**Progetto di reti Logiche**

Windowed watchdog counter programmabile

Simone Lucca

# Introduzione

Nel contesto dei sistemi embedded, l’affidabilità e la stabilità dell’esecuzione del software sono di fondamentale importanza. Per garantire il corretto funzionamento di un microprocessore esterno e reagire ad eventuali errori software, è stato progettato un windowed watchdog counter.

Immagine che contiene diagramma, linea, schermata, testo

Descrizione generata automaticamenteQuesto dispositivo si basa sulla conoscenza dell’intervallo temporale in cui ci si aspetta che il ciclo di istruzioni eseguito dal microcontrollore termini in modo corretto. Questo intervallo statistico viene determinato precedentemente tramite test o previsioni di diverso genere.

Il dispositivo implementa un prescaler capace di divere il segnale di clock per una potenza del due tra 1 e 1024.

Per impostare le soglie si utilizza il segnale di “data”, che contiene il valore della soglia, unito con il segnale di “command” che contiene l’informazione riguardante quale dato si sta inserendo. Questo viene utilizzato anche per impostare il valore di prescaling.

Una volta portato a “1” il segnale di start, il watchdog non è più in grado di impostare i valori di soglia e inizia a contare gli istanti di tempo. Il microcontrollore deve essere in grado di segnalare al watchdog che la routine in esecuzione è terminata, lo fa tramite il segnale di clear.

Di seguito è riportato il comportamento atteso del dispositivo per l’arrivo del segnale di clear in diversi istanti temporali.

Immagine che contiene linea, diagramma, Diagramma, Parallelo

Descrizione generata automaticamente

Il segnale di NMI (Non-Maskable Interrupt) viene utilizzato come avviso per segnalare al microcontrollore che la routine in esecuzione sta richiedendo troppo tempo e che il segnale di reset sarà presto attivato.

Una volta attivato il segnale di reset, è possibile disattivarlo dal microcontrollore impostando il segnale di start ad un livello alto. Ciò non modificherebbe i valori di soglia preimpostati e consentirebbe al microcontrollore di continuare l'esecuzione normalmente.

Come alternativa, è possibile reimpostare il watchdog attraverso l'attivazione del segnale "rst" in ingresso. Questa operazione svuoterebbe i registri e riporterebbe il dispositivo nella fase di setup.

# Specifica

Il windowed watchdog counter è stato progettato con tre soglie differenti:

* Soglia minima: Nel caso in cui la routine termini prima del suo raggiungimento, viene generato un segnale di reset destinato al controllore, poiché l’esecuzione è avvenuta in modo troppo rapido
* Soglia massima: Nel caso il programma impieghi un’eccessiva quantità di tempo per terminare, il watchdog genera un segnale di reset in uscita
* Soglia di NMI: Il microcontrollore ha la necessità di essere notificato qualora il tempo a disposizione per la terminazione del task stia per terminare. Per questa ragione viene mandato un segnale di non maskable interrupt una volta raggiunta la soglia in questione.

Il dispositivo è dotato di un segnale di clock condiviso con il microcontrollore, questo rende i valori temporali associati ai valori di soglia strettamente legati alla velocità di clock del processore. Per questa ragione è stato necessario introdurre un circuito divisore di clock programmabile con valori che sono tutte le potenze del 2 da 1 a 1024, in maniera tale da aumentare la flessibilità del sistema.

Configurazione:

Precedentemente all’avvio del dispositivo nasce la necessità di impostare le soglie citate precedentemente, si suppone di conoscere:

* frequenza di clock del microcontrollore fck
* tempo di soglia minimo Tmin
* tempo di soglia massimo Tmax
* tempo di soglia di nmi Tnmi

Questo è sufficiente per configurare il watchdog counter.

Il circuito trattato contiene un contatore a 16 bit. Questo significa che il conteggio massimo raggiungibile è di . Il contatore si attiva su ogni fronte di salita del clock in uscita dal prescaler.

Decisione del fattore di divisione:

Periodo

Si cerca un valore n (esponente della potenza del 2 che divide il segnale di clock) tale per cui

Risolvendo per n

Introducendo l’approssimazione

Essendo che , ci saranno valori fuori portata per questo dispositivo.

Nello specifico:

Per quanto riguarda il Tmin si ha il limite minimo del valore 0, che significa rimuovere completamente la soglia di minimo.

Impostazione dei valori di soglia:

Per impostare i valori di soglia bisogna comprendere che i valori da impostare non corrispondono direttamente al tempo desiderato. Ciò è dovuto al fatto che il contatore interno al watchdog viene incrementato ogni colpo di clock, non ogni unità temporale standard.

