75.03 & 95.57 Organización del Computador

U3 – Arquitectura del conjunto de instrucciones (segunda parte)

U3 – Arquitectura del conjunto de instrucciones

- ISA (Instruction Set Architecture) / Arquitectura de Programación
  - Repertorio de instrucciones
  - Especificación de su operación
  - Registros
  - Tipos de datos
  - Modos de direccionamiento
  - Formato de instrucciones
  - Memoria
    - Word size
    - Big / Little Endian
    - Direccionamiento
    - Espacio de direcciones (address space)

U3 – Arquitectura del conjunto de instrucciones

- Repertorio de instrucciones
  - ¿Qué es una instrucción de máquina?
    - Opcode + Operandos (0 a n)
  - Definición de repertorio de instrucciones
  - Categorías
    - Aritméticas y lógicas
      - Ejemplos:
        - add, subtract, multiply, divide (BPF c/s, Decimal, BPFlot)
        - and, or, xor
    - Movimiento de datos
      - Ejemplos:
        - load, store, move
    - Entrada / Salida
      - Ejemplos:
        - start I/O
    - Control de flujo
      - Ejemplos:
        - branch, jump, compare, call, return

U3 – Arquitectura del conjunto de instrucciones

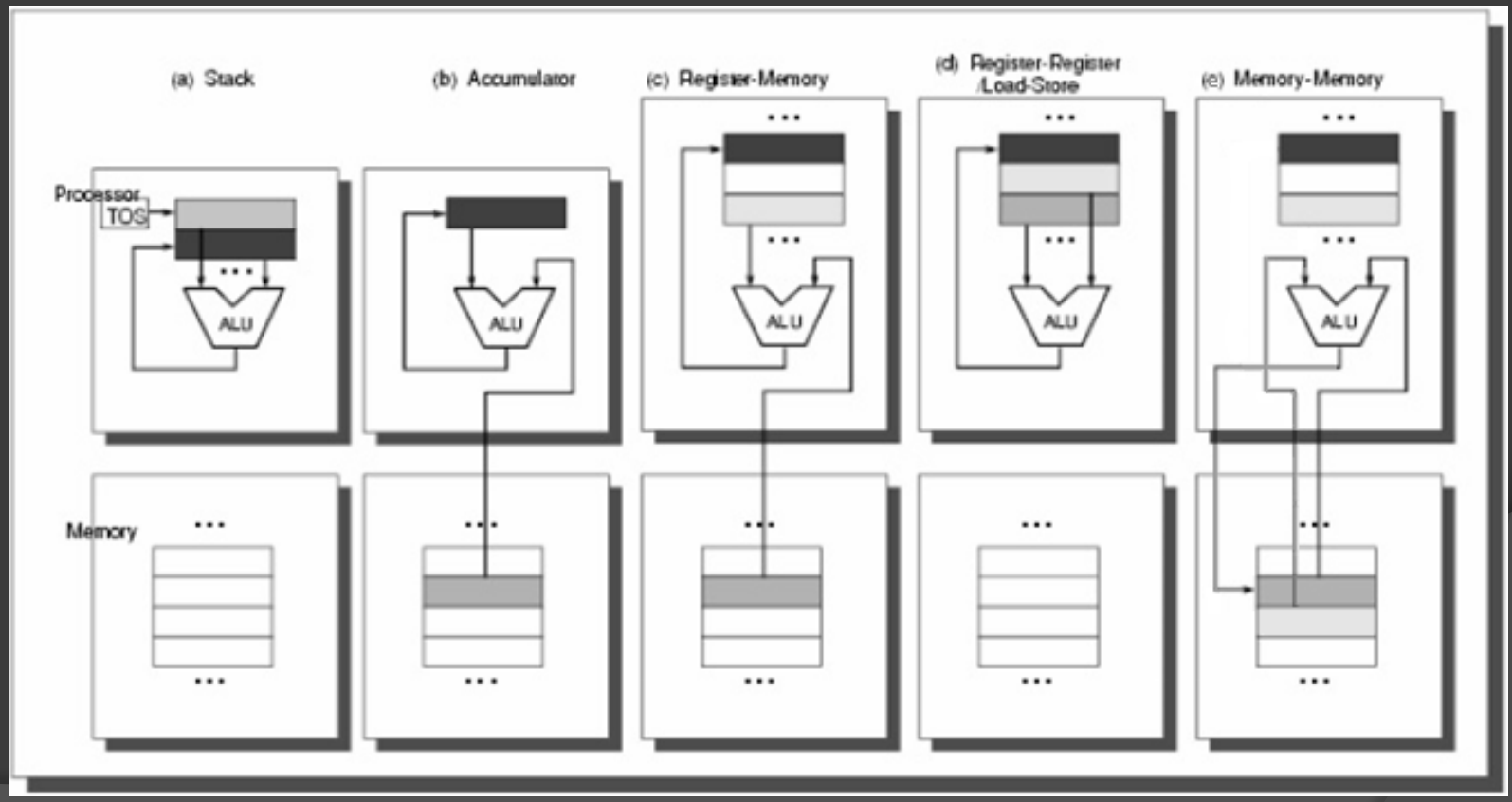
- Repertorio de instrucciones
  - Tipos de operandos
    - Registro
      - Ejemplo: *ADD EAX,EBX* (Intel IA-32)
    - Memoria
      - Ejemplo: *MOV EAX,DATO* (Intel IA-32)
    - Inmediato
      - Ejemplo: *MOV EAX,5* (Intel IA-32)

