

Introducción

Estructura de Computadores
1^a-2^a Semana

Bibliografía:

- | | |
|-----------------|--|
| [TOC] Temas 1-3 | Apuntes Tecnología y Organización de Computadores |
| [HAM03] Cap.1 | Organización de Computadores. Hamacher, Vranesic, Zaki. McGraw-Hill 2003 Signatura ESIIT/ C.1 HAM org |
| [BRY11] Cap.1 | Computer Systems: A Programmer's Perspective. Bryant, O'Hallaron. Pearson, 2011 Signatura ESIIT/ C.1 BRY com |
| [PRI10] | Introducción a la Informática. Prieto, Lloris, Torres. McGraw-Hill Interamericana 2010 Signatura ESIIT/ A.0 PRI int |

Guía de trabajo autónomo (4h/s)

■ Repaso

- Apuntes TOC

■ Lectura

- Cap.1 Hamacher
- Cap.1 CS:APP (Bryant/O'Hallaron)
- Guión de la Práctica 1

Bibliografía:

[TOC] Temas 1-3

Apuntes Tecnología y Organización de Computadores

[HAM03] Cap.1

Organización de Computadores. Hamacher, Vranesic, Zaki. McGraw-Hill 2003

Signatura ESIIT/[C.1 HAM org](#)

[BRY11] Cap.1

Computer Systems: A Programmer's Perspective. Bryant, O'Hallaron. Pearson, 2011

Signatura ESIIT/[C.1 BRY com](#)

[PRI10]

Introducción a la Informática. Prieto, Lloris, Torres. McGraw-Hill Interamericana 2010

Signatura ESIIT/A.0 PRI int

TOC

■ Tecnología y Organización de Computadores

■ TEMARIO TEÓRICO:

- 1. Introducción
 - 1.1 Conceptos básicos
 - 1.2 Estructura funcional de un computador
 - 1.3 Niveles conceptuales de descripción de un computador
 - 1.4 Clasificación de computadores
 - 1.5 Parámetros que caracterizan las **prestaciones** de un computador
- 2. Unidades funcionales de un computador
 - 2.1. El procesador
 - 2.2. La memoria
 - 2.3. Periféricos de E/S
 - 2.4. Estructuras básicas de interconexión
- 3. Representación de la información en los computadores
 - 3.1 Representación de textos
 - 3.2 Representación de sonidos
 - 3.3 Representación de imágenes
 - 3.4 Representación de datos numéricos

Vocabulario

■ Arquitectura

- Aspectos necesarios para redactar programa ensamblador correcto
- Incluye: registros CPU, repertorio instrucciones, modos direccionamiento

■ Organización (del computador, de la CPU, de la ALU)

- **Estructura:** componentes y su interconexión (“foto fija”)
- **Funcionamiento:** dinámica procesamiento información

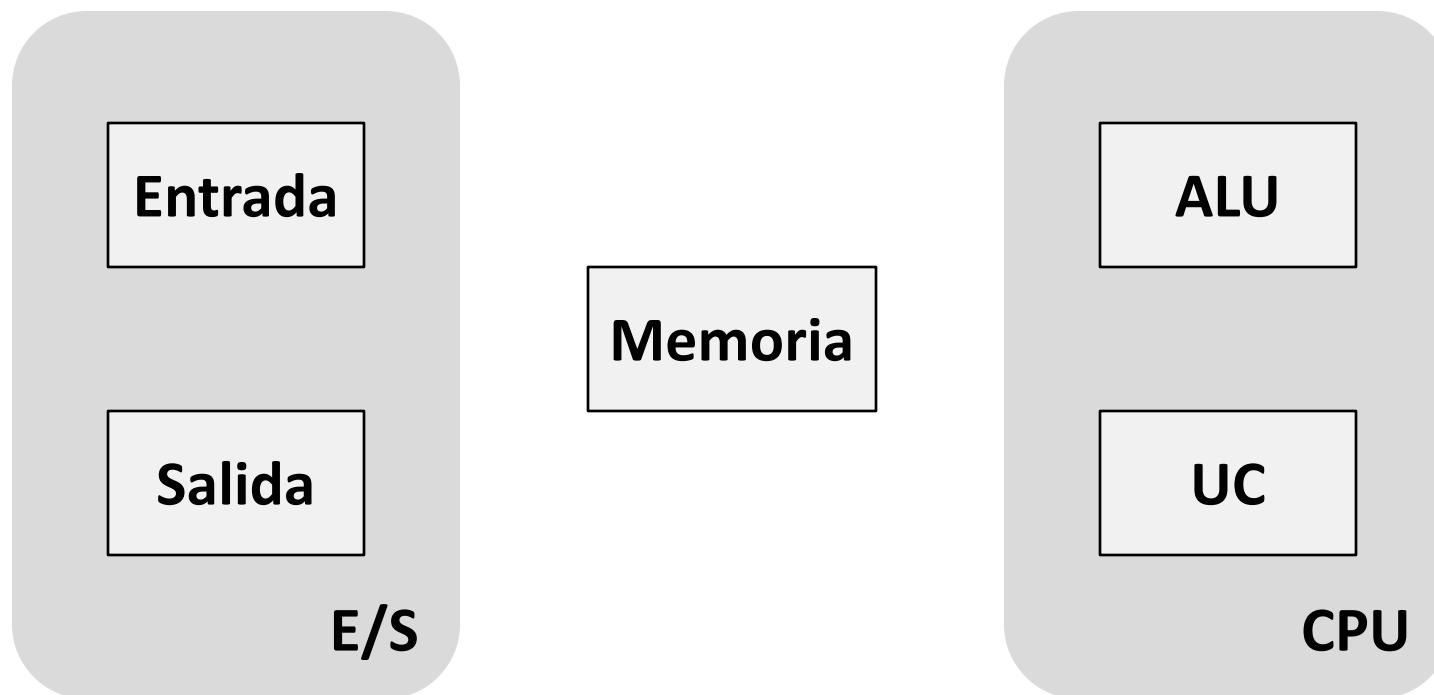
■ Computador (digital): E/S, M, CPU (ALU+UC)

- Computador personal
 - Sobremesa (desktop)
 - Portátil (laptop)
- Estación de trabajo (más prestaciones, gráficos)
- Sistemas de empresa (más CPU y almacenamiento)
- Servidores (bases de datos, gran volumen peticiones)
- Supercomputadores (cálculos científicos)

Introducción

- **Unidades funcionales**
- **Conceptos básicos de funcionamiento**
- **Estructuras de bus**
- **Rendimiento**
- **Perspectiva histórica**

Unidades Funcionales



Unidades Funcionales

■ Arquitectura von Neumann

- Distingue 5 componentes: E/S, M, CPU (ALU+UC)

■ E: codificar / digitalizar / transmitir (lectura)

- teclado, ratón, red, disco, CD...

■ M: almacenar

- programas, datos E, resultados operaciones...

■ CPU: Unidad de procesamiento central

- procesa información E/M ejecutando programa
- ALU: Unidad aritmético-lógica: operaciones
- UC: Unidad de control: controla circuitos

■ S: codificar / almacenar / transmitir (escritura)

- pantalla, impresora, disco, red...

M almacena *instrucciones* y datos

■ Instrucciones máquina

- Transferencia (mov, in, out) M, E/S
- Operaciones (add, and) ALU
- Control (jmp, call, ret, set) UC

■ Concepto de “programa almacenado”

- Determina comportamiento máquina (salvo IRQ)
 - porque instrucciones reproducibles y flujo programa predeterminado

■ Datos

- En memoria, todo son datos
 - interpretado como programa (codop): si leido en etapa captación
 - Compilar, desensamblar: código usado como datos
- Codificación:
 - Instrucciones: codops (codificación en bloque, por extensión, según fabricante)
 - Enteros: binario (complemento a dos), BCD...
 - Alfabéticos: ASCII, EBCDIC...
 - Punto flotante: IEEE-754 simple/doble precisión...

TOC: 1.1 Conceptos básicos. Lenguaje máquina

- El *lenguaje máquina* es el único que entienden los circuitos del computador (CPU). Las instrucciones se forman por bits agrupados en campos:
 - **Campo de código de operación** indica la operación correspondiente a la instrucción.
 - **Campos de dirección** especifican los lugares (o posición) donde se encuentra o donde ubicar los datos con los que se opera.

E/S

■ Entrada:

- codificar información **operador** → M / CPU
 - teclado, ratón/palanca (junto con pantalla), micrófono
- recuperar información **previamente** almacenada
 - HD, CD/DVD, lector tarjetas magnéticas...
- comunicar **ordenadores** entre sí
 - tarjeta de red, módem...

