alimentazione) assicurano un ulteriore livello di protezione;
- la struttura della porta dovrebbe consentire di rilevare condizioni di interruzioni e di corto circuito senza nessun onere in termini di programmazione. Ciascun pin di I/O dovrebbe essere indirizzabile contemporaneamente come ingresso e come uscita, per garantire la massima semplicità ed efficienza nelle operazioni di diagnostica.

Logica di I/O

La logica della porta di I/O contiene due registri associati con ciascuna porta: un registro dei dati e un registro di configurazione. Ciascun pin della porta di I/O può essere configurato singolarmente sotto il controllo software, come riportato nella figura 3.
Le caratteristiche di self testing possono anche essere utilizzate per leggere in maniera affidabile lo stato di ingressi esterni (come ad esempio la rilevazione di un evento esterno come la chiusura di un commutatore o l'innesco di un'operazione mediante un pulsante). Per esempio, quando un pulsante non viene premuto, il pin della porta di I/O dovrebbe leggere un "1" se il pin viene portato nello stato logico alto dal resistore di pull up presente a bordo. Ma se il pin legge una "0" non è possibile sapere se questo "0" è imputabile al fatto che il pulsante è premuto oppure a un cortocircuito. A questo punto si pone il problema di determinare in maniera affidabile quale situazione si è verificata.

La procedura di seguito esposta può essere utile allo scopo:
1 - mettere in modalità tri-state il driver per il pin e leggere lo stato sul pin. La lettura di uno "0" significa che il pin è portato a un livello "0" logico o che il pin è in modalità tri-state;
2 - modificare il driver in modo da attenuare l'azione del resistore di pull up e leggere ancora lo stato del pin. Nel caso si legga un "1", il pin è in modalità tri-state, ovvero non c'e cortocircuito a massa. Mediante la verifica di questo stato prima che il pulsante venga premuto, il software può controllare che questo ingresso non è cortocircuitato a massa;
3 - porre in modalità tri-state il driver per ritornare alle condizioni di setup originali e leggere di nuovo lo stato del pin;
4 - nel caso venga letto un "1", modificare lo stato del pin ponendolo a livello logico basso per un breve periodo di tempo, quindi porlo in modalità tri-state ed effettuare la lettura Se viene letto uno "0", il pin è in modalità tri-state, mentre nel caso venga letto un "1", questo corrisponde a un "1" logico.

Timer di reset del watchdog

Un reset innescato dal watchdog fornisce una "verifica di integrità" per quanto concerne il software. Sebbene parecchie architetture di microcontrollori rendano disponibile un timer di watchdog, questo differisce in maniera sensibile a livello di implementazione e di funzionalità offerte. Alcune caratteristiche che i timer di watchdog dovrebbero possedere vengono riassunte di seguito:
- uscita del watchdog - l'uscita del timer dovrebbe azionare un ingresso di reset per la CPU. Alcuni timer di watchdog attivano un interrupt NMI che può risultare inefficace se i disturbi a livello software impediscono di soddisfare le richieste di interrupt;
- blocco del registro - deve risultare possibile bloccare i registri di controllo per il timer di watchdog dopo che questo è stato impostato. Ciò impedisce la disabilitazione del timer da parte di software "errante" (errant software);
- reset codificato - quando il software ripristina il timer di watchdog, tale operazione dovrebbe venire effettuata scrivendo un codice speciale nel registro del timer. Il codice dovrebbe essere un valore che il software errante non sia in grado di scrivere per proprio conto (come 0xFF);
- finestra di reset - il timer dovrebbe supportare una finestra di refresh, in modo da poter invocare un reset quando il timer esegue tale operazione troppo presto o troppo tardi;
- numero troppo elevato di reset di watchdog - la logica di watchdog dovrebbe rilevare una condizione nella quale il software si blocca in un loop che comprende il codice che resetta il watchdog timer. Se questa condizione non attiva il watchdog, il software rimane bloccato nel loop. Sono peraltro disponibili un certo numero di circuiti di supervisione esterni per complementare il circuito di watchdog presente on chip oppure per fornire questa funzione quando si utilizza una CPU che non disponga di tale funzionalità (come ad esempio un processore x86).
Tali dispositivi integrano timer di watchdog e funzionano gestendo la linea di reset del processore.

