Introducción al Diseño Digital Moderno

Clase 1:

Verilog:

Introducción al lenguaje de descripción de *hardware* Verilog; el módulo, porciones declarativa y ejecutiva; Conexiones (*nets*) como tipo de dato, declaración de variables; asignación continua; conexiones externas e internas, puertos de entrada y salida, parámetros; lógica de 4 estados; buses; operadores lógicos, aritméticos, relacionales, binarios. Instanciación de módulos. *Test benches*, pasos básicos de construcción de *test benches*.

DSP:

Punto fijo: concepto, matemática de precisión finita, definición; punto flotante versus punto fijo; notación; operaciones, suma y resta, multiplicación, pasos generales. Ajuste de rango: *overflow* (*wrap*), *overflow* en ángulos, saturación. Ajuste de resolución: redondeo, truncado. Implementación en hardware.

Trabajos prácticos:

1. Multi-compuerta.
2. *Half adder* y *Full adder*.
3. Generador y verificador de paridad.
4. Multiplicador con saturación y redondeo.
5. *Slicer*.

Clase 2:

Verilog:

Niveles de descripción: RTL, algorítmico y estructural.

Descripción RTL: Asignación condicional y su operador; multiplexores. *Initial* y *always*, descripción con sentencias secuenciales, lista de sensibilidad. *If-else*, *case*, *for*. Asignaciones secuenciales, bloqueante y antibloqueante.

Descripción algorítmica: Inferencia de componentes: multiplexores, *flip-flops*, contadores, registros de desplazamientos. Máquinas de estado finito (FSM): máquinas de Mealy y de Moore, codificación de estados, diagramas de transición de estados, tablas, versión en Verilog. Reglas para evitar comportamiento de *latch* no deseado. Funciones sintetizables.

DSP: Filtro FIR e IIR. Teoria. Formas Directa y transpuesta. Detalles de implementación. Introducción a filtros adaptivos.

Trabajos prácticos:

6. Unidad Aritmético Lógica (ALU) parametrizada.
7. Filtro FIR
8. Filtro IIR
9. Secuenciador para filtro adaptivo

Clase 3:

Descripción estructural: bloques *generate*, *if-else*, *case*, *for*, variables *genvar*. Verilog para simulación: funciones y tareas; *automatic* para recursividad; directivas del compilador; tareas del sistema; manejo de archivos, modificadores para cadenas.

Sugerencias de código para síntesis: reglas generales, codificadores de prioridad, multiplexores, operaciones con números *signed* y *unsigned*, inferencia de sumadores con acarreo, paréntesis y balanceo de árboles de suma o producto, *pipelining*, bibliografía de consulta.

Trabajos prácticos:

10. Multiplicador de matrices
11. Filtro FIR Parametrizado

Clase 4:

Caso de estudio: DFE (*Decision Feedback Equalizer*), teoría y análisis de implementación.

Trabajos prácticos:

12. *Decision Feedback Equalizer*

Clase 5:

Caso de estudio: FFT compleja *Radix*-2. Teoría del algoritmo. Resolución en lenguajes secuenciales. Aritmética compleja. *Bit reverse ordering*. *Butterflies*. Estructura de interconexión.

Trabajos prácticos:

13. FFT Radix-2

Lista completa de trabajos prácticos

Clase 1:

1. Multi-compuerta
2. *Half adder* y *Full adder*.
3. Generador y verificador de paridad.
4. Multiplicador con saturación y redondeo.
5. Slicer

Clase 2:

6. Unidad Aritmético Lógica (ALU) parametrizada.
7. Filtro FIR
8. Filtro IIR
9. Secuenciador para filtro adaptivo

Clase 3:

10. Multiplicador de matrices
11. Filtro FIR Parametrizado

Clase 4:

12. *Decision Feedback Equalizer*

Clase 5:

13. FFT Radix-2