Pg 4

Cnt\_epp-> separa linea de direcciones y línea de datos

Dpram\_mem una memoria por cada tensión vo1 y vo2

Gen\_dir en función de la frec seleccionada lee más rápido o más lento

Cnt\_dac coger un dato de 12bits y aplica protocolo en serie

Dcm controlar la generación de la señal de reloj

\*CLK, señal de reloj exclusiva para todos los modulos (para que el diseño sea síncrono)

Pg 5 6 y 7

Cnt epp: transforma protocolo USB en protocolo paralelo

Ciclo de lectura (escritura de la dirección: ASTRB=nivel bajo(acvtivado) y PWRITE=Nivel bajo y lectura del dato: DSTRB=nivel bajo(activado) y PWRITE NIVEL BAJO)

Ciclo escritura (escritura de dirección astrb y pwrite nivel bajo y escritura de dato )

Pg 7

***Entidad cnt\_epp***

**CICLO DE LECTURA**

PWAIT -> genera estados de espera inv ASTRB e DTRB, no es necesario en esta aplicacion

DIR-> valor de la dirección de un ciclo de acceso

DIR\_VLD->indica con un nivel alto y durante un periodo de la señal clk que hay una nueva dirección

Tenemos que detectar mediante un sistema secuencial el flanco de la señal ASTRB para sacar la señal DIR\_VLD (dibujo santy) [Qo en reposo, Q1 dir\_vld =1] DEBEMOS MODELAR LA MÁQUINA DE ESTADOS (mejor moore que mealy porque si no la salida S tiene un tiempo indefinido (moore es una maquina de mealy con biestables)), se puede solucionar con el proceso secuencial del biestable y la AND (asignación concurrente) (MEJOR SOLUCIÓN)

\*CON UN PROCESO SECUENCIAL DONDE SE MODELAN LOS 3 BIESTABLES y SECUENCIAS CONCURRENTES (mejor solución)

O un proceso por cada biestable, o una máquina de estados

**CICLO DE ESCRITURA**

DATO-> valor del dato correspondiente a un ciclo de escritura

DATO\_VLD-> indica con un nivel alto y durante un período de la señal clk que hay dato válido de un ciclo de escritura (nueva dirección)

\*\*\*Modelado similar a ciclo de lectura pero con DSTRB (detectar su flanco de subida)\*\*\*

Controlar la señal RD

DATO\_RD-> dato de entrada a ser leído en un ciclo de lectura.

CE\_RD-> señal de habilitación o de información que permanece a nivel bajo el mismo tiempo que la señal DSTRB durante el ciclo de lectura (para que el periférico ponga el dato)

Puerta tristate, ya que el puerto es bidireccional, y tenemos que aislar la entrada de la salida del pc para evitar problemas (es un cortocircuito)

\*\*Se modela con una asignación concurrente, en el que una de las alternativas sea Z.

DATA <= DATO\_RD When {DSTRB=0 AND PWRITE=1} else { OTHERS => ‘Z’ }

Creamos los procesos correspondientes a la lógica, algunos mediante sentencias concurrentes,

**APARTADO 1. DISEÑO DE CONTROLADOR DEL PUERTO PARALELO EPP**

Testbench: Generar ciclos que simulen el ciclo de escritura y el ciclo de lectura, con epp\_device sólo es válido para simulación (ojo que ponga sólo simulación en el ISE)

--ciclos de lectura/escritura

Realizar ciclos mediante las 3 constantes que selección la frecuencia y la ram de vo1 y vo2

Llamamos al proceso (añadir llamadas) para hacer el ciclo de escritura en una dirección e igualmente con el ciclo de lectura

**Apartado 2. Descarga en la placa del controlador del puerto paralelo.**

Top\_system1 donde se instancia la entidad, tenemos que implementar dos biestables tipo D Pg 18, realizar multiplexor

El bloque PSH\_BUTTON = 0 => se visualiza la dirección registrada =1 => se visualiza el dato registrado

El bloque al que tiene conectado los 8 switches de la placa (puerto *SWITCHES\_I*) deberá llevarlos a la entrada ***DATO\_RD*** del componente ***cnt\_epp*** cuando se realice un ciclo de lectura en la posición 32HEX.