Plantilla Señales Wishbone

Interfaz Master

Señal	Uso y comentarios
RST_I	Reiniciar la interfaz
CLK_I	Define el ritmo de transmisión de los datos por el bus
ADR_O(M..N)	Selecciona el dispositivo con el que se quiere comunicar y que parte del mismo
DAT_I(M..N)	Recogida de datos del bus
DAT_O(M..N)	Envío de datos por el bus
WE_O	Señal de escritura en el dispositivo
STB_O	Señal de envío de datos dentro de un ciclo
ACK_I	Confirmación del esclavo de una comunicación correcta
CYC_O	Determina el ciclo de la ejecución

Interfaz Slave

Señal	Uso y comentarios
RST_I	Recibe la señal de reiniciar la interfaz
CLK_I	Recibe la señal define el ritmo de transmisión de los datos por el bus
ADR_I(M..N)	Recibe la señal selecciona qué parte del dispositivo se quiere acceder
DAT_I(M..N)	Recibe la señal recogida de datos del bus
DAT_O(M..N)	Envío de datos por el bus
WE_I	Recibe la señal de escritura en el dispositivo
STB_I	Recibe la señal de envío de datos dentro de un ciclo
ACK_O	Envía confirmación a el master de una comunicación correcta
CYC_I	Recibe la señal determina el ciclo de la ejecución

En este proyecto encontramos 2 cpus con sus correspondientes interfaces wishbone, memorias con sus correspondientes interfaces wishbone y un árbitro de master wishbone.

Las 2 cpu están ejecutando el mismo programa y por lo que están accediendo a las mismas posiciones de memorias y guardando los mismos datos.
Por lo tanto hay conflicto en el accesos de memoria y el árbitro tendrá que mediar quien tiene acceso al bus dando siempre prioridad a la cpu principal.

Esto hace que durante la ejecución en la primera petición de accesos al bus wishbone el árbitro de accesos a la cpu principal y durante ese ciclo la secundaria espera. A partir de aquí la cpu secundaria ha quedado desplazada un ciclo, por lo que sus instrucciones se ejecutarán a destiempo con la primera cpu impidiendo que vuelvan a suceder conflictos en el bus wishbone.