BIST - Built-In Self test

Tale funzionalità permette di effettuare verifiche del sistema durante la fase di power on reset (POR), ovvero quando viene effettuato il reset in presenza della tensione di alimentazione e può essere chiamata in altre occasioni, come ad esempio nel corso di un reset attivato dall'utente o durante il tempo di idle (ovvero di inattività) del sistema. Essa può essere una semplice funzione che si preoccupa di eseguire verifiche di quelle risorse che hanno un'elevata probabilità di subire un guasto (memoria flash, periferiche di natura elettromeccanica), come pure essere capace di effettuare un collaudo completo che prende in considerazione il set di istruzioni della CPU, nonché eseguire test pattern di RAM e così via.
Nella figura 4 viene riportato lo schema a blocchi di tipo generale di un sistema basato su un microcontrollore. Il microcontrollore a sua volta risulta composto da CPU, circuito per la gestione degli interrupt, clock, memoria, percorso dati interno, circuiti per le comunicazioni esterne, I/O, dispositivi di monitoraggio e da numerosi altri blocchi. Utilizzando come linea guida lo standard UL-1998, nella tabella 1 vengono riportati i potenziali guasti ed errori di ciascun blocco e le soluzioni di tipo hardware/software più idonee.

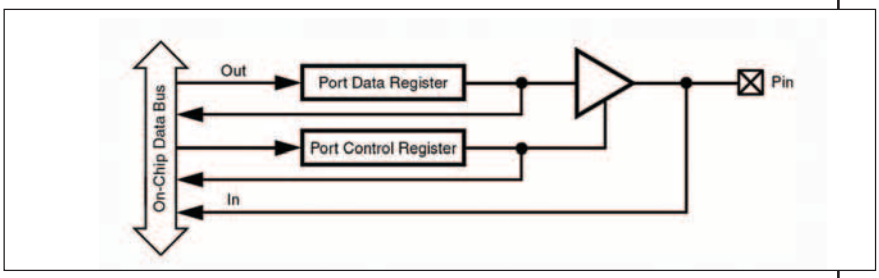

Fig. 3 - Logica della porta di I/O secondo lo standard UL 1998

Interferenze elettromagnetiche (EMI)

All'aumentare della diffusione delle tecnologie wireless, cresce di pari passo la possibilità che si verifichino interferenze tra i vari dispositivi. Gli enti normatori si sono occupati di questo problema che assume un'importanza sempre più rilevante. Ad esempio la FCC (Federal Communications Commission) ha fatto parecchie pressioni sui produttori di

apparecchiature elettroniche affinché riducano l'inquinamento dello spettro elettromagnetico a causa dell'incremento del rischio che si verifichino interferenze tra i dispositivi. A causa della crescente diffusione di microcontrollori embedded in sistemi di tipo "safety-critical" per applicazioni automobilistiche, aerospaziali e medicali, non è più accettabile che i dispositivi consumer non siano sottoposti a regole severe nei confronti delle interferenze elettromagnetiche. I progettisti devono quindi adottare le metodologie idonee a soddisfare, se non addirittura ad anticipare, le regole imposte dagli enti formatori per ridurre i livelli delle emissioni elettromagnetiche indesiderate. Sebbene parecchie tecniche atte alla riduzione delle emissioni riguardano la componente hardware, le metodologie software possono fornire un valido contributo. Ad esempio è possibile sviluppare software atto a minimizzare la commutazione interna oppure adottare schemi di gestione della potenza sia per la CPU sia per le periferiche (ad esempio attivando modalità a bassa potenza per ridurre la frequenza di clock della CPU, oppure eliminando le periferiche inattive).
Gli enti normatori, sia in Europa sia negli Stati Uniti, e numerose organizzazioni industriali hanno messo a punto numerosi standard EMC. Il software applicativo che viene eseguito dal microcontrollore deve essere incluso in questi test per assicurare l'assenza di interruzione nel corso dell'esecuzione delle operazioni. Tali test comprendono: emissioni irradiate involontarie, emissioni condotte involontarie; scariche elettrostatiche; campi irradiati; fenomeni transitori veloci di natura elettrica; emissioni RF condotte; interruzioni e cadute di tensione.