■ Salida:

- codificar información resultado → **operador humano**
 - impresora, pantalla
- almacenar para uso **posterior**
 - HD, CD/DVD...
- comunicar con otros **computadores**
 - red, módem...
- muchos dispositivos son duales E/S (aceptan R/W)

M

■ Memoria:

- Almacenamiento primario (memoria semiconductor)
 - Palabras n bits accesibles en 1 operación básica R/W
 - Longitudes palabra típicas: 16-64bits
 - Muy frecuente: memoria de bytes (asuntos alineamiento, ordenamiento)
 - Accesible aleatoriamente (RAM) por dirección (posición)
 - Bus direcciones, bus datos, bus control (R/W), T_{acceso}
 - Tamaños memoria típicos (PC): 8GB...32GB
 - Tiempos acceso típicos: ~ns (DDR4-2400 19.2GB/s CL15 Lat 12.5ns)
 - » DDR4-2400 → $F_{bus}=1200\text{MHz}$, $T_{cyc}=0.833\text{ns}$, $\text{Lat}_{CAS} \text{ CL15} \rightarrow 12.5\text{ns}$
 - Jerarquía memoria: cache L1, L2 (on-chip), L3, MP
 - Programa almacenado en MP
- Almacenamiento secundario (óptico/magn. E/S)
 - No es memoria von-Neumann, es E/S
 - Fichero swap se considera como parte de la jerarquía memoria

CPU

■ ALU:

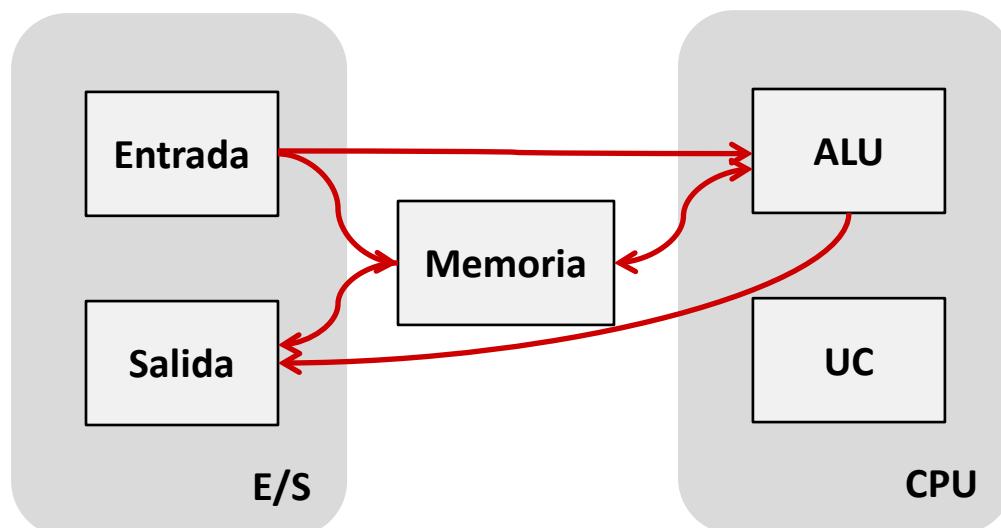
- Componente más rápido del computador (junto con UC)
- **Registros**: almacenamiento más rápido (más que L1)
 - Operandos/Resultado de/a memoria/registros
 - Arquitecturas R/R, R/M, M/M
- Operaciones **aritméticas** (add, mul, div...)
 - Enteras y punto flotante
- Operaciones **lógicas** (and, rol...)
 - Bit a bit

■ UC:

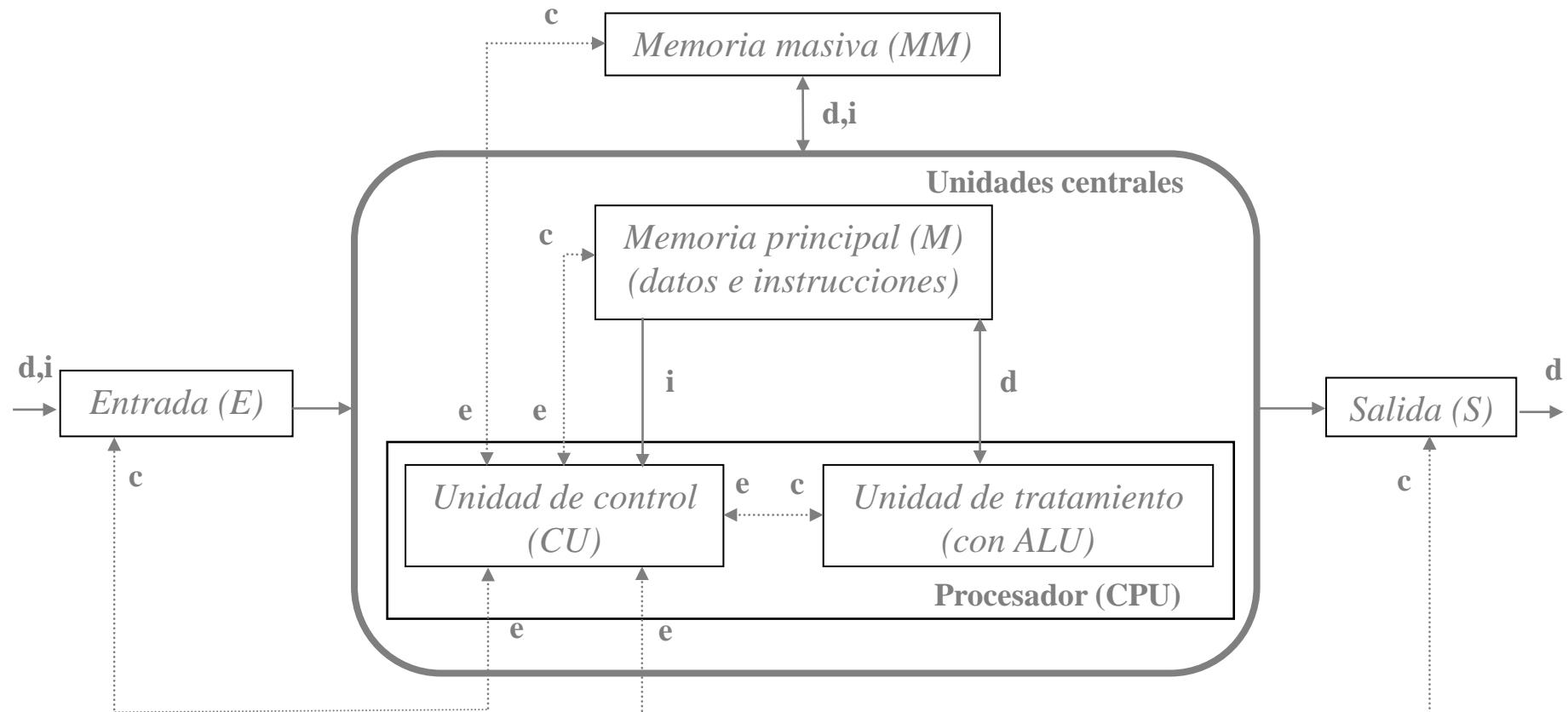
- Componente más rápido del computador (con ALU)
- **Controla** todos los demás circuitos (ALU, M, E/S)
 - Según lo indicado por el programa almacenado en MP
 - Instrucciones transferencia → señales control **M y E/S**
 - Instrucciones aritm/lógicas → señales control **ALU**
 - **Temporización** señales (dirección, datos, R/W)

■ Posibilidades funcionamiento

- Programa E → MP
- Datos E → MP
- Ejecución programa: Datos → ALU → resultados
 - E / M → ALU → S / M
- Resultados → S
- Todo controlado según indique programa MP
 - Interpretado por la UC