U3 – Arquitectura del conjunto de instrucciones

- Repertorio de instrucciones
  - Clasificación según la ubicación de los operandos
    - Stack (‘60s a ‘70s)
    - Acumulador (antes de ‘60s)
    - Registro-Memoria (‘70s hasta ahora)
    - Registro-Registro (Load/Store) (‘60s hasta ahora)
    - Memoria-Memoria (‘70s a ‘80s)

U3 – Arquitectura del conjunto de instrucciones

- Repertorio de instrucciones
  - Clasificación según la ubicación de los operandos



U3 – Arquitectura del conjunto de instrucciones

- Repertorio de instrucciones
  - ¿Cómo se resuelve C = A + B según cada arquitectura?

Stack	Accumulator	Register (register-memory)	Register (load-store)	Memory (memory-memory)
Push A	Load A	Load R1,A	Load R1,A	Move C,A
Push B	Add B	Add R3,R1,B	Load R2,B	Add C,B
Add	Store C	Store R3,C	Add R3,R1,R2	
Pop C			Store R3,C	

U3 – Arquitectura del conjunto de instrucciones

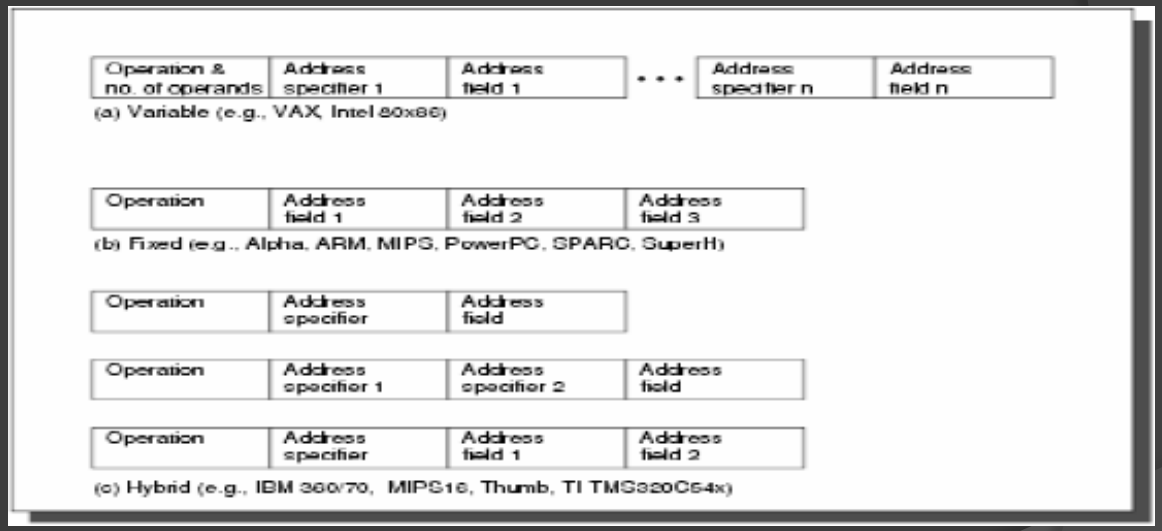
- Clasificación de la ISA según el número de direcciones
  - 0 direcciones (Stack)
    - Ejemplo: *add* — *TOS ← TOS + Next*
  - 1 dirección (Acumulador)
    - Ejemplo: *add A* — *AC ← AC + Mem[A]*
  - 2 direcciones (Reg-Mem/Reg-Reg/Mem-Mem)
    - Ejemplo: *add R1,A* — *R1 ← R1 + Mem[A]*
  - 3 direcciones (Reg/Mem)
    - Ejemplo: *add R1,R2,R3* — *R1 ← R2 + R3*

U3 – Arquitectura del conjunto de instrucciones

- Formato de instrucciones (Encoding)
  - Definición
    - “Define el despliegue de los bits que componen la instrucción”
  - Componentes
    - Opcode
    - 0 a n operandos
    - Modo de direccionamiento de cada operando
    - Flags

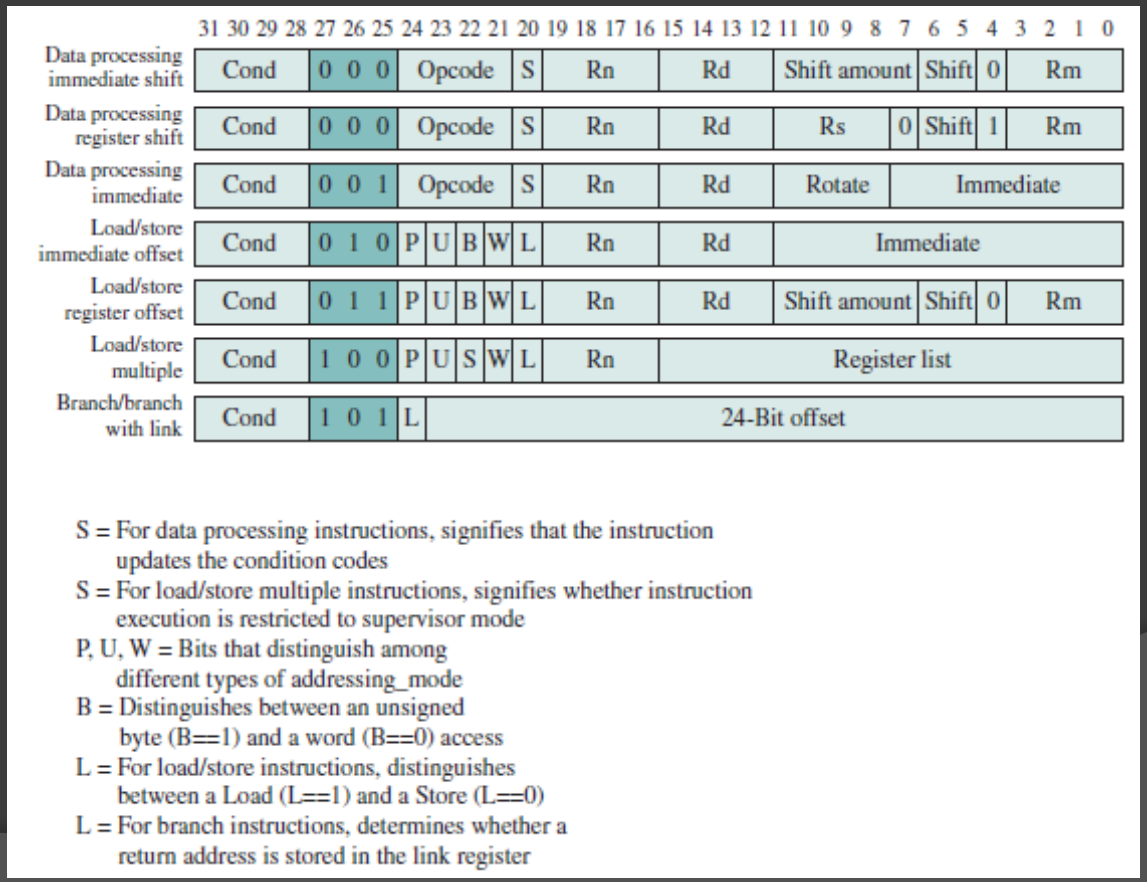
U3 – Arquitectura del conjunto de instrucciones