Errori software nel corso dell'implementazione

Gli errori di implementazione sono probabilmente la classe di difetti software con cui i programmatori hanno più familiarità. In altre parole, la specifica è corretta ma la maniera con cui è stata realizzata risulta errata. L'adozione di metodologie per lo sviluppo di prodotti affidabili permette di evitare tali errori minimizzando la possibilità che il prodotto si comporti in modo non specificato dal progettista. Di seguito vengono riportati alcuni esempi di tali errori:
- rischi legati alla perdita di potenza - nel momento in cui si verifica un arresto dell'alimentazione o una condizione di sottotensione in un sistema basato su micro, sussiste il rischio che questo entri in uno stato arbitrario. Poiché la CPU non arriva in questo stato attraverso un percorso specificato dal progettista software, i segnali di controllo generati dal microcontrollore potrebbero portare a una condizione di funzionamento errato o addirittura di pericolo. La soluzione in questo caso potrebbe essere rappresentata da un circuito per il rilevamento di fenomeni di brown out che permette di riportare il microcontrollore in uno stato conosciuto quando il valore della tensione di alimentazione non rispetta le specifiche;
- rischi legati all'esecuzione del programma - la crescente diffusione di dispositivi portatili e palmari che sfruttano lo spettro RF si traduce in un aumento del rischio di alterazione del funzionamento del microcontrollore a causa di eventi transitori. Il problema assume una maggiore rilevanza nel caso si influenzi l'esecuzione del programma, perché ciò potrebbe provocare l'entrata in un loop del software che impedisce l'adozione di procedure di ripristino. La difesa più efficiente in questo caso è rappresentata dall'uso di un timer di watchdog;
- overflow dello stack - una causa frequente di malfunzionamento del software è l'incapacità di gestire un numero imprevisto di eventi di interrupt. Ciò provoca un overflow dello stack, perché troppi contesti di interrupt sono salvati nello spazio dello stack. Nelle architetture dei microprocessori che prevedono la gestione della memoria, questo fenomeno è rilevabile sotto forma di violazione dei limiti del segmento dello stack o come tentativo di scrivere su una pagina protetta da scrittura (nel caso una pagina protetta da scrittura è posta alla fine dello spazio dello stack);
- verifica dell'integrità dei dati - sebbene un timer di watchdog fornisca una primitiva indicazione della regolarità del flusso di esecuzione, può accadere che l'immissione di dati privi di logica provochi l'emissione di dati altrettanto illogici. Frequenti controlli dell'integrità dei dati che prevedono l'impiego del timer di watchdog e la verifica dell'overflow dello stack possono essere utilmente impiegati per rilevare comportamenti anomali del programma. Per l'individuazione di dati errati si dovrebbe far ricorso alle verifiche degli indici (bounds check) dei dati.

Errori software imputabili al progetto

Errori di questo tipo, molto insidiosi, sono imputabili a una corretta implementazione di una specifica errata. Poiché l'errore risiede nella specifica, e non nella sua realizzazione pratica, le metodologie di sviluppo software solo raramente sono in grado di prevenire tali errori. Di seguito vengono riportati alcuni esempi:
- variazione di stato silenziose - alcuni eventi mutano in maniera silenziosa stati o modalità critiche;
- risposta imprevista a ingressi indefiniti - il comportamento di un dispositivo conseguente a un ingresso impostato dall'utente dovrebbe essere definito per ogni possibile ingresso. Se l'interfaccia utente non è stata progettata per gestire un determinato ingresso, dovrebbe impedire la visualizzazione dell'indicazione che l'ingresso è stato accettato;
- lunghezza della chiave di sicurezza - parecchie messaggerie telefoniche possono funzionare in modo remoto, ovvero consentono al chiamante di ascoltare e cancellare i messaggi dal dispositivo. Tali apparecchiature sono solitamente protetti da una password a tre cifre (o anche inferiore) che può essere facilmente violata. Le portiere di alcune automobili controllate mediante tecniche a radiofrequenza sono protette da chiavi di sicurezza di lunghezza inadeguata: con un'apposita apparecchiatura da scasso è possibile far girare tutte le possibili combinazioni finché la portiera non si apre. Il livello di sicu-