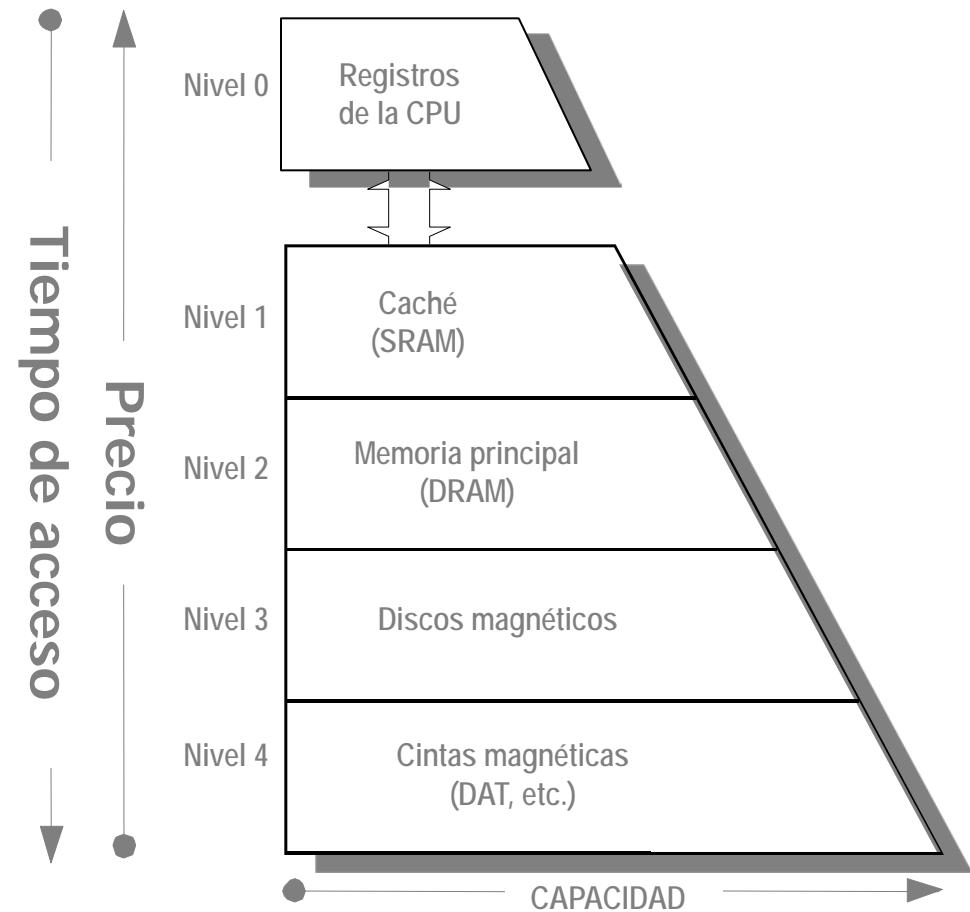
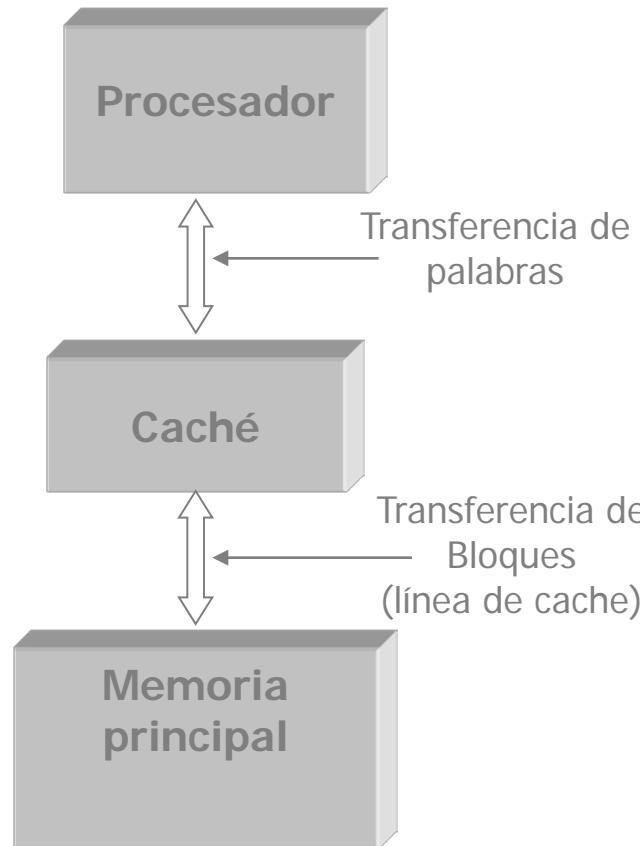
TOC: 2. Unidades funcionales de un computador



d: datos ; i: instrucciones

e: señales de estado c: señales de control

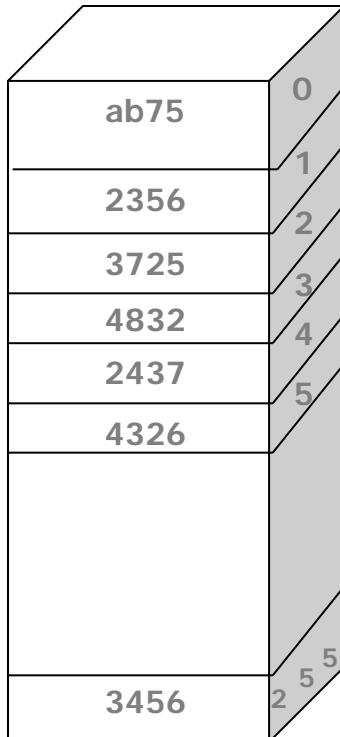
TOC: 2.2 Jerarquía de memoria



sobre Memoria

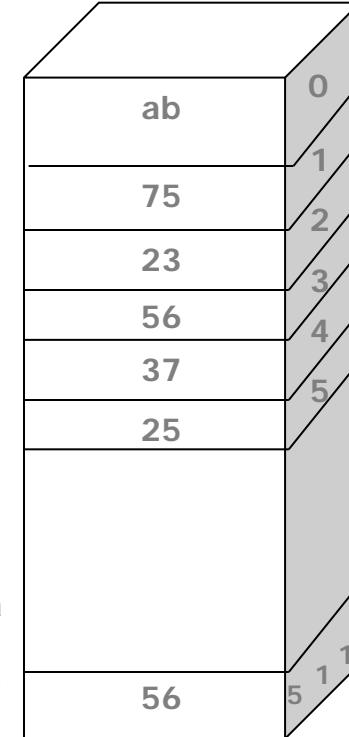
■ Organización en bytes

- ¿tamaño posición M = registro CPU (longitud palabra)?
 - ideal, pero no frecuente
 - típicamente, posiciones 1B (direcciónamiento por bytes)
 - no necesidad empaquetamiento cadenas (strings)
 - Problemas: **alineamiento, ordenamiento**



TOC: hipotética memoria
256 palabras de 16bits
(apuntes TOC §1.2)

misma cantidad de memoria
organizada como 512 bytes
words alineadas, big-endian



sobre Memoria de bytes

■ Ordenamiento en memoria de bytes

- Criterio del extremo menor (**little-endian**)
 - Primero se almacena el byte menos significativo (LSB)
 - LSB en posición M más baja, MSB en posición más alta
- Criterio del extremo mayor (**big-endian**)
 - Primero el MSB (en posición M más baja)

Dirección Contenido

| | |
|-----|----|
| 0 | ab |
| 1 | 75 |
| 2 | 23 |
| 3 | 56 |
| 4 | 37 |
| 5 | 25 |
| . | ⋮ |
| . | ⋮ |
| 1FF | 56 |

mismo contenido
en big-endian

Dirección Contenido

| | |
|-----|----|
| 0 | 75 |
| 1 | ab |
| 2 | 56 |
| 3 | 23 |
| 4 | 25 |
| 5 | 37 |
| . | ⋮ |
| . | ⋮ |
| 1FF | 34 |

mismo contenido
en little-endian

sobre Memoria de bytes

■ Alineamiento en memoria de bytes

- Palabra de n bytes alineada \Leftrightarrow comienza en dirección múltiplo de n
 - Algunas CPUs requieren alineamiento accesos M (si no, bus error)
 - Otras acceden más rápido si acceso alineado
 - palabra no cruza línea de cache, página, etc

| Dirección | Contenido |
|-----------|-----------|
| 0 | 01 |
| 1 | 00 |
| 2 | 00 |
| 3 | 00 |
| 4 | FE |
| 5 | FF |
| 6 | FF |
| 7 | FF |
| . | . |
| . | . |
| . | . |

palabra 16bits
valor 1, alineada

palabra 32bits
valor -2, alineada

(máqu)

| Dirección | Contenido |
|-----------|-----------|
| 0 | AB |
| 1 | 75 |
| 2 | FF |
| 3 | 00 |
| 4 | 00 |
| 5 | 00 |
| 6 | 48 |
| 7 | FF |
| . | . |
| . | . |
| | . |

pal. 32bits, valor 255, no alineada

byte suelto valor -1

little-endian)

Clasificaciones m/n y pila-acumulador-RPG

■ Tipos de CPU según operandos de las instrucciones ALU

- también suele afectar a operandos instrucciones transferencia

■ Clasificación m/n

- Operaciones ALU admiten n operandos, m de ellos de memoria

■ Combinaciones típicas

- Máquinas **pila**: 0/0

- Repertorio: Push M, Pop M, Add, And...

- Máquinas de **acumulador**: 1/1

- Operando implícito: registro acumulador A (más rápido que M)

- Repertorio: Load M, Store M, Add M, And M...

- Máquinas de **RPG** (Registros de Propósito General): (x/2, x/3)

- Múltiples “acumuladores”

- Repertorio: Move R/M R/M, Add R/M R/M R/M

RPG: Clasificación R/M

■ Para máquinas RPG

- Arquitecturas **R/R** (registro-registro)
 - 0/2, 0/3
 - Add R1, R2, R3
 - típico de RISC
- Arquitecturas **R/M** (registro-memoria)
 - 1/2, 1/3 (2/3 poco frecuente)
 - Add R1, A
 - típico de CISC
- Arquitecturas **M/M** (memoria-memoria)
 - 2/2, 3/3 (poco frecuente)
 - Add A, B
 - permite operar directamente en memoria
 - demasiados accesos memoria por instrucción máquina

sobre Repertorios

■ ISA

- Arquitectura del Repertorio (Instruction Set Architecture)
- Registros, Instrucciones, Modos de direccionamiento...

