SENTENCIAS CONCURRENTES

Son sentencias condicionales que tienen al menos un valor por defecto para cuando no se cumple ninguna de las condiciones.

When - else

Señal a modificar ←

Valor_1 **when** condición_1 **else**

Valor_n **when** condición_n **else**

Valor_pordefecto;

En esta sentencia siempre modificamos el valor de una misma señal, pero las condiciones pueden ser independientes (actuar sobre distintas señales cada una), dónde la colocación de las condiciones indica la preferencia de unas sobre otras

----Ejemplos when – else ----

C ← "00" when A = B else
"01" when A < B else
"10";

C ← "00" when A = B else
"01" when D = "00"
else
"10";

When – select - when

With señal_condición select

Señal_a_modificar ← valor_1 when valor_1_señal_condición

valor_n when valor_n_señal_condición….

Valor_por_defecto when others;

Esta sentencia es menos general que la anterior. En este caso se modificará el valor de una señal dependiendo de los valores de una señal condición, aparecerán como máximo tantas líneas como valores posibles pueda tener la señal condición.

```
----------------------------------
----Ejemplos with – select----
----------------------------------
With entrada select
Salida ← "00" when "0001",
         "01" when "0010",
         "10" when "0100",
         "11" when others;
```

Sentencias secuenciales – process

VHDL presenta una estructura particular denominada process que define los límites de un dominio que se ejecutará (simulará) si y sólo si alguna de las señales de su lista de sensibilidad se ha modificado en el anterior paso de simulación. Un process tiene la siguiente estructura:

Process (lista_de_sensibilidad)

- **Asignación de variables**
- **Opcional no recomendable**

Begin

- **Sentencas condicionales**
- **Asignaciones**

End process;

La sentencia process es una de las más utilizadas en programación con VHDL ya que tanto las sentencias condicionales como la descripción de HW secuencial se realiza dentro de él. Pero a la vez es, para aquellos que se acercan por primera vez a la simulación y síntesis con VHDL

Propiedad I: En una estructura process sólo se ejecutan las instrucciones internas en el instante 0 de simulación y cuando varía alguna de las señales de su lista de sensibilidad

Propiedad II: Las asignaciones a señales que se realizan dentro de un process tienen memoria.

Propiedad III: Dentro de un process todas las instrucciones se ejecutan en paralelo, igual que ocurre con las instrucciones que se encuentran fuera de los process. Sin embargo, si dentro del process se asigna valor a una señal en dos sitios diferentes el resultado será aquel de la última asignación, exactamente igual que en los lenguajes de programación comunes

Propiedad IV: Los process se ejecutan en paralelo entre sí.

Propiedad V: Los valores de las señales que se modifican internamente en los process no se actualizan hasta que no se ha ejecutado el process completo.

Propiedades de SIGNAL

🡸 una SIGNAL solo puede ser declarada fuera del código secuencial (PROCESS).

🡸 una SIGNAL que se usa dentro de un código secuencial no se actualiza inmediatamente; el valor de SIGNAL se actualiza hasta que se concluye el código secuencial (PROCESS).

🡸 una asignación de SIGNAL en función de otra señal puede causar la inferencia de un registro.

🡸 solo se le permite una asignación, no permite múltiples asignaciones.

Vector

El std_logic_vectortipo se puede utilizar para crear buses de señal en VHDL. El std_logices el tipo más comúnmente utilizado en VHDL, y el std_logic_vectores la versión matriz de la misma.

Si bien std_logices excelente para modelar el valor que puede llevar un solo cable, no es muy práctico para implementar colecciones de cables que van hacia o desde los componentes. El std_logic_vectores un tipo de material compuesto, lo que significa que es una colección de subelementos. Las señales o variables del std_logic_vectortipo pueden contener un número arbitrario de std_logicelementos.