Per questa ragione si deve impostare un valore di 16 bit tale che:

In cui:

* X: valore da 16 bit da impostare nel watchdog
* Ti: Tempo di soglia scelto

Non è sempre possibile avere il tempo esatto a causa della natura discreta del contatore, per questa ragione nella formula è stato inserito il simbolo “

**Utilizzo:**

Per attivare il dispositivo è necessario aver completato la fase di configurazione, quindi rendere alto il segnale di start. Questo porterà il watchdog in uno stato in cui non sarà più possibile ritornare alla fase di configurazione.

Nel caso si avesse bisogno di modificare i valori è possibile farlo rendendo il segnale di reset in ingresso ad un valore alto. Questo riporterà il dispositivo allo stato di configurazione.

In seguito, il microcontrollore dovrà essere in grado di mandare in uscita un segnale di clear per notificare il watchdog della terminazione del programma, dopo questo segnale il conteggio ricomincerà.

Inoltre, è necessario configurare il watchdog in modo tale che Xmin < Xnmi < Xmax. Questo perché, senza il rispetto di questo vincolo i valori di soglia perdono completamente il loro significato

## Interfaccia del sistema

Il dispositivo è dotato dei seguenti ingressi:

* Command: segnale composto da 2 bit utilizzato per selezionare quale tipo di dato memorizzare, i valori possibili sono:
* 00 -> Memorizza il valore di prescaling
* 01 -> Memorizza il valore di soglia minima
* 10 -> Memorizza il valore di soglia nmi
* 11 -> Memorizza il valore di soglia massima
* Data: segnale da 16 bit che contiene il dato da memorizzare. Si nota che la memoria per il prescaling è di soli 4 bit. Pertanto, nel caso in cui il comando sia impostato su “00”, vengono solo utilizzati i 4 bit meno significativi del vettore data.
* Start: segnale utilizzato per passare dalla modalità di configurazione alla modalità di lavoro, viene anche utilizzato per riportare il watchdog nello stato di lavoro in caso il segnale di reset in uscita si dovesse attivare.
* Clr (clear): segnale dedicato al ripristino del conteggio, viene utilizzato per segnalare la terminazione del ciclo corrente di esecuzione
* Clock: segnale di clock
* Rst (reset): ripristina il watchdog in maniera tale da essere in grado di modificare i valori di soglia e di prescaling

Le uscite sono le seguenti:

* Reset: questo segnale viene attivato nel caso la terminazione avvenisse al di fuori del range determinato dalla soglia minima e dalla soglia massima. Rimane alto finché non viene ripristinato dal microcontrollore attraverso l’attivazione del segnale di start.
* NMI: questo segnale viene attivato al raggiungimento della soglia di nmi e disattivato con il proseguimento del conteggio

In seguito, è riportato il diagramma temporale riguardante la fase di setup:

Immagine che contiene testo, diagramma, schermata, linea

Descrizione generata automaticamente

Nota: il valore di data è stato riportato in esadecimale mentre command in binario

I valori sono memorizzati all’interno di registri nel watchdog sul fronte di salita del clock, per questa ragione è necessario che i valori di data e command siano già a regime nell’istante di lettura.

L’ordine temporale usato per impostare il valore delle soglie di max, min e nmi è arbitrario. Esiste la possibilità di modificare i valori impostati ma non dopo aver impostato il valore del segnale di start a ‘1’.

## Architettura del sistema

Si riporta lo schematico ad alto livello del circuito, sono stati omessi segnali in ingresso più semplici come il clock per rendere più chiaro il funzionamento ad alto livello.

Immagine che contiene testo, diagramma, Piano, Parallelo

Descrizione generata automaticamente

Register File:  
Questo modulo si occupa di memorizzare le informazioni impostate dal microcontrollore nella fase di setup. Possiede un segnale di clock enable in ingresso per impedire al register file di memorizzare dati quando il segnale di start viene attivato.

Counter 16 bits:

Il contatore è necessario per scandire le unità di tempo, possiede in ingresso un segnale di clear per reimpostare il conteggio al valore di 0 e un segnale di count enable in grado di fermare il conteggio (0) o farlo proseguire (1). Il prescaler utilizza proprio questo segnale per gestire la divisione temporale.

Prescaler:

Utilizzando il valore di 4 bit inserito dal microcontrollore si coordina con il contatore principale per far in modo che il conteggio prosegua solo dopo il tempo desiderato, questo produce l’effetto di una divisione temporale.