- Formato de instrucciones (Encoding)
  - Clasificación
    - Fijo
      - Ejemplos:
        - ARM
        - MIPS
        - SPARC
        - PowerPC
    - Variable
      - Ejemplos:
        - Intel x86
        - VAX
    - Híbrido
      - Ejemplo:
        - IBM Mainframe



U3 – Arquitectura del conjunto de instrucciones

- Formato de instrucciones (Encoding)
  - Formatos ARM

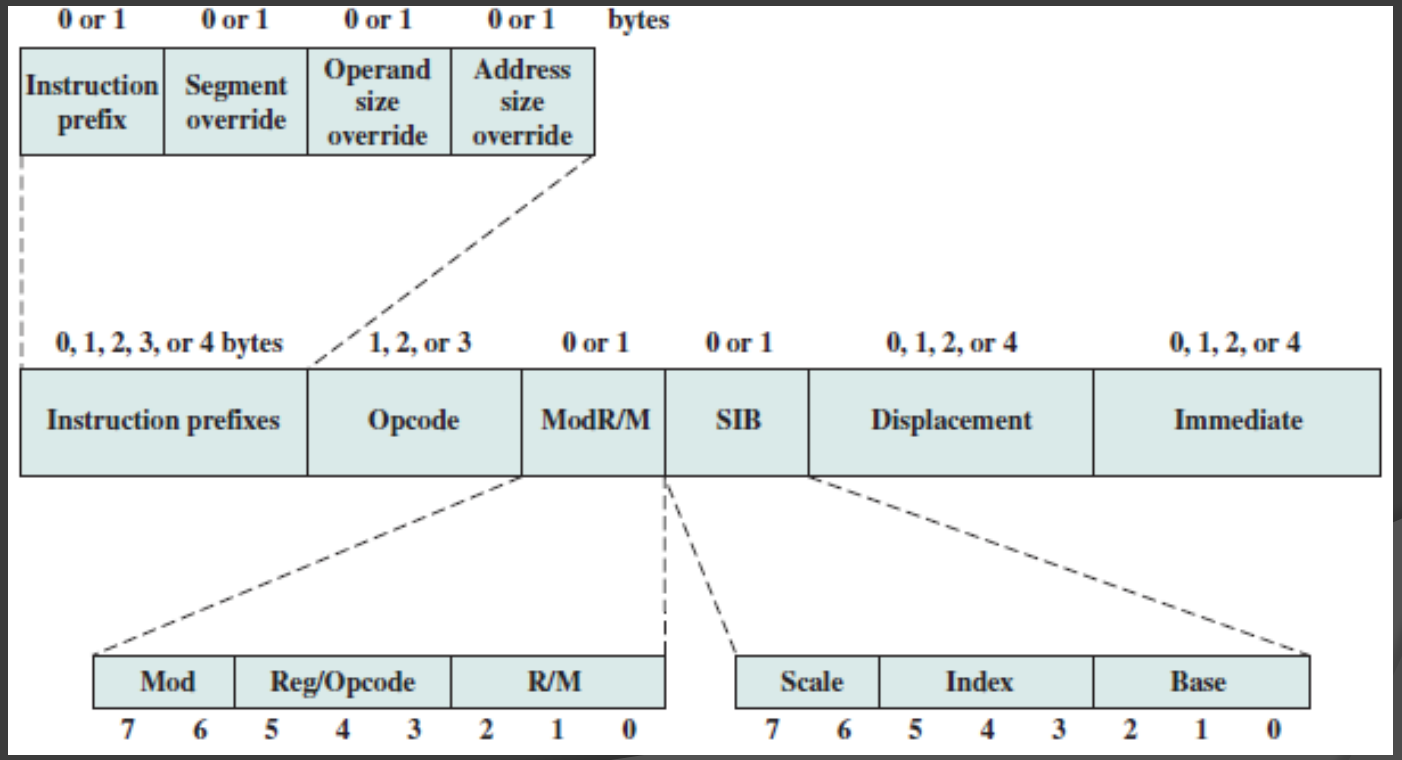


	31 30 29 28	27 26 25	24	23 22 21	20	19 18 17 16	15 14 13 12	11 10 9 8	7	6 5	4	3 2 1 0
Data processing immediate shift	Cond	0 0 0	Opcode		S	Rn	Rd	Shift amount		Shift	0	Rm
Data processing register shift	Cond	0 0 0	Opcode		S	Rn	Rd	Rs	0	Shift	1	Rm
Data processing immediate	Cond	0 0 1	Opcode		S	Rn	Rd	Rotate	Immediate			
Load/store immediate offset	Cond	0 1 0	P	U B W	L	Rn	Rd	Immediate				
Load/store register offset	Cond	0 1 1	P	U B W	L	Rn	Rd	Shift amount		Shift	0	Rm
Load/store multiple	Cond	1 0 0	P	U S W	L	Rn	Register list					
Branch/branch with link	Cond	1 0 1	L	24-Bit offset								

S = For data processing instructions, signifies that the instruction updates the condition codes
S = For load/store multiple instructions, signifies whether instruction execution is restricted to supervisor mode
P, U, W = Bits that distinguish among different types of addressing_mode
B = Distinguishes between an unsigned byte (B==1) and a word (B==0) access
L = For load/store instructions, distinguishes between a Load (L==1) and a Store (L==0)
L = For branch instructions, determines whether a return address is stored in the link register