-Tabella 1 - Copertura dei guasti e degli errori a livello hardware e software

Componente	Errore/guasto	Azione correttiva
CPU/registri	Stuck-At	Test funzionale eseguito prima del funzionamento o periodicamente durante il funzionamento normale. Il software di test risiede nella memoria di programma ed esegue a) validazione del set di istruzioni b) manipolazione dei bit sui contenuti della memoria c) operazioni di scrittura e verifica sui registri funzione
	Guasto DC	Si devono verificare funzioni del registro, indirizzamento della memoria e movimento dei dati. È possibile eseguire altre operazioni come il self test della memoria
Decodifica/esecuzione delle istruzioni	Errori nella decodifica/esecuzione	Una routine di test, in maniera periodica o prima dell'esecuzione delle routine critiche, può effettuare test sul set di istruzioni
Registro istruzioni/indirizzamento	Stuck-at/guasto DC	È possibile eseguire il collaudo funzionale mediante confronto reciproco. Il monitoraggio logico del programma viene eseguito mediante hardware di watchdog presente sul chip
Percorsi dati	Guasto DC	In molti casi non è possibile l'accesso al bus dei dati interno: la funzione di read back delle porte nel corso del pilotaggio dei dati consente la verifica dei percorsi dati interni
Esecuzione e gestione degli interrupt troppo frequenti	Nessun interrupt o interrupt	La capacità di assegnare una priorità più elevata alle routine correlate alla sicurezza e l'elaborazione in tempi rapidi degli interrupt possono rappresentare un valido ausilio. Nel caso una routine collegata alla sicurezza sia pilotata da interrupt e non viene immessa (o viene immessa poche volte), il timer di watchdog rileva questa condizine
Memoria Memoria non volatile Memoria volatile	Errori su bit singoli Errori su bit singoli e doppi Guasto DC e crosslink dinamico	Il set di istruzioni dovrebbe comprendere un'istruzione per effettuare una lettura efficiente e veloce dalla memo ria di programma. L'istruzione dovrebbe semplificare l'implementazione dei calcoli relativi alla somma di controllo e il CRC in fase di esecuzione. I dispositivi in architettura Harvard (con la memoria dei dati separa da quella di programma) rendono disponibile questa funzionalità. Le routine per il test di memoria possono eseguire test di memoria statica: è inoltre possibile l'integrazione di self test periodici.
Dati interni, percorsi dati interni/indirizzamento	Stuck-At/guasto DC	Il controllo CRC periodico può essere effettuato sia sulla Il controllo CRC periodico può essere effettuato sia sulla memoria dati sia sulla memoria di programma. Le locazioni della memoria dati possono essere verificate per quanto concerne l'aliasing. I percorsi dati, i dati interni e l'indirizzamento possono essere verificati simultaneamente
Input/output		I pin di I/O dovrebbero essere configurabili come ingressi o uscite. Ciò permette la lettura dello stato dei pin della porta mentre le porte sono pilotate come uscite. Il software è in grado di rilevare con facilità interruzioni e corti
Clock	Frequenza inesatta	Per i prodotti ideali che derivano l'alimentazione da una linea AC, la frequenza della linea può essere usata come sorgente di temporizzazione per porre il sistema nello stato PRA (Potential Risks Addressed)