Time Comparator:

Si ha la necessità di confrontare il conteggio attuale con i valori nei registri per comprendere se le soglie sono state raggiunte. Questo componente pone una delle sue uscite ad un valore alto nel caso in cui il vettore “count value” sia uguale al vettore corrispondente tra i tre vettori in ingresso.

Output Reset:

Questo modulo gestisce l’attivazione del segnale di reset in base all’arrivo (o no) del segnale di clear, è capace di far ritornare il segnale di reset a 0 se riceve il segnale di start in ingresso. In questo caso non nasce il bisogno di reimpostare il register file, questo rimarrà con gli stessi valori precedenti al reset.

## Contatore principale (16 bit)

Per contare le unità di tempo è necessario l’utilizzo di un contatore.

Ingressi:

* Clr: reimposta il contatore al valore 0
* Rst: imposta il contatore al valore 0 all’accensione
* CE: clock enable, rende il contatore sensibile al fronte di salita del clock (se 1)

Uscite:

* Count: conteggio attuale (vettore da 16 bit)

Struttura:

Il contatore è stato implementato utilizzando bistabili di tipo D, sensibili al fronte di salita.

Di seguito è riportato il circuito fermandosi al quarto bit.

Immagine che contiene schizzo, diagramma, Disegno tecnico, Piano

Descrizione generata automaticamente

Si è iniziata la progettazione con un contatore binario naturale, per poi aggiungere la funzionalità di clear.

Nota: la funzione in ingresso ad ogni bistabile è la “xor” tra l’uscita del flip flop e la “and” tra tutte le uscite dei bistabili precedenti (bit meno significativi).

Sarebbe stato possibile utilizzare la versione più economica in termini di area, utilizzando le porte “and” su cui si era precedentemente calcolato il prodotto dei bit in uscita precedenti, questo avrebbe comportato un ritardo circa 4-5 volte superiore. Non avendo specifiche riguardanti l’area o la frequenza è stato deciso di utilizzare la versione più veloce, dato che il guadagno è più alto della perdita.

Si nota che il segnale “ce” è in grado di fermare il conteggio, se posto a 0.

## Prescaler

Di seguito è riportato lo schematico del prescaler realizzato.

Immagine che contiene diagramma, Disegno tecnico, linea, Piano

Descrizione generata automaticamente

Concetto base:

Si parte descrivendo il componente su cui si basa il prescaler realizzato, ipotizzando uno scenario in assenza della possibilità di scegliere il fattore per il quale si vuole dividere il clock, per poi passare al caso generale implementato.

Si consideri un vettore di bit di dimensione generica “000 … 0”, se si decide di assegnare “1” al bit in posizione i-esima (la posizione è contata considerando il bit meno significativo come posizione 0) si ottiene il valore decimale di . Ciò significa che è possibile incrementare il vettore per -1 volte prima che il bit i-esimo diventi 1. Questa caratteristica può essere sfruttata per dividere il segnale di clock utilizzando un contatore che incrementa di 1 il vettore ogni volta che il clock originale presenta un segnale di fronte. Se serve dividere il segnale di clock per sarà necessario un contatore a i+1 bit.

Implementazione effettiva:

Il prescaler deve essere in grado di dividere per una potenza del 2 scelta dall’utente, fino a 1024. Per questo motivo è necessario implementare il contatore per il caso pessimo (11 bit) e, per gli altri casi, utilizzare solo porzioni di questo contatore.

A questo scopo è stato inserito un multiplexer che utilizza i 4 bit preimpostati dal microcontrollore come selettore. Questo porta effettivamente ad utilizzare solo la porzione di contatore fino al bit i-esimo. Nel caso il microcontrollore inserisca un valore > 10, si utilizza comunque 10 come selettore.

A questo punto si avrebbe un segnale in uscita che è pari al segnale di clock diviso per il fattore richiesto. Il problema è che il segnale in uscita non è esente da ritardi di propagazione. Per questa ragione, non è corretto utilizzarlo come se fosse un normale segnale di clock. Si procede a risolvere questo problema utilizzando il clock enable del contatore principale.

Viene riportato un esempio di divisione per 2.

Immagine che contiene Rettangolo, linea, diagramma, schermata

Descrizione generata automaticamente

Nota: Si è lasciato uno spazio tra il fronte di salita del clock e l’attivazione (o disattivazione) del clock enable. Questo rappresenta i ritardi di propagazione nel prescaler.