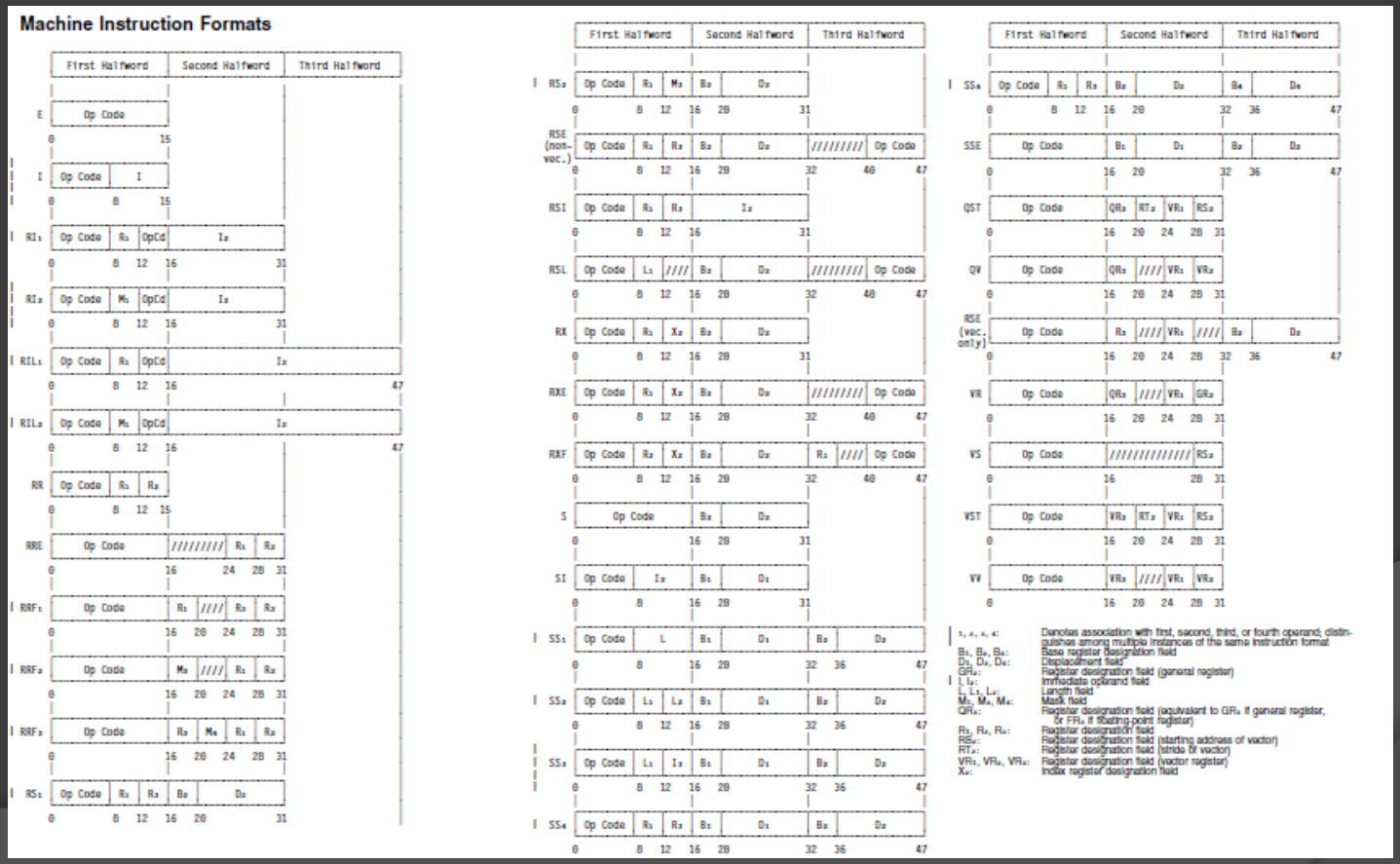
U3 – Arquitectura del conjunto de instrucciones