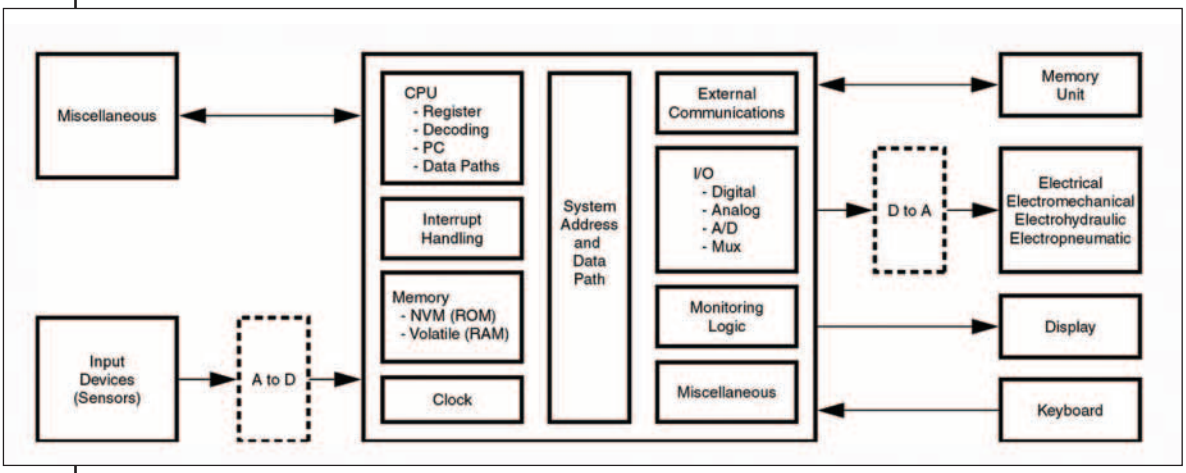

Fig. 4 - Schema a blocchi di un sistema basato su microcontrollore

rezza può essere innalzato utilizzando una chiave che preveda un numero maggiore di cifre, aumentando i tempi di ciclo per il collaudo di una chiave oppure forzando un periodo di blocco temporaneo dopo un certo numero di tentativi andati a vuoto;
- algoritmi di sicurezza - nel tempo sono state messe a punto numerose tecniche per recuperare chiavi segrete e password celate nella memoria dei microcontrollori che si trovano sulle smart card o altri dispositivi di autenticazione. Le tecniche sono le più svariate: da quelle a basso livello tecnologico, come ad esempio il disturbo del processore mediante appositi segnali di clock, transitori sull'alimentazione e interferenze elettromagnetiche fino ad arrivare a tecniche sofisticate che prevedono l'accesso agli stati interni del chip mediante fasci laser e microsonde. Per recuperare dati critici si è anche fatto ricorso a all'analisi delle caratteristiche del funzionamento del dispositivo, come ad esempio i consumi. Le contromisure che è possibile adottare prevedono l'impiego di algoritmi di cifratura concepiti per eliminare la dipendenza dai dati di tutte le caratteristiche misurabili esternamente. Nel caso il microcontrollore embedded è chiamato a svolgere funzioni critiche dal punto di vista della sicurezza, deve disporre di algoritmi di sicurezza adeguati all'importanza della funzione oggetto della protezione;
- modalità di sicurezza - nel caso un dato di ingresso indichi un guasto in un sensore (o qualsiasi altra condizione che possa portare a un funzionamento non corretto) il software dovrebbe essere in grado di portare automaticamente il sistema in uno stato che comprometta il meno possibile la sicurezza dell'utente. Nel momento in cui si verifica un guasto in un sensore, il sistema deve quindi portarsi in uno stato di sicurezza "ragionevole" (secondo quanto stabilito dalla specifica PRA - Potential Risks Addressed State dello standard UL 1998);
- pericoli derivati dall'aggiornamento del software - in passato, le spese associate alla sostituzione dei dispositivi o al loro rinvio alla fabbrica per la riprogrammazione hanno contribuito a rendere quasi proibitivi i costi legati all'aggiornamento del software. Il sempre più ampio utilizzo di memorie flash e di tecniche di programmazione in-system ha permesso di semplificare e rendere più economiche le operazioni di aggiornamento. Una tecnica sempre più diffusa prevede l'impiego del segnalamento in banda (in band signaling) per scaricare gli aggiornamenti software. Per esempio, un lettore CD potrebbe venire aggiornato attraverso i codici celati in un CD musicale, così come un telefono cellulare potrebbe essere aggiornato mediante una semplice chiamata al costruttore. L'impiego del segnalamento in banda per effettuare aggiornamenti software dovrebbe soddisfare i seguenti requisiti: a) prima dell'installazione deve essere stato ricevuto l'intero aggiornamento; b) l'aggiornamento deve essere legittimato (per evitare i pericoli derivanti dall'azione di eventuali hacker); c) l'aggiornamento deve essere più recente rispetto alla versione installata; d) l'aggiornamento deve essere idoneo per il modello e il tipo di dispositivo sul quale viene installato.

Manutenzione del software

Successivamente al lancio di un prodotto, il team che si occupa dello sviluppo software non deve presumere di aver concluso il proprio lavoro. Infatti potrebbero esserci difetti che non si manifestano finché non sono in circolazione un numero rilevante di unità. Quando l'adozione di procedure di sviluppo affidabili e di collaudo del prodotto non riesce a evitare il rilascio di un prodotto difettoso, il ricorso a opportune procedure può risultare utile per gestire le conseguenze:
- modalità diagnostica - i prodotti che sono in grado di rilevare in maniera autonoma i propri difetti potrebbero essere capaci di indicare il tipo di problema incontrato, un elemento utile in fase di riparazione o di analisi dei guasti. Non va comunque dimenticato che le funzioni di diagnostica potrebbero a loro volta essere causa del guasto di un prodotto.
- logging - quando la funzione BIST rileva un guasto, questo dovrebbe essere registrato in modo che il personale addetto possa avere la visibilità dei motivi del guasto del prodotto. Oltre a ciò, gli elementi principali dell'esecuzione del pro-