■ RISC

- Comput. repertorio reducido (Reduced Instruction Set Computer)
- 0/2, 0/3
- Pocas instrucciones, pocos modos, formato instrucción sencillo
- UC sencilla → muchos registros

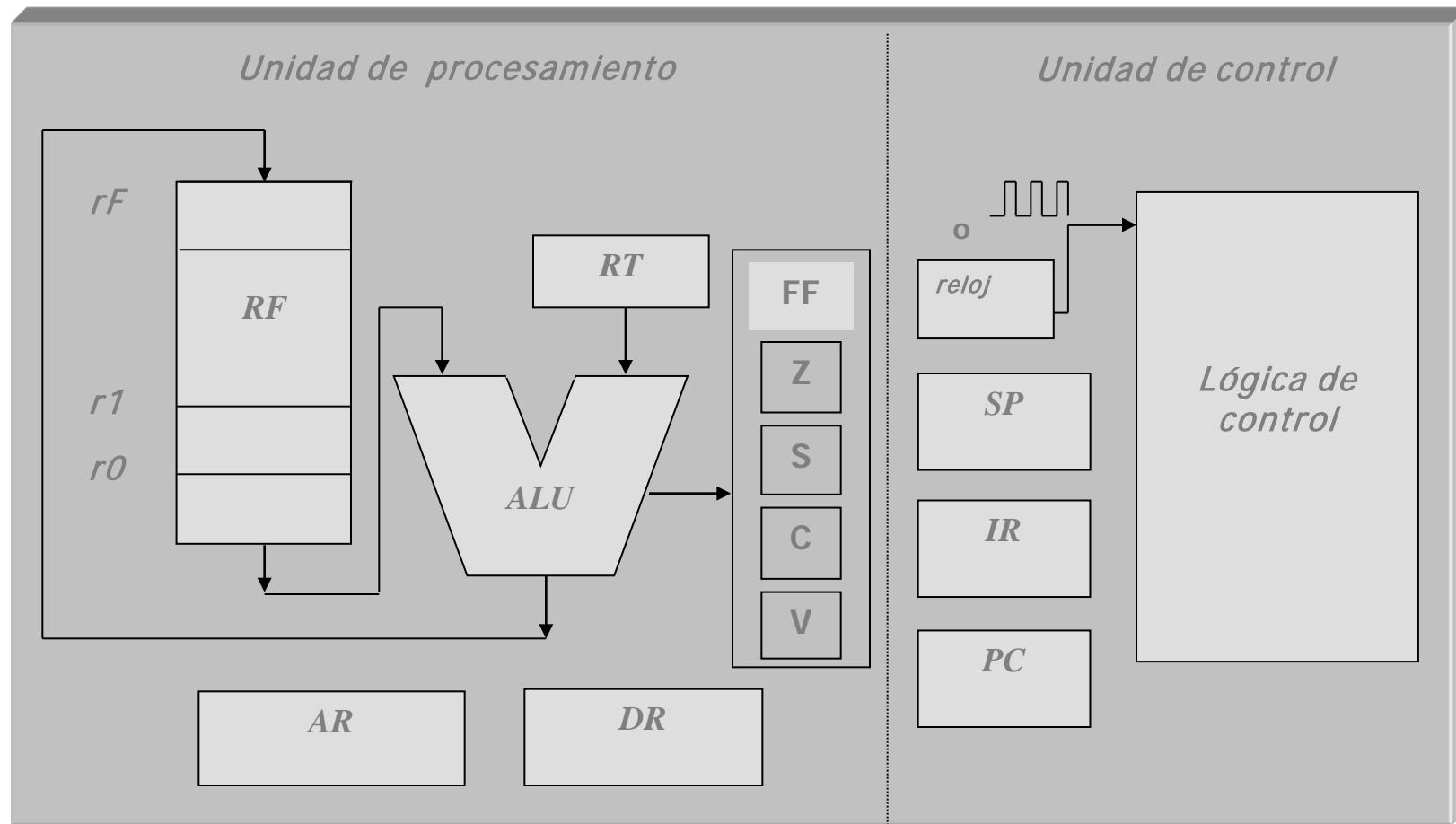
■ CISC

- Comput. repertorio complejo (Complex Instruction Set Computer)
- 1/2, 1/3 (y resto)
- “más próximos a lenguajes alto nivel”
- Debate RISC/CISC agotado, diseños actuales mixtos

Introducción

- Unidades funcionales
- **Conceptos básicos de funcionamiento**
- Estructuras de bus
- Rendimiento
- Perspectiva histórica

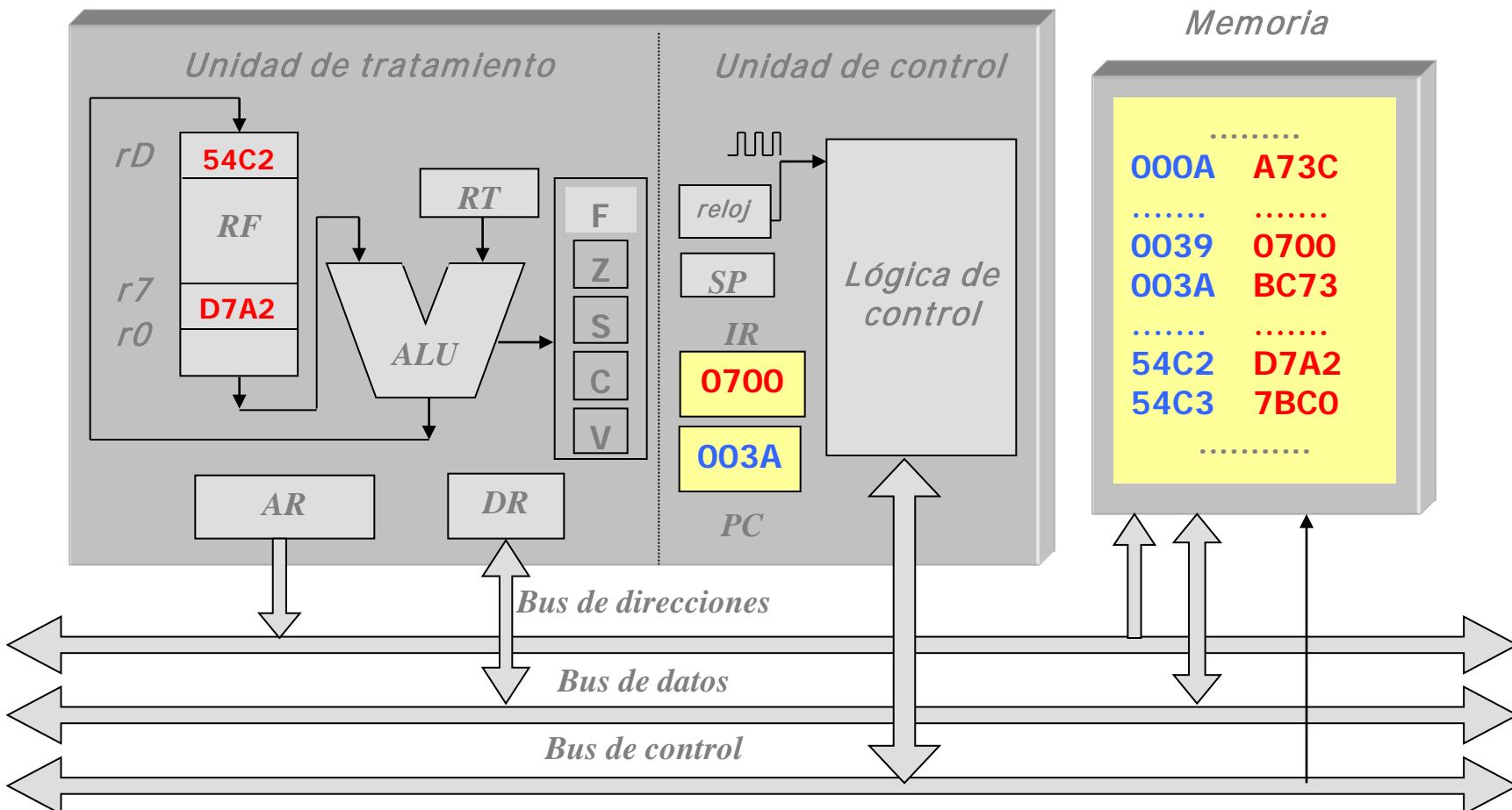
TOC: 2.1 Elementos internos de un procesador



