# Laboratorio de Diseño Digital – Año 2006
# Recomendaciones para síntesis de diseños en VHDL orientada a dispositivos programables en ambientes EDA.

El lenguaje VHDL permite utilizar una gran variedad de estructuras que aportan flexibilidad y alto nivel de abstracción a la descripción del sistema a diseñar. Sin embargo, se debe recordar, que estos elementos que resultan de gran utilidad para modelado y simulación, no siempre son sintetizables al momento de implementar el circuito en un dispositivo concreto. La herramienta de trabajo que se seleccione, determinará las restricciones al proceso de síntesis, pero se pueden hacer ciertas recomendaciones que son válidas para la mayoría de ellas. Por lo tanto, al encarar un diseño, si el objetivo es llegar a la síntesis del mismo utilizando algún entorno EDA, es fundamental tenerlas en cuenta para evitar problemas. A continuación vamos a listar algunas.

1. *Evitar las cláusulas temporales (AFTER):* Normalmente los sintetizadores prohíben expresamente el uso de asignaciones con retraso en las señales, otros simplemente las ignoran, pero lo que está claro es que la herramienta intentará implementar el circuito funcionalmente, por lo que estas cláusulas carecen de sentido.
2. *Evitar las sentencias de espera:* El uso de WAIT está bastante restringido, algunos entornos que lo permiten exigen que sea la primera instrucción del PROCESS, y sólo se admite una sentencia de este tipo por proceso. En general se aconseja no utilizar WAIT, puesto que la herramienta puede tener dificultades para interpretarla. Es preferible reemplazar estas sentencias por listas sensibles, y para muchos sintetizadores es prácticamente la única posibilidad.
3. *Cuidado con las listas sensibles:* La mayoría de sintetizadores admiten la lista sensible en los procesos, pero no siempre la interpretan como lo haría un simulador. Al sintetizar, puede ocurrir que el proceso se ejecute cuando se produce un evento asociado a una señal que se encuentra dentro del proceso, pero que no es mencionada en su lista de sensibilidad. Es decir, el sintetizador, a veces, "amplía" la lista sensible según le parece y sin avisar, normalmente con el objetivo de que la lógica que describe el proceso sea puramente *combinacional*.
4. *Inicialización de variables y señales:* En simulación esto funciona correctamente, pero no necesariamente en síntesis. En general los sintetizadores no toman en cuenta estas inicializaciones y tienen un sistema propio de inicialización de señales, que representan registros. Es preferible realizar el reset explícitamente.
5. *Evitar IFs anidados:* Normalmente las herramientas tienden a no sintetizar de manera óptima varios condicionales anidados entre sí. Los condicionales es mejor utilizarlos a solas. Utilizar CASE mejor que varios IFs. Las estructuras CASE tienen para los sintetizadores un modelo optimizado de síntesis, generalmente mejor que lo mismo descrito mediante IFs.
6. *Niveles lógicos:* Ya se mencionó cuales son los niveles lógicos del tipo de datos std_logic que las herramientas de síntesis pueden interpretar, y no todas lo hacen de igual forma. En especial el ´-´ (*don´t care*), suele pensarse como un mecanismo para indicar al sintetizador que elija el mejor valor posible (para minimizar lógica) y en

cambio no es así: es muy probable que la herramienta lo interprete como un nivel lógico más.

7. *Asignación múltiple a señal:* Un error común es la asignación de una misma señal en procesos diferentes o, lo que es lo mismo, asignación de una misma señal en distintas instrucciones concurrentes. En general, y para evitar esto, conviene dividir el problema por las salidas y no por las entradas, de manera que en cada proceso se integre toda la lógica referida a una salida o grupos de salidas relacionadas. Si bien la utilización de los tipos de datos *resueltos* puede parecer que soluciona este problema (al incluir el nivel Z), podríamos estar enmascarando un error de lógica, difícil de depurar. El uso de los tipos *resueltos* debe dejarse sólo para la implementación de *buses* (recursos compartidos).
8. *Especificar la arquitectura:* Es posible que se creen varias descripciones para un mismo circuito. Normalmente el sintetizador escogerá la primera que le parezca, por lo que conviene especificar cuál de todas las arquitecturas se desea sintetizar mediante un bloque de configuración CONFIGURATION.
9. *Permitir discrepancia:* Normalmente es fácil sintetizar algo simple como s<=NOT s ya que no es más que una puerta inversora conectada sobre si misma que puede servir muy bien para generar una señal de reloj con periodo el doble que el retraso que la puerta presente. Si se intenta simular algo como la instrucción anterior, se comprobará que la simulación se queda colgada en esa instrucción puesto que no hay retrasos y se llama a sí misma una y otra vez. Por lo tanto, en estos casos, aunque la simulación es incorrecta, la síntesis no lo es.
10. *Consideraciones sobre la señal de reloj:* Para que un circuito secuencial sea sintetizado con éxito, se deben tener en cuenta algunas directrices que atañen sobretodo a la señal de reloj:
    - 10.1. *Debe usarse un único reloj por proceso:* Normalmente cada proceso en una descripción en VHDL corresponde a una salida o señal interna del sistema. Si se utilizan dos relojes en un mismo proceso, dicha señal resulta sincronizada por dos relojes distintos. Esto implica en síntesis, realizar lógica sobre la señal de reloj, que no es para nada aconsejable. Por lo tanto, de ser necesario, se deja en manos del diseñador obtener una única señal de reloj que pueda ser función de otras.
    - 10.2. *Especificar el flanco de reloj:* Se debe indicar en un proceso el flanco activo de reloj con el atributo *event*. Además, esta especificación no debe ser usada como operando en una sentencia: la instrucción IF not (clk'event and clk='1') THEN... sería incorrecta.
    - 10.3. *Cuidado con el uso de else*: Cuando en una sentencia IF se comprueba el flanco del reloj, no debe seguir un ELSE. La comprobación sintáctica puede permitirlo. Por lo tanto se podría poner, pero desde un punto de vista de realización física del circuito no tendría ningún sentido.