Riferimenti

DIN 0801	**"Principles for Computers in Safety-Related Systems"**
DIN 19250	**"Basic Safety Evaluation of Measuring and Control Protective Equipment"**
IEC 61508	**"Functional Safety of Electrical/Electronic/Programmable Electronic Safety Related Systems"**
UL 1998	**"The Standard for Software in Programmable Components"**

gramma (eventi di rest, interrupt e così via) possono risultare utili nella diagnosi dei guasti in fase di esecuzione che si verificano senza essere rivelati dal circuito BIST. È possibile allocare un buffer circolare in una porzione di memoria flash inutilizzata per salvare una descrizione del funzionamento del software appena prima che si verifichi il guasto;
- analisi dei guasti - il guasto di un prodotto rappresenta un serio problema per i produttori. In passato questi ultimi hanno sfruttato la velocità di obsolescenza di un prodotto per ignorare i guasti sul campo dei dispositivi più datati. Ciò a lungo andare provoca una perdita di immagine. L'analisi dei guasti deve essere vista alla stregua di un'opportunità in quanto fornisce preziose informazioni riguardo a componenti inaffidabili, metodologie di produzione fuori controllo e altre fonti di guasto dei prodotti.

Caratteristiche di affidabilità del microcontrollore

Anche nel momento in cui un team di progettisti non tiene conto del problema dell'affidabilità, la scelta di un microcontrollore che supporta per via hardware meccanismi di affidabilità offre un numero maggiore di opzioni. Tra i microcontrollori che integrano funzionalità hardware e software capaci di soddisfare le esigenze di affidabilità si possono segnalare i dispositivi COP8TA e CP3BT23/26 di National Semiconductor.
Tra le caratteristiche di rilievo di COP8TA si possono segnalare:
- architettura hardware modificata che utilizza blocchi di memoria separati per i dati e le istruzioni in modo da impedire l'esecuzione di dati come istruzioni, la modifica involontaria del codice e condizioni di errori non prevedibili;
- hardware di watchdog flessibile e circuito di monitor del clock che assicurano il rilevamento indipendente di loop infiniti e di salti al di fuori dei limiti. Il limite superiore della logica di watchdog può essere impostato per rilevare gli effetti di un numero troppo limitato di ingressi nelle routine di sicurezza. Il limite inferiore del watchdog garantisce la protezione contro il blocco di una routine in un ciclo di aggiornamento;
- tutte le risorse interne critiche del dispositivo (come gli stati di I/O e i registri di controllo e di stato) vengono inizializzate a partire da uno stato noto;
- la funzione di arresto della CPU è impostata in fase di reset e, una volta impostata, questa condizione, non può essere modificata;
- porte di I/O flessibili che consentono il monitoraggio dei pin senza particolari oneri in termini di programmazione. Ciascuna porta di I/O può essere indirizzata contemporaneamente come ingresso e come uscita;
- gli accessi alla memoria di programma sono sicuri per progetto;
- presenza di tabella di ricerca, somme di controllo e CRC per ridurre l'overhead software;
- ridotto livello di emissioni Emi.

I processori di connettività CP3BT23/26 sono membri di una famiglia di prodotti già utilizzati in volumi in applicazioni critiche in ambito automotive. Le versioni per applicazioni Bluetooth di questi microcontrollori integrano già caratteristiche adatte a soddisfare le esigenze di questo mercato, come ad esempio certificazione QS9000, intervallo di temperatura di funzionamento tra - 40 e +85 °C; tasso di difettosità sul campo di 1 PPM; disponibilità di versione operanti fino a + 125 °C. Oltre a ciò i produttori operanti nel settore automobilistico hanno esigenze specifiche in termini di collaudo e testabilità del microcontrollore che comprendono:
- collaudo di scansione - utile per assicurare il funzionamento di tutti i transistor;
- collaudo di sollecitazione elettrica - il dispositivo viene collaudato a una tensione superiore a quella massima di funzionamento ma inferiore alla tensione di breakdown;
- BIST - sfrutta il codice che rappresenta differenti funzioni on chip e test specifici;
- Maverick part testing - eliminazione dei dispositivi che si trovano in prossimità dei limiti massimi di variazione prestabiliti.

In definitiva, l'adozione di procedure idonee, l'utilizzo di un programma di collaudo completo e la manutenzione sono i tre elementi base di una strategia che ha come obiettivo la progettazione di prodotti affidabili.

National Semiconductor
readerservice.it n. 28