TOC: 2.1 Ejecución de Mov (rD), r7

| Direc. | Contenidos | Fase | Microoperación | Contenidos de los registros | | | | | |
|--------|------------|--------------------------|----------------|-----------------------------|------|------|------|------|--|
| | | | | PC | IR | AR | DR | r7 | |
| 0000 | 7AC4 | Valores iniciales | | | | | | | |
| 0007 | 65C9 | Captación de instrucción | AR ← PC | 0039 | | 0039 | | | |
| | 0700 | | DR ← M(AR) | 0039 | | 0039 | 0700 | | |
| | 607D | | IR ← DR | 0039 | 0700 | 0039 | 0700 | | |
| | 2D07 | | PC ← PC+1 | 003A | 0700 | 0039 | 0700 | | |
| | C000 | | AR ← rD | 003A | 0700 | 54C2 | 0700 | | |
| 54C2 | D7A2 | Ejecución de instrucción | DR ← M(AR) | 003A | 0700 | 54C2 | D7A2 | | |
| | 3FC4 | | r7 ← DR | 003A | 0700 | 54C2 | D7A2 | D7A2 | |
| rD | 54C2 | | | | | | | | |

TOC: 2.1 Situación después de la ejecución



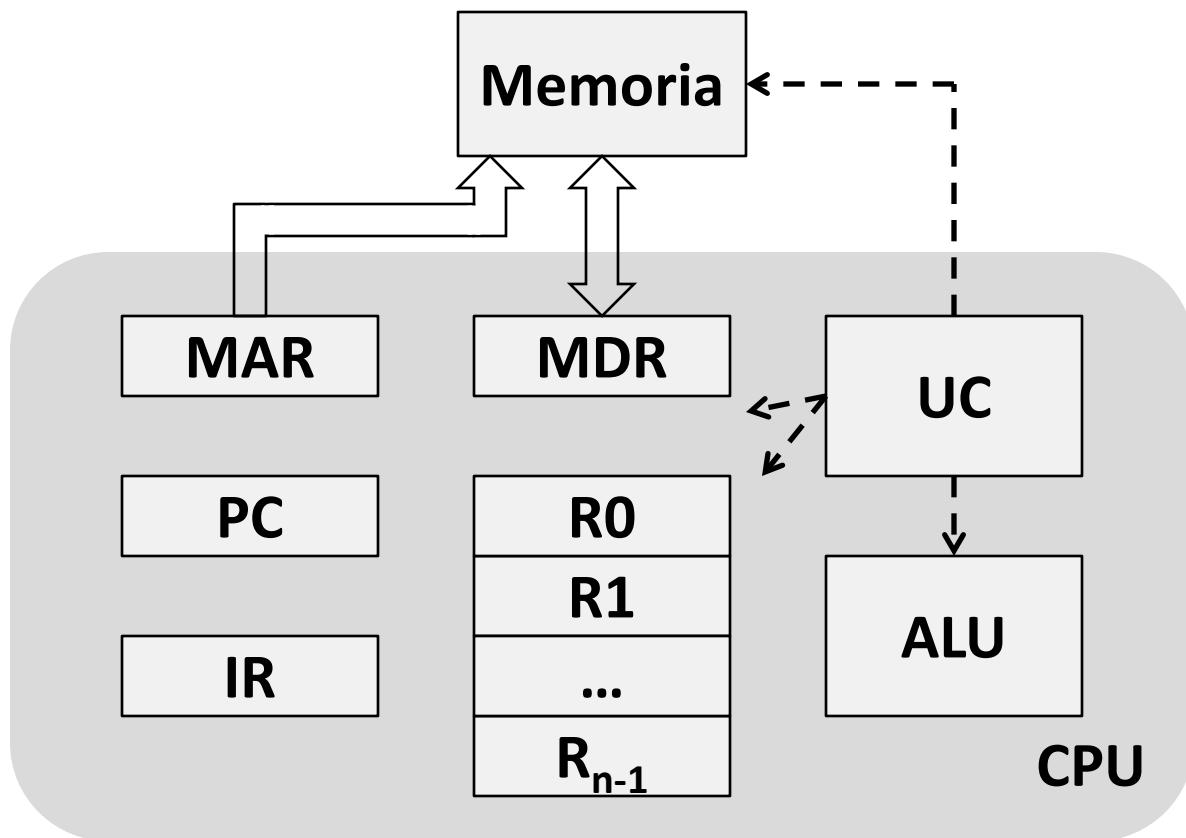
Ciclo ejecución instrucciones: fases

- Programa en MP
- CPU (UC) tiene PC (*program counter*)
 - posición MP de la siguiente instrucción
 - *Captación*: leer dicha posición $IR \leftarrow M[PC]$
 - Usando MAR/MDR (Memory Address/Data Register), Instruction Register (IR)
 - Se interpreta como codop
 - Incrementar PC
 - *Decodificación*: desglosar codop/operandos(regs)
 - Posible etapa *Operando(M)*: captar dato/ incrementar PC
 - *Ejecución*: llevar datos ALU / operar
 - *Almacenamiento*: salvar resultado regs / MP
 - Nombres en inglés:
 - Fetch, Decode, Operand, eXecute, Write/Store

Ciclo ejecución instrucciones

■ Pensar tareas realizadas por UC para ejecutar instrucción

- Por ejemplo: Add A, R0
 - $M[A] + R0 \rightarrow R0$
- Detalles en [HAM03] Cap-1.3
- Ejercicios similares en TOC §2.1



Add A, R0

- $M[POS_A] + R0 \rightarrow R0$
 - Ensamblador traduce p.ej: $POS_A = 100$
 - Valor anterior R0 perdido, el de POS_A se conserva
 - Arquitectura R/M
- **Pasos básicos de la UC**
 - PC apunta a posición donde se almacena instrucción
 - **Captación:** $MAR \leftarrow PC$, Read, $PC \leftarrow PC + 1$, $T_{acc} \leftarrow$ MDR \leftarrow bus, $IR \leftarrow MDR$
 - **Decodificación:** se separan campos instrucción
 - Codop: ADD $mem + reg \rightarrow reg$
 - Dato1: 100 direcciónamiento directo, habrá que leer $M[100]$
 - Dato2: 0 direcciónamiento registro, habrá que llevar R0 a ALU
CPUs con longitud instrucción variable – dirección (100) en siguiente palabra
 - **Operando:** $MAR \leftarrow 100$, Read, $T_{acc} \leftarrow$ MDR, $ALU_{in1} \leftarrow MDR$
 - **Ejecución:** $ALU_{in2} \leftarrow R0$, add, T_{alu}
 - **Almacenamiento:** $R0 \leftarrow ALU_{out}$

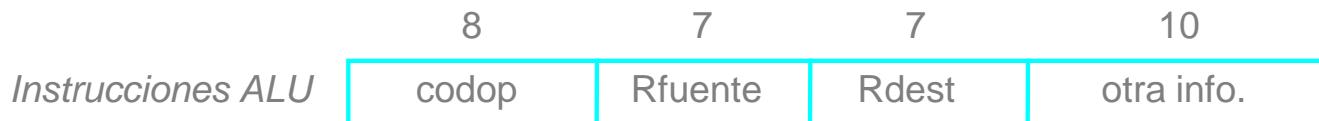
Otras consideraciones

- Arquitectura M/M: **varias captaciones** operando
 - PC++, si las direcciones ocupan más posiciones M
- Arquitectura R/M: cuando **resultado en M** (Add R0, A):
 - acceso memoria adicional (Write): $M[\text{MAR}] \leftarrow \text{MDR} \leftarrow \text{ALU}_{\text{out}}$
 - UC activa señal Write
- Arquitectura R/R: **varias instrucciones** (Load A, R1 / Add R1, R0)
 - efecto colateral: R1 perdido
 - ventaja: CPU más simple, veloz, pequeña (longitud/formato instrucción)
- Ciclo interrumpido por **IRQ**→ISR
 - mecanismo subrutina / salvar contexto (PC/estado)
 - salvo eso, comportamiento totalmente predeterminado por programa
- CPU completa (+L1+L2...+L3) en 1 chip VLSI
 - La CPU nunca lee de memoria un dato aislado
 - **lee de cache**
 - si hay fallo, se trae un bloque entero
 - se explica en clase así por motivos académicos

sobre Formatos de Instrucción

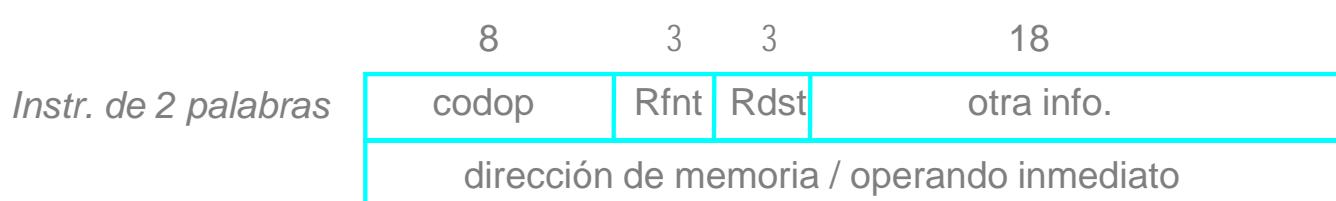
■ RISC

- Pocas instrucciones, pocos modos, muchos registros, 0/2-0/3
 - formato instrucción sencillo, tal vez sólo 2-3: transferencia, ALU, ctrl
 - ej: formato instrucciones ALU de un RISC 32bits 128regs tipo 0/2