Il contatore principale adesso prosegue il conteggio una volta ogni due fronti di salita del clock principale.

Un altro problema notevole si presenta quando la divisione è > 2.

Immagine che contiene diagramma, linea, Rettangolo, Piano

Descrizione generata automaticamente

In questo caso il clock enable farà in modo che il contatore prosegua 2 volte e poi si fermi per altri 2 cicli, producendo un risultato errato.

La soluzione implementata fa in modo che il contatore ritorni al valore di “0…01” ogni volta che l’uscita diventa alta. Il vettore in uscita al contatore va resettato a 1 a causa del conteggio perso nell’impostare il clock enable.

Immagine che contiene diagramma, linea, Rettangolo, schermata

Descrizione generata automaticamente

Adesso si nota che il conteggio avviene una volta ogni 4 cicli di clock, come si desiderava.

Contatore 11 bit:

Il contatore utilizzato internamente al prescaler è stato realizzato analogamente al contatore principale, l’unica differenza sostanziale è che il segnale “clr” reimposta il conteggio al valore 1. Questo per compensare il ciclo di clock perso per il clock enable. Per questa ragione è stato necessario modificare la logica in ingresso al bistabile con uscita il bit meno significativo, introducendo una porta “or” al posto della porta “and”.

## Register file

Il microcontrollore deve essere in grado di impostare i tre valori di soglia e il valore di prescaling. Queste informazioni devono rimanere invariate per il completo periodo di operazione del watchdog e devono essere accedute continuamente da altri componenti interni al watchdog counter. Nasce quindi la necessità di memorizzare le informazioni in una memoria veloce.

Si utilizzano, per questa ragione, quattro registri parallelo-parallelo. Tre di questi sono dedicati ai valori di soglia (16 bit), mentre il rimanente è dedicato a memorizzare il valore di prescaling (4 bit).

Immagine che contiene diagramma, Disegno tecnico, Piano, schematico

Descrizione generata automaticamente

Per essere in grado di scegliere il registro su cui scrivere il dato, è stato introdotto un decoder, il quale alza l’unica delle sue uscite corrispondente al valore dell’ingresso “command”. Inoltre, per essere in grado di fermare la scrittura su ogni registro (questo avviene quando viene attivato il segnale di start), si è deciso di usare un segnale di WrReg (write register) che, se posto a 0, impedisce la scrittura sui registri.

Nota: una volta che il watchdog riceve in ingresso il segnale di start alto, il watchdog salva in memoria l’avvenimento di questo evento (segnale “started”). Questo segnale è legato a WrReg con una relazione di negazione. Per maggiore chiarezza è stato omesso il segnale di rst che imposta i registri a 0, va utilizzato solo all’accensione.

Registri:

I registri implementati sono stati progettati utilizzando un’architettura parallelo-parallelo e sono sensibili al fronte di salita.

## Time Comparator

Il dispositivo ha il bisogno di vedere se il conteggio attuale ha raggiunto uno dei tre livelli memorizzati nel register file.

Questo semplice modulo funziona solo utilizzando logica combinatoria e confronta bit per bit il vettore “count” e i vettori memorizzati per verificare che siano uguali.

Nota: questo modulo utilizza tre componenti più piccoli (chiamati vector\_equals) che hanno lo scopo di confrontare due stringhe di bit e resituire in uscita ‘1’ nel caso siano uguali.

## Output Reset

Il comportamento del segnale di reset in uscita varia in base a quale delle due soglie viene raggiunta e dipende anche dall’arrivo (o no) del segnale di clear. Per questa ragione è stato necessario progettare un modulo in grado di attivare e disattivare il segnale di reset in uscita in base alle informazioni ricavate da “vector equals”, clear e start.

Ingressi:

* Rst
* Clk
* Eq\_min: ‘1’ quando la soglia minima è stata raggiunta
* Eq\_max: ‘1’ quando la soglia massima è stata raggiunta
* Clr (clear)
* Start

Uscite:

* reset

Per la progettazione del circuito si parte da una macchina di moore di base, nel senso che non si considera la soglia di massimo e il segnale di start:

Le transizioni sono caratterizzate da clr (primo bit) e min (secondo bit):

Immagine che contiene disegno, schizzo, diagramma, calligrafia

Descrizione generata automaticamente

Si nota che il circuito è stato progettato in modo tale che nell’eventuale caso il clear avvenisse insieme al raggiungimento della soglia, il segnale di reset non sarebbe attivato.

