

INTRODUCCIÓN A LOS COMPUTADORES

OBJETIVOS FORMATIVOS RESPECTO A CONOCIMIENTOS

Que el alumno sea capaz de:

Comprender el lenguaje máquina/ensamblador, la estructura interna a nivel de análisis/síntesis de circuitos digitales y la ejecución de las instrucciones de un procesador sencillo¹.

A. INTRODUCCIÓN

1 Justificar la necesidad y entender el proceso de codificación de la información para su procesado en un circuito digital.

- 1.1 Definir conceptos básicos tales como: información, señal eléctrica analógica, señal eléctrica digital, codificación de la información y sincronización.

2 Conocer la estructura y el funcionamiento de un computador de propósito general a nivel de grandes bloques (Unidad de Proceso, Unidad de Control, Memoria, Entradas/Salidas, Bus).

- 2.1 Describir la funcionalidad de los grandes bloques que forman un computador (Procesador -Unidad de Proceso y Unidad de Control-, Memoria, Entradas/Salidas y Bus), como están interconectados y como interaccionan en la ejecución de las instrucciones.

3 Conocer, a grandes rasgos, los procesos involucrados en la ejecución de un programa escrito en lenguaje de alto nivel (compilación, ensamblado, carga y ejecución).

- 3.1 Describir en que consisten los procesos de compilación, ensamblado, carga y ejecución de un programa.

B. SISTEMAS DE NUMERACIÓN Y OPERACIONES BÁSICAS

4 Comprender el sistema convencional en base b para la representación de números naturales y saber como se realizan las operaciones aritméticas básicas.

- 4.1 Definir el sistema convencional en base b para la representación de números naturales.
- 4.2 Expresar el rango de la representación en binario con n bits.
- 4.3 Efectuar extensiones de rango de un número en binario.
- 4.4 Expresar en binario cualquier número natural y viceversa.
- 4.5 Expresar en Hexadecimal cualquier tira de bits.

1. Usaremos la Máquina Rudimentaria (MR), que se describe en el Anexo A. No obstante, podría utilizarse cualquier subconjunto de un procesador RISC como el MIPS, DLX,...

- 4.6 Saber realizar la suma, multiplicación por potencias de 2 y división por potencias de 2 de números naturales representados en binario, incluida la detección de resultado no representable.

5 Comprender la representación de números enteros en complemento a 2 y saber como se realizan las operaciones aritméticas básicas.

- 5.1 Definir la representación en complemento a 2 para números enteros.
- 5.2 Expresar el rango de la representación en complemento a 2 con n bits.
- 5.3 Efectuar extensiones de rango de un número en complemento a 2.
- 5.4 Expresar en complemento a 2 cualquier número entero y viceversa.
- 5.5 Saber realizar la suma, cambio de signo, resta, multiplicación por potencias de 2 y división por potencias de 2 de números enteros representados en complemento a 2, incluida la detección de resultado no representable.

C. ANÁLISIS Y SÍNTESIS DE SISTEMAS DIGITALES

6 Adquirir los conocimientos básicos sobre sistemas digitales necesarios para comprender, a nivel de análisis y síntesis, implementaciones sencillas de la MR construidas a partir de los elementos del Anexo B.

- 6.1 Obtener la tabla de verdad y/o las expresiones lógicas de un sistema combinacional (de como máximo 4 bits de entrada) a partir de una descripción textual de su funcionamiento.
- 6.2 Analizar sistemas combinacionales.
A partir del esquema de un circuito con puertas y/o bloques combinacionales (del Anexo C) obtener:
 - a) Su tabla de verdad (si tiene como máximo 4 bits de entrada) o una descripción de su funcionalidad.
 - b) El cronograma de las señales de salida (conocido el retardo de cada elemento y la evolución temporal de las señales de entrada).
 - c) El tiempo de propagación del circuito.
- 6.3 Comprender los bloques combinacionales enumerados en el Anexo C.
 - a) Describir la funcionalidad de cada uno de los bloques.
 - b) Especificar la tabla de verdad y/o una expresión lógica de cada salida para bloques de pocas entradas.
 - c) Interconectar bloques de pocas entradas para formar bloques de más entradas.
- 6.4 Sintetizar sistemas combinacionales.
 - a) A partir de una tabla de verdad o de una expresión lógica, obtener:
 1. La expresión lógica y su implementación directa a 2 niveles en suma de minterms.
 2. Otras implementaciones, a nivel de puertas lógicas, obtenidas mediante la manipulación de expresiones lógicas usando las leyes del álgebra de Boole enumeradas en el Anexo D. Saber elegir los diseños que requieren menor hardware y/o menor tiempo de propagación.
 3. La implementación con el mínimo número de puertas NOT-AND-OR a 2 niveles usando tablas de Karnaugh (sólo para sistemas de como máximo 4 entradas).
 4. La implementación con un decodificador y una puerta OR.
 5. La implementación con una ROM.

