Relazione Progetto: Intel Timer 8254

Obiettivo:

Realizzare una simulazione del Timer 8254, rispettando la trattazione del chip indicata da Intel nella scheda tecnica pubblicamente disponibile (che troverà allegata nella mail)

Strumenti utilizzati:

Oltre alla scheda tecnica del Timer, è stata riscattata una licenza studentesca per Sigasi (un IDE per il linguaggio Verilog), per una maggiore facilità di organizzazione dei moduli e testbench, un migliore controllo ortografico, e messaggi di errore più verbosi rispetto a Verilogger.

Poiché però sprovvisto di diagramma temporale, è stata riscattata un'altra licenza studentesca per ModelSIM, un programma in grado di importare codice proveniente da Sigasi e simularlo effettivamente.

Procedimento:

Come prima cosa è stato suddiviso il lavoro in due fasi: una "strutturale" il cui obiettivo è costruire tutte le componenti del timer verificando la sincronia della parti, e una "comportamentale" in cui si implementano le modalità di conteggio, i due comandi speciali di lettura (Counter Latch Command e Read-Back Command) e la modalità di conta BCD (Binary Coded Decimal).

Il primo passo della fase strutturale è stato di creare un macromodulo che rappreseti "il guscio" del timer, elencando le varie porte esterne come descritte nella scheda tecnica.

Poi sono stati affrontati i moduli di comunicazione con il mondo esterno, ovvero il Data Bus Buffer (DBB), e il Read/Write Logic (RWL). Da notare che la programmazione del DBB come modulo separato avrebbe richiesto una rete piuttosto complessa e ridondante di porte inout; in realtà è sufficiente la gestione di una sola porta inout (quella del macromodulo) per svolgere la funzione di porta tristato svolta dal DBB.

Il RWL non è particolarmente interessante, ma affacciandosi sul bus interno decidiamo di affrontarlo come prossimo step. Questo si compone di un bus dati, bus indirizzi e bus di controllo da 1 bit che descrive uno stato di lettura/scrittura. Questa organizzazione del bus è arbitraria, in quanto questa componente non è descritta nella scheda tecnica.

A questo punto scriviamo il Control Word Register (CWR), che si occupa di indirizzare le parole di controllo ricevute al contatore corretto. Inoltre controlla se è stato richiesta la funzione di read-back.

Dopo una fase di test in cui ci si assicura che la struttura sia solida, procediamo a creare tre istanze dei contatori. Ogni istanza è resa distinta dalle altre attraverso un parametro che ne descrive l’indirizzo.

Il cervello di ogni Counter è Control Logic (CL), che si occupa di orchestrare le altre componenti del

contatore: ad esempio gestisce le operazioni di scrittura e lettura, e genera diversi segnali di output a seconda della istruzioni ricevute da CWR e degli ingressi nel GATE da parte dell'utente.

Il Counting Elemente (CE) è invece il cuore del contatore, ovvero l'elemento che si occupa effettivamente del conteggio decrescente (seguendo ovviamente la direzione di CL, e segnalandogli il proprio status). Per relazionarsi con il bus interno deve fare uso di Count Register (CR) e Output Latch (OL), che sono registri rispettivamente in entrata e in uscita (rispondono alle esigenze di lettura/scrittura dell'utente, e sono regolati da CL).

Infine lo Status Register è un registro il cui compito è collezionare informazioni sullo stato del contatore, e riferirle all'utente in caso di un Read-Back Command.

La struttura interna del Counter è stata realizzata avvicinandosi il più possibile alle indicazioni della scheda tecnica; alcune deviazioni sono inevitabili, perché la scheda tecnica non descrive appieno tutti i segnali di controllo impiegati, lasciando all'interpretazione del lettore come implementare determinati sistemi. In ogni caso le parti "originali" sono state scritte avendo cura di mantenere il progetto semplice e rispettoso delle indicazioni fornite.

Giunti alla fine dello stadio strutturale, i vari moduli sono stati individualmente testati (si noterà che ogni modulo ha un suo testbench personale), e poi assemblati e connessi tra loro. Infine è stato aggiornato il testbench del macromodulo per verificare il comportamento del complesso. Ben presto è sorta la necessità di poter creare test personalizzati in maniera rapida e piacevole. Questo obiettivo è stato raggiunto dichiarando parole chiave come parametri, che possono essere combinate per produrre rapidamente parole di controllo, e quindi scenari di utilizzo.

Conclusa la fase di test e stirati diversi errori (principalmente di mancata o tardiva comunicazione tra i moduli), ci troviamo ad affrontare la fase comportamentale, che consiste nelle sei modalità di conteggio, i due comandi speciali di lettura, e il conteggio in Binary Coded Decimals (BCD).

Di queste feature solo le prime tre modalità di conteggio sono state implementate e testate.
Questa parte era di competenza del mio compagno di lavoro (Davide Corridori), che ad oggi non l'ha completata. In effetti la quasi totalità del progetto è stata scritta dal sottoscritto (Edoardo Caproni), inclusi i moduli della fase strutturale che teoricamente avrebbe dovuto gestire lui.

Abbiamo raggiunto assieme la decisione di consegnare solamente con la mia firma, e ho cercato di coprire quanto più potessi nel tempo che mi era rimasto a disposizione. Da notare che le strutture necessarie ad ospitare le funzionalità mancanti sono già presenti, e che il progetto è comunque funzionante e testabile.

Edoardo Caproni