Si traduce il diagramma degli stati, dopo essersi assicurati che 3 sia il numero minimo di stati possibili, nella corrispondente tabella di stato simbolica:

Immagine che contiene calligrafia, testo, Carattere, diagramma

Descrizione generata automaticamente

Nota: r sta per reset

Risolvendo con bistabili di tipo D:

Immagine che contiene testo, calligrafia, diagramma, Carattere

Descrizione generata automaticamente

Nota: la codifica degli stati è scritta nella figura in alto a destra

Il circuito ottenuto non ha funzionalità complete, manca ancora il reset dopo il raggiungimento della soglia massima e la funzionalità aggiuntiva di ritornare allo stato iniziale nel caso avvenga la pressione del pulsante di “start” mente si ha il segnale di reset in uscita alto.

Si presenta un diagramma degli stati non formale per esprimere in maniera chiara cosa va implementato:

Immagine che contiene testo, disegno, schizzo, calligrafia

Descrizione generata automaticamente

Si procede in maniera intuitiva utilizzando le porte evidenziate in figura:

Immagine che contiene diagramma, testo, disegno, Piano

Descrizione generata automaticamente

In questo modo si raggiunge il comportamento desiderato dal componente

# Verifica

I test sono stati eseguiti prima sui singoli moduli, per assicurarsi che rispettino il comportamento richiesto, in seguito si è testato il watchdog come unione dei moduli.

La verifica avviene assicurandosi inizialmente che il circuito funzioni nel caso standard per poi esplorare ogni possibile caso particolare.

Verifica behavioral:

Utilizzata per assicurarsi che il componente rispetti le specifiche trascurando gli effetti di ritardo dovuti ai transistor e componenti elettriche interne al circuito.

Verifica post-place & route:

Il componente deve essere in grado di funzionare nonostante ritardi dovuti alle porte logiche. Questo modo di verifica permette di comprendere se il circuito sarà in grado di funzionare alle frequenze di clock desiderate per capire se è necessario diminuire i livelli di logica, migliorando le prestazioni.

## Test-bench e casi d’uso

I casi test del sistema top-level si ispirano al funzionamento reale del circuito valutando le diverse possibilità:

1. Senza prescaling:
   * Si valuta se il segnale di reset si alza al raggiungimento della soglia massima, e se rimane attivo per il tempo corretto.
   * Si valuta se il segnale di reset si alza se il segnale di clr arriva prima della soglia minima, e se rimane attivo per il tempo corretto.
   * Si valuta se il watchdog non attiva nessuna soglia quando il segnale clr arriva dopo la soglia minima ma prima del segnale nmi
2. Con valore di prescaling alto (divisione per 128):
   * Si valuta se il segnale di reset si alza al raggiungimento della soglia massima, e se rimane attivo per il tempo corretto.
   * Si valuta se il watchdog attiva solo il segnale di nmi nel caso in cui il clr arrivasse dopo la soglia di nmi ma prima della soglia massima
   * Si valuta se il watchdog non attiva nessuna soglia quando il segnale clr arriva dopo la soglia minima ma prima del segnale nmi
   * Si valuta se il segnale di reset si alza se il segnale di clr arriva prima della soglia minima, e se rimane attivo per il tempo corretto.
3. Con valore di prescaling basso (divisione per 2):
   * Si valuta se il watchdog impedisce di cambiare i valori nei registri quando è già in esecuzione
   * Si valuta se il segnale di reset si alza al raggiungimento della soglia massima, e se rimane attivo per il tempo corretto.
4. Verificare l’efficacia delle formule:
   * Si è simulato un caso reale, utilizzando 10 ms come soglia massima e 15ns come periodo di clock, si è calcolato il valore ideale per il prescaler e il corrispondente valore da impostare nel registro. In seguito, si è visto che il segnale di reset si attiva effettivamente dopo 10ms.

Nota: con l’espressione “rimane attivo per il tempo corretto” si intende che nei test è stato specificato un tempo in cui si attende tra l’inizio del conteggio e l’arrivo del segnale di start. Se questo tempo scade nel momento voluto significa che il watchdog ha funzionato correttamente.

## Prestazioni

Dall’analisi temporale calcolata tramite simulazione si evidenzia un tempo di risposta massimo di 11,995 ns. Utilizzando un margine del 20% si conclude che il tempo minimo di clock del dispositivo deve essere di 15 ns e di conseguenza la frequenza di clock deve essere non superiore a 66,6 MHz.

Non avendo imposto vincoli temporali allo strumento di sintesi, questa frequenza può essere considerata accettabile in svariate applicazioni.