- b) A partir de una descripción textual de un sistema que procesa números de n bits encontrar una implementación ad-hoc usando puertas y bloques combinacionales del Anexo C.
- 6.5 Obtener el grafo de estados de un sistema secuencial (de Moore de como máximo 8 estados) a partir de una descripción textual de su funcionamiento.
- 6.6 Analizar sistemas secuenciales.
- a) A partir del esquema de un circuito secuencial de Moore, formado por elementos del Anexo B con $\text{NúmeroBitsEntrada} + \log_2(\text{NúmeroEstados}) \leq 4$, obtener:
1. Su grafo de estados.
 2. El cronograma de las señales de salida (conocido el retardo de cada elemento, el estado de los biestables y la evolución temporal de las señales de entrada).
 3. El tiempo de ciclo mínimo del sistema (dados los tiempos de propagación de las puertas).
- b) A partir de un circuito secuencial que procesa números de n bits, formado por unidad de proceso (construida con puertas y bloques del Anexo C) y unidad de control (especificada mediante un grafo de Moore), obtener una descripción textual de su funcionalidad y el tiempo de ciclo mínimo (conocidos los tiempos de propagación de los bloques).
- 6.7 Sintetizar sistemas secuenciales.
- a) A partir de un grafo de estados de Moore (con $\text{NúmeroBitsEntrada} + \log_2(\text{NúmeroEstados}) \leq 4$) encontrar un circuito que lo implemente con el mínimo número de biestables D y alguna de las técnicas de síntesis de combinacionales (con puertas, con decodificador y OR o con ROM).
- b) A partir una descripción textual de la funcionalidad de un circuito, obtener un circuito secuencial compuesto por una unidad de control secuencial de Moore y una unidad de proceso (construida con puertas y bloques del Anexo C).
- 6.8 Implementar los bloques secuenciales del anexo C con bloques combinacionales y biestables D.
- 6.9 Comprender la funcionalidad de una memoria RAM. Describir la funcionalidad y un cronograma de las señales de entrada/salida de una memoria RAM.

D. MAQUINA RUDIMENTARIA

- 7 Conocer el juego de instrucciones de la Máquina Rudimentaria (MR) y su especificación en lenguaje máquina y ensamblador.**
- 7.1 Indicar como queda modificado el estado del computador después de la ejecución de una secuencia de como máximo 6 instrucciones especificadas en lenguaje ensamblador de la MR.
- 7.2 Escribir pequeños programas en lenguaje ensamblador de la MR (como máximo 6 instrucciones) cuya funcionalidad se especifica mediante un texto o mediante una sentencia sencilla de un lenguaje de alto nivel.
- 7.3 Saber ensamblar y desensamblar código de la MR ayudados por una tabla que contiene el formato y codificación de las instrucciones.

8. Comprender la estructura interna y el funcionamiento de una implementación dada de la MR (especificada a nivel de bloques para la unidad de proceso y un grafo de estados para la unidad de control) y saber calcular el tiempo que tarda en ejecutarse un programa.

- 8.1 Indicar la evolución temporal (cronograma) de algunas señales de la Unidad de Proceso y de la Unidad de Control durante los ciclos que dura la ejecución de alguna instrucción de lenguaje máquina (o una secuencia de 2 ó 3 instrucciones).
- 8.2 Saber encontrar el tiempo de ciclo mínimo (dados los tiempos de propagación de los circuitos combinacionales), el número de ciclos en que se ejecuta cada instrucción y el tiempo de ejecución de un programa (dado el número de instrucciones ejecutadas de cada tipo).
- 8.3 Elegir razonadamente entre varias alternativas de diseño de la UP y/o UC, aquella que reduzca el tiempo de ejecución de los programas y/o la que requiera menor hardware.

9. Proponer pequeñas modificaciones a la MR

- 9.1 Proponer modificaciones de la implementación de la MR para que pueda ejecutar una nueva instrucción de lenguaje máquina (de complejidad equivalente a las originales).
- 9.2 Proponer pequeñas modificaciones de la implementación de la MR para que la cumpla restricciones respecto del hardware disponible, tiempo de ciclo o número de ciclos de ejecución de cada instrucción.

Anexo A

El lenguaje máquina original de la Máquina Rudimentaria consiste en 4 instrucciones aritmético-lógicas con operandos y resultado en registros (ADD, SUB, ASR y AND), 2 instrucciones aritmético-lógicas con un operando inmediato (ADDI, SUBI), 7 instrucciones de ruptura de secuencia según los flags N y Z y con dirección destino absoluta (BR, BEQ, BL, BLE, BNE, BGE y BG) y 2 instrucciones de acceso a memoria con direccionamiento base más desplazamiento (LOAD y STORE). Es una máquina con instrucciones y datos de 16 bits, 8 registros generales de 16 bits y un espacio de direccionamiento de solo 8 bits.

Anexo B

Puertas lógicas NOT, AND, OR y XOR, Biestable D activado por flanco y generador de señal de reloj. Con estos elementos básicos se construyen otros mas complejos. No es necesario saber como estos elementos básicos están contruidos internamente, sólo debe conocerse su funcionalidad a nivel de entrada/salida.

Anexo C

Bloques combinacionales: decodificador, multiplexor, multiplexor de buses, sumador/restador, ALU y ROM.
Bloques secuenciales: registro, registro con control de carga y banco de registros con 1 ó 2 buses de lectura y 1 de escritura.

Anexo D

Las leyes (axiomas y teoremas) del álgebra Booleana que el alumno debe conocer para manipular expresiones lógicas son:
 $A+0=A$ y $A*1=A$
 $A+1=1$ y $A*0=0$
 $A+\text{NOT}(A)=1$ y $A*\text{NOT}(A)=0$
 $A+B=B+A$ y $A*B=B*A$
 $A+(B+C)=(A+B)+C$ y $A*(B*C)=(A*B)*C$
 $A*(B+C)=(A*B)+(A*C)$ y $A+(\text{NOT}(A)*B)=A+B$
Ley de Morgan: $\text{NOT}(A*B)=\text{NOT}(A)+\text{NOT}(B)$