■ CISC

- Muchas instrucciones y modos, menos registros, 1/2-1/3 (y resto)
 - varios formatos de instrucción, distintas longitudes, codops long. var. también



Formatos de Instrucción

■ ejemplo: IA-32 (Intel 64)

- Instrucciones de longitud variable, 1-15 bytes (memoria de bytes)
 - prefijos modificar detalles de algunas instrucciones
 - **codop** de 4bits a 3B + 3bits (campo reg en ModRM)
 - Mod-R/M modo de direccionamiento (5 bits)
 - Reg para indicar registro (hasta 8 regs)
 - SIB para indicar 2 registros y escala índice (x1,x2,x4,x8)
 - desplazamiento 32bits dirección memoria (u offset)
 - inmediato 32bits valor operando

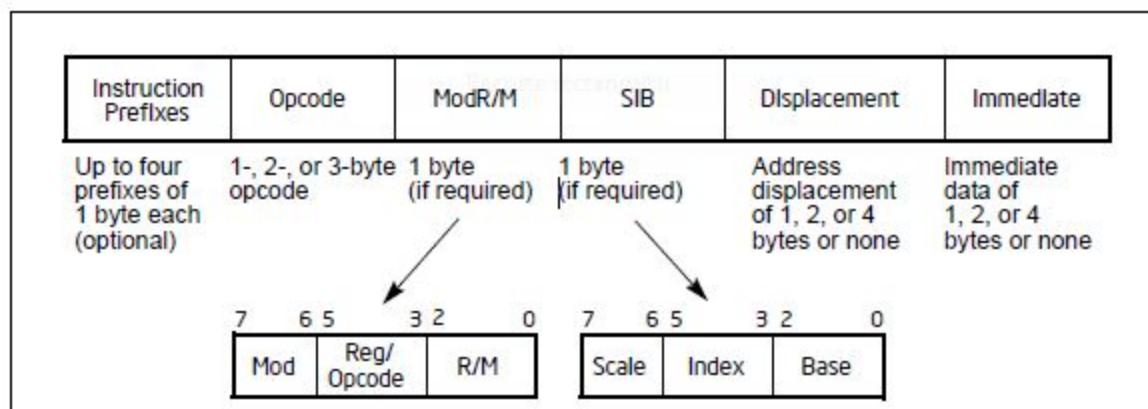


Figure 2-1. Intel 64 and IA-32 Architectures Instruction Format

Modos de Direccionamiento

- un número acompañando a un codop puede significar muchas cosas
 - según el formato de instrucción, la instrucción concreta, etc
- cada operando de la instrucción tiene su modo de direccionamiento

■ Inmediato (ej: \$0, \$variable)

- El número es el valor del operando

■ Registro (ej: %eax, %ebx...)

- El número es un índice de registro (ese registro es el operando)

■ Memoria (en general: disp(%base,%index,scale))

- instrucción lleva índices de registros y/o desplazamiento (dirección memoria)
- La dirección efectiva (EA) es la suma de todos ellos. El operando es M[EA].

| | | |
|---------------------------|-------------------------------|--------------------------|
| ■ Directo | sólo dirección (disp) | op=M[disp] |
| ■ Indirecto a través reg. | sólo registro (reg) | op=M[reg] |
| ■ Relativo a base | registro y desplazamiento | op=M[reg+disp] |
| ■ Indexado | índice (x escala) y dirección | op=M[disp + index*scale] |
| ■ Combinado | todo | op=M[disp+base+idx*sc] |

ej: modos IA-32

a veces puede ser ventajoso
+ instrucciones -tamaño

inmediato ≠ directo

resto modos indirectos

Código fuente ASM:

```
.section .text
_start: .global _start

mov    $0, %eax      # inm - registro
xor    %ebx, %ebx    # reg - registro
inc    %ebx          # reg
mov    $array, %ecx  # inmediato - reg
mov    array, %edx   # directo - reg

mov    (%ecx)        , %edx # indirecto
add    (%ecx,%ebx,4), %edx # combinado
add    array( ,%ebx,4), %edx # indexado
mov    -8(%ebp)      , %edx # rel.base
```

Desensamblado del ejecutable:

Disassembly of section .text:

08048074 <_start>:

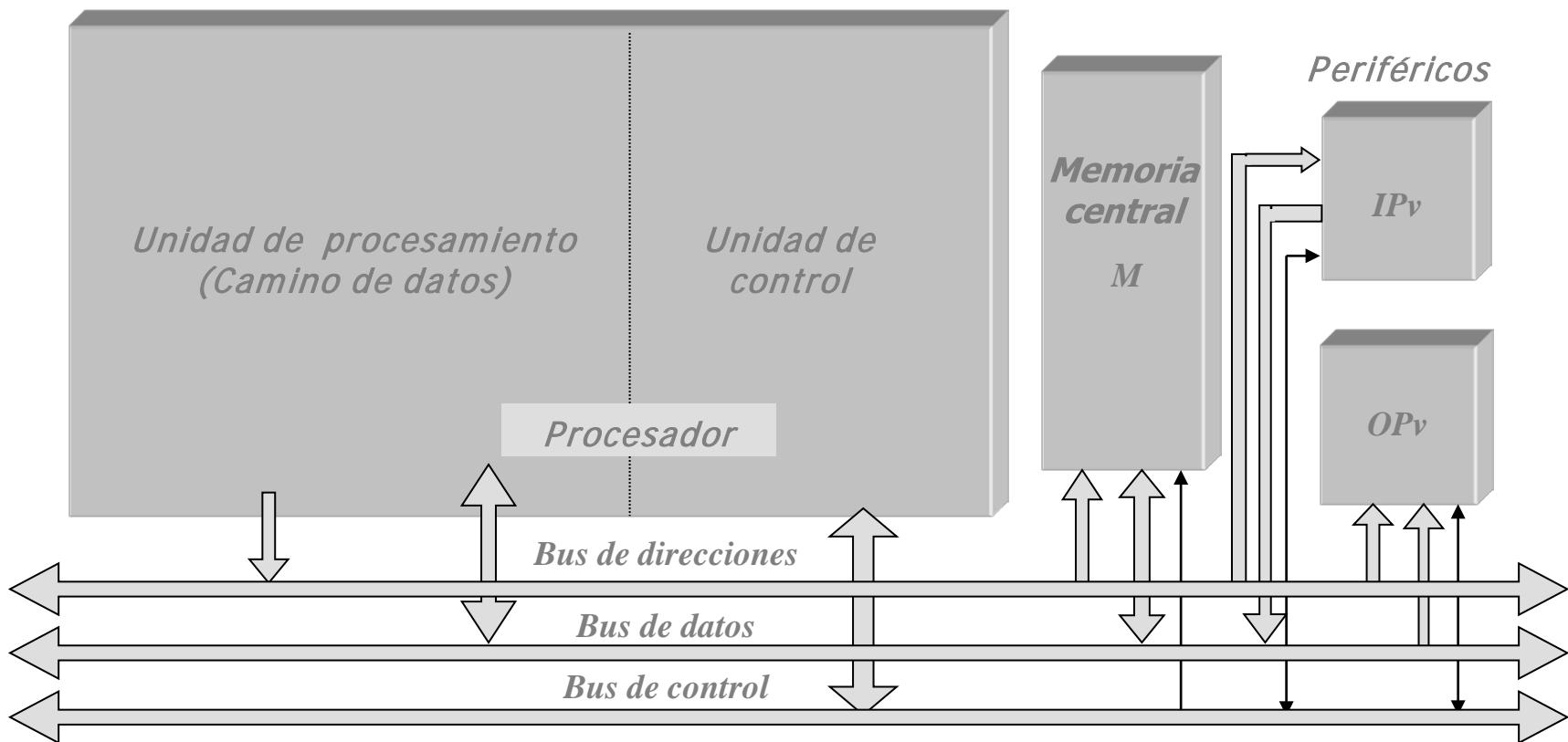
| | | |
|----------|----|-------------------|
| 8048074: | b8 | 00 00 00 00 |
| 8048079: | 31 | db |
| 804807b: | 43 | |
| 804807c: | b9 | 98 90 04 08 |
| 8048081: | 8b | 15 98 90 04 08 |
| 8048087: | 8b | 11 |
| 8048089: | 03 | 14 99 |
| 804808c: | 03 | 14 9d 98 90 04 08 |
| 8048093: | 8b | 55 f8 |

| | |
|-----|------------------------|
| mov | \$0x0,%eax |
| xor | %ebx,%ebx |
| inc | %ebx |
| mov | \$0x8049098,%ecx |
| mov | 0x8049098,%edx |
| mov | (%ecx),%edx |
| add | (%ecx,%ebx,4),%edx |
| add | 0x8049098(%ebx,4),%edx |
| mov | -0x8(%ebp),%edx |