- Formato de instrucciones (Encoding)
  - Formatos x86



U3 – Arquitectura del conjunto de instrucciones

- Formato de instrucciones (Encoding)
  - Formatos IBM Mainframe

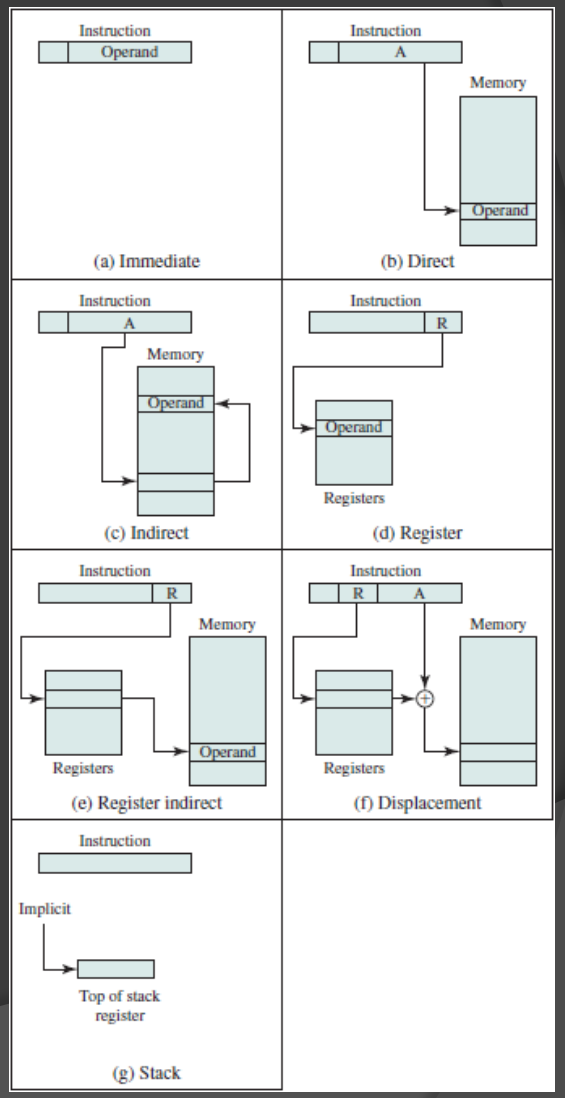


U3 – Arquitectura del conjunto de instrucciones

- Tipos de datos
  - Numéricos
    - BPF s/s
    - BPF c/s
    - BPFlotante (IEEE 754 o propietarios)
    - BCD (decimales)
  - Caracteres
    - ASCII
    - EBCDIC
    - Unicode
  - Datos lógicos
  - Direcciones

U3 – Arquitectura del conjunto de instrucciones

- Modos de direccionamiento
  - Inmediato
  - Memoria Directo
  - Memoria Indirecto
  - Registro
  - Registro Indirecto
  - Desplazamiento
    - Relativo (Program counter)
    - Registro Base
    - Indexado
  - Stack



U3 – Arquitectura del conjunto de instrucciones

- Memoria
  - Direccionamiento (Celda)
  - Tamaño de palabra (Word size)
  - Big vs Little Endian
  - Espacio de direcciones (address space)

U3 – Arquitectura del conjunto de instrucciones

- Memoria
  - Big vs Little Endian
    - Orden de los bytes (Ej. 12345678_{16})
    - Big endian
      - IBM Mainframe
      - SPARC (RISC en general)
      - Motorola 68k
    - Little endian
      - Intel x86
      - VAX
      - Alpha

Address	Value
184	12
185	34
186	56
187	78

Address	Value
184	78
185	56
186	34
187	12

U3 – Arquitectura del conjunto de instrucciones

- Control de flujo
  - Métodos para evaluar condiciones de bifurcación
    - Condition Code (CC)
      - Ejemplos: Intel x86, ARM, PowerPC, SPARC
    - Condition Register
      - Ejemplos: Alpha, MIPS
    - Compare and Branch
      - Ejemplos: VAX

Name	Examples	How condition is tested	Advantages	Disadvantages
Condition code (CC)	80x86, ARM, PowerPC, SPARC, SuperH	Special bits are set by ALU operations, possibly under program control.	Sometimes condition is set for free.	CC is extra state. Condition codes constrain the ordering of instructions since they pass information from one instruction to a branch.
Condition register	Alpha, MIPS	Tests arbitrary register with the result of a comparison.	Simple.	Uses up a register.
Compare and branch	PA-RISC, VAX	Compare is part of the branch. Often compare is limited to subset.	One instruction rather than two for a branch.	May be too much work per instruction for pipelined execution.

U3 – Arquitectura del conjunto de instrucciones

- Referencias
  - “Computer Organization and Architecture – Designing for Perfomance” 10ma edición. William Stallings (http://williamstallings.com/ComputerOrganization/)
  - “Structured Computer Organization” 6ta edición. Andrew Tanenbaum / Todd Austin (http://www.pearsonhighered.com/educator/product/Structured-Computer-Organization-6E/9780132916523.page)
  - “Computer Architecture: A Quantitive Approach” 5ta edición. John L. Hennessy / David A. Patterson
  - “Computer Organization and Embedded Systems” 6ta edición. C. Hamacher / Z. Vranesic / S. Zaky / N. Manjikian