Introducción

- **Unidades funcionales**
- **Conceptos básicos de funcionamiento**
- **Estructuras de bus**
- **Rendimiento**
- **Perspectiva histórica**

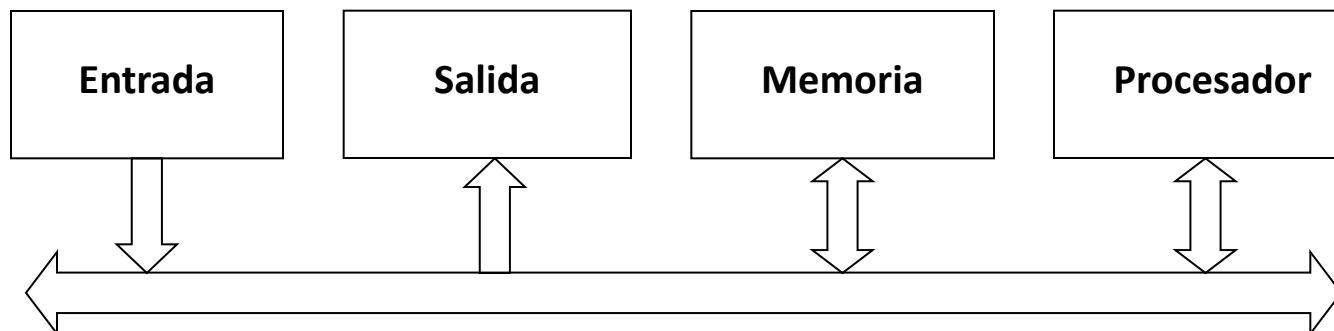
TOC: 2.1 Interconexión de las distintas unidades



Estructuras de bus

■ Justificación buses (paralelos):

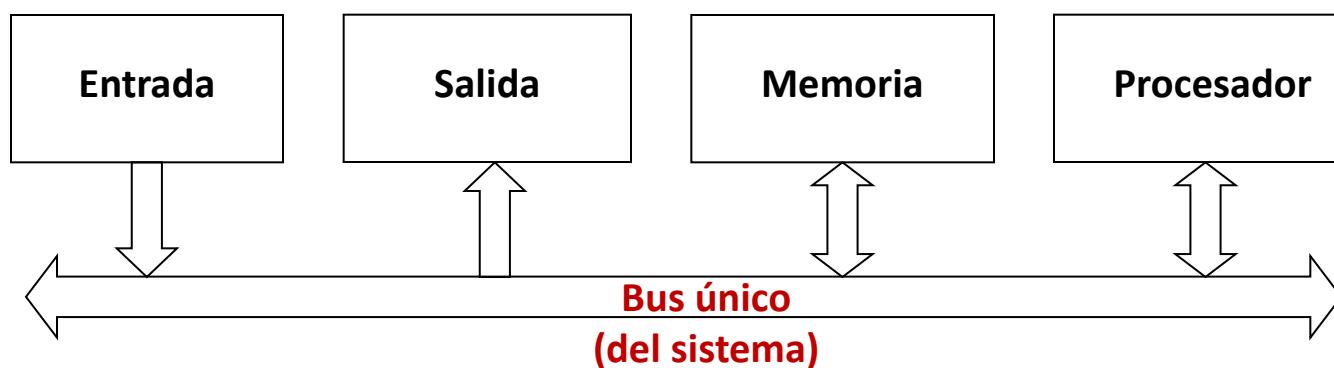
- E/S, M, CPU deben conectarse para pasar datos
- Representación binaria / velocidad transferencia
 - palabras n bits M/ALU → bus **datos** n bits
 - direcciones m bits M → bus **addr** m bits
 - bus **control** para líneas UC (R/W, etc)



Estructuras de bus

■ Bus único

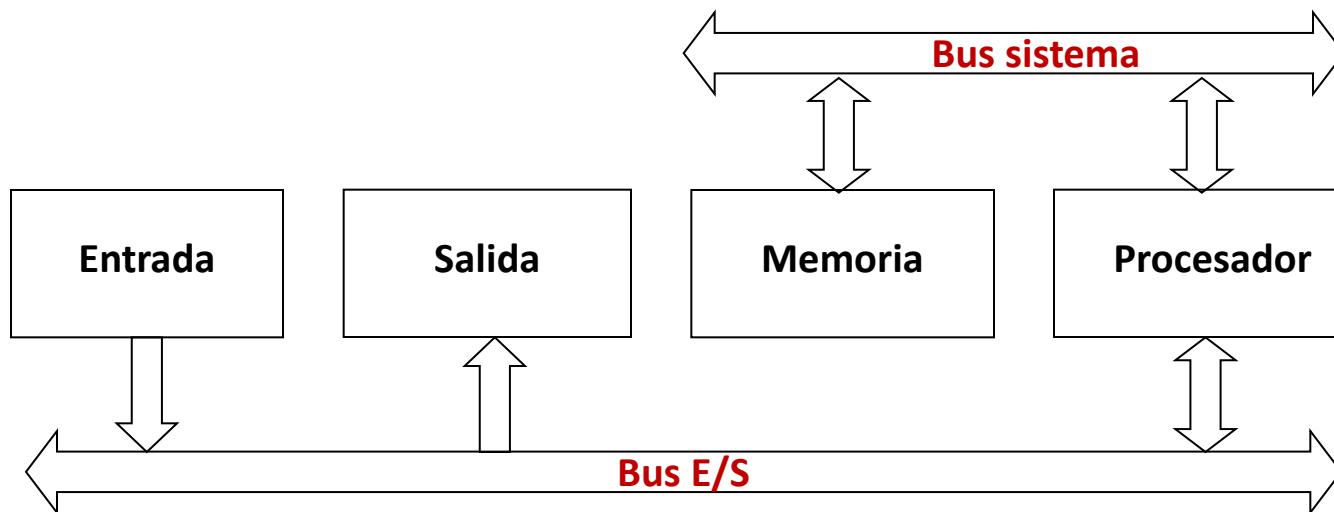
- CPU escribe bus dirección y control R/W
 - también escribe bus datos si Write
 - puede haber señales IOR/W separadas de MemR/W
- E/S/M comprueban si es su dirección
 - sólo en ese caso se conectan al bus de datos
 - evitar cortocircuito bus datos
- Ventaja: sencillez, bajo coste, flexibilidad conexión
 - fácil añadir más dispositivos
 - posibilidad líneas control arbitraje para varios master



Estructuras de bus

■ Buses múltiples

- típicamente: bus sistema (CPU-M) y bus E/S
 - también: múltiples buses E/S
 - separar dispositivos según velocidades
 - incluso: doble bus sistema
 - memoria datos/programa (arquitectura Harvard)
- Ventajas: uno más rápido, ambos funcionan en paralelo
- Inconveniente: coste, complejidad



Adaptación de velocidades

■ Velocidad componentes

- CPU > Memoria >> E/S

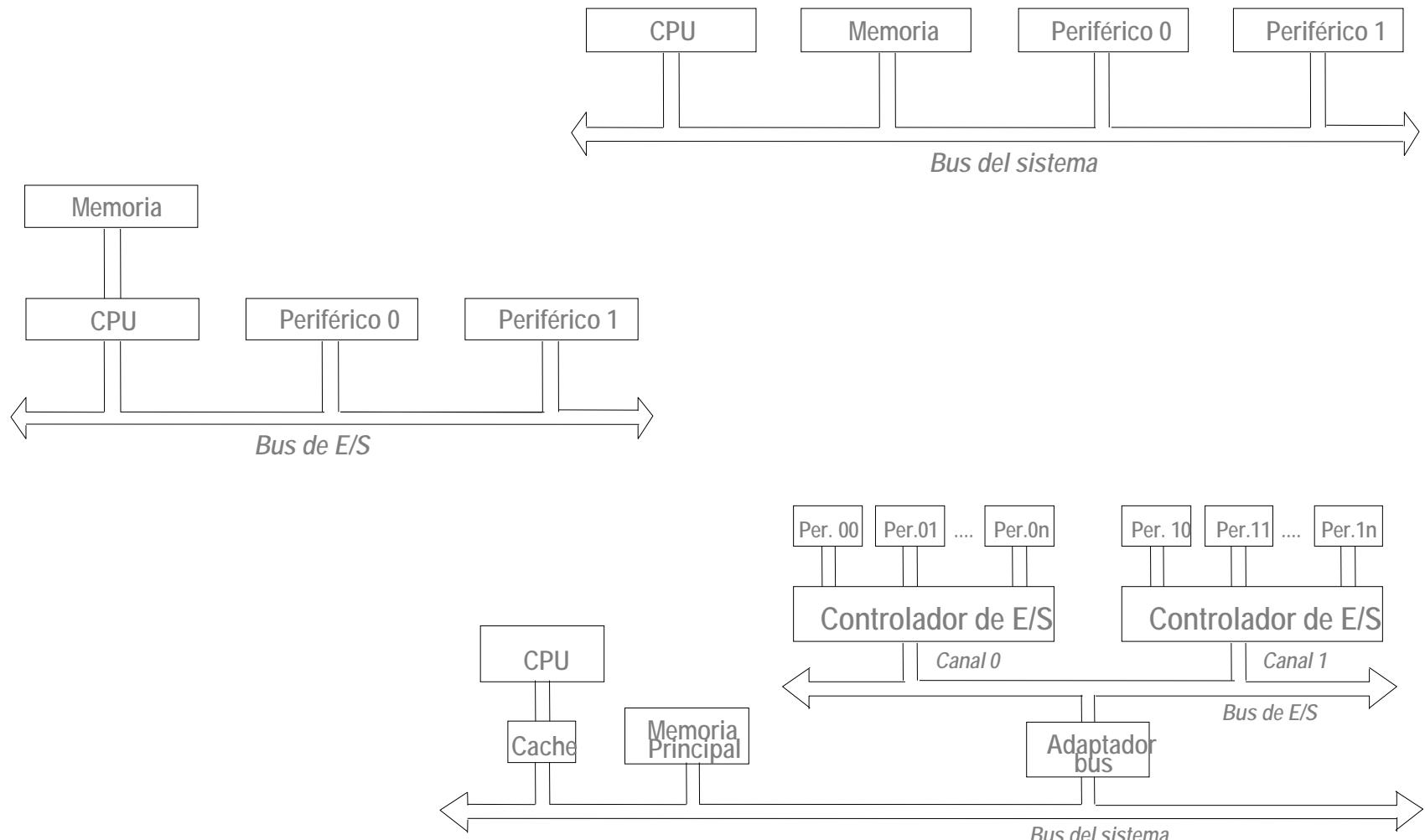
■ Estados de espera:

- alargar ciclo bus si no se activa señal RDY (bus control)
- permite conectar periféricos lentos a bus único

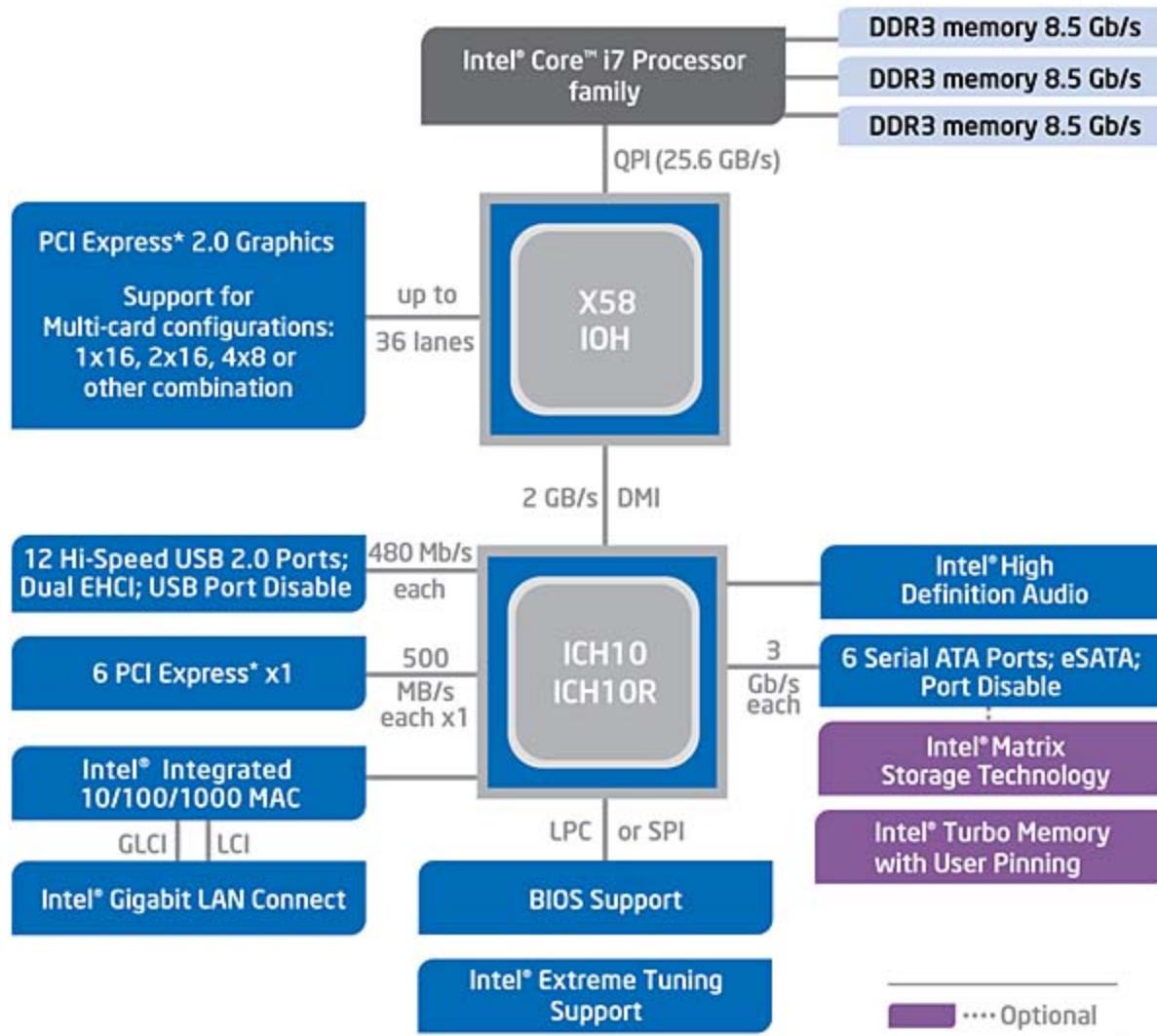
■ Buffers/IRQ:

- dispositivo lento almacena datos en buffer rápido
 - evita retrasar CPU con estados de espera
 - CPU se dedica a otra tarea mientras tanto
- transferencia CPU a velocidad buffer (normal Memoria)
- Write: CPU escribe buffer, dispositivo genera IRQ al final
 - Ej: impresora
- Read: CPU encarga lectura, dispositivo hace IRQ cuando listo
 - Ej: escáner

TOC: 2.4 Estructuras básicas de interconexión



TOC: 2.4 Estructuras básicas de interconexión



Decodificación

■ Evitar cortocircuito bus datos

- Suponer por ejemplo que CPU es único dispositivo **activo** del bus
 - es decir, que puede escribir bus Addr. y Ctrl.
 - cuando lo hace, se convierte en **maestro** del bus
 - Luego veremos multiprocesadores, controladores DMA, etc
 - varios activos requiere arbitraje para escoger maestro
- demás dispositivos **pasivos**
 - sólo “escuchan” bus Addr, no pueden escribir, sólo leer
 - Cuando la CPU les habla, se convierten en **esclavos**
 - Es decir, se conectan al bus de datos y obedecen la orden R/W
- #bits bus Addr. determina el “**espacio de memoria**”

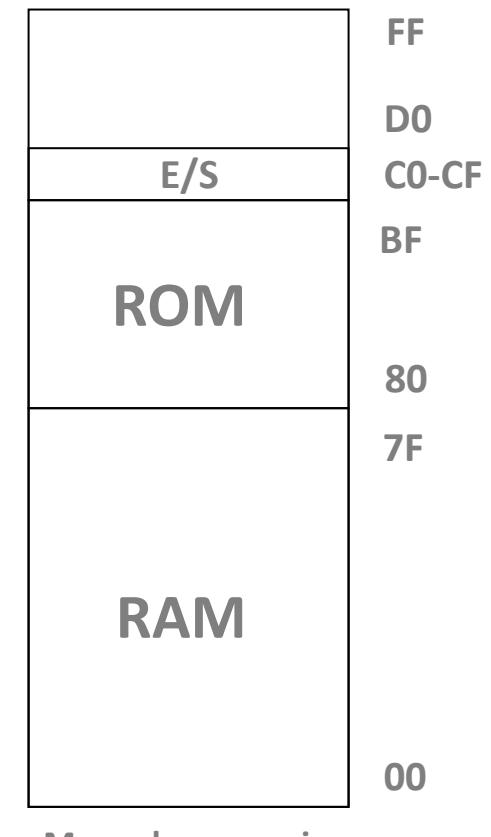
■ Mapa de Memoria

- Dibujo de dónde está cada dispositivo en espacio Memoria
- E/S puede ser “mapeada a memoria” o en espacio E/S separado

Decodificación

■ Ej: diseñar mapa memoria para

- 1 CPU 8bits
 - 8bits Addr. A7...A0
 - 8bits Data: D7...D0
- 1 RAM 128 Bytes
 - decodificada en 0...127
- 1 ROM 64 Bytes
 - a continuación
- 1 Puerto Serie 16 Regs
 - 16 puertos de 8 bits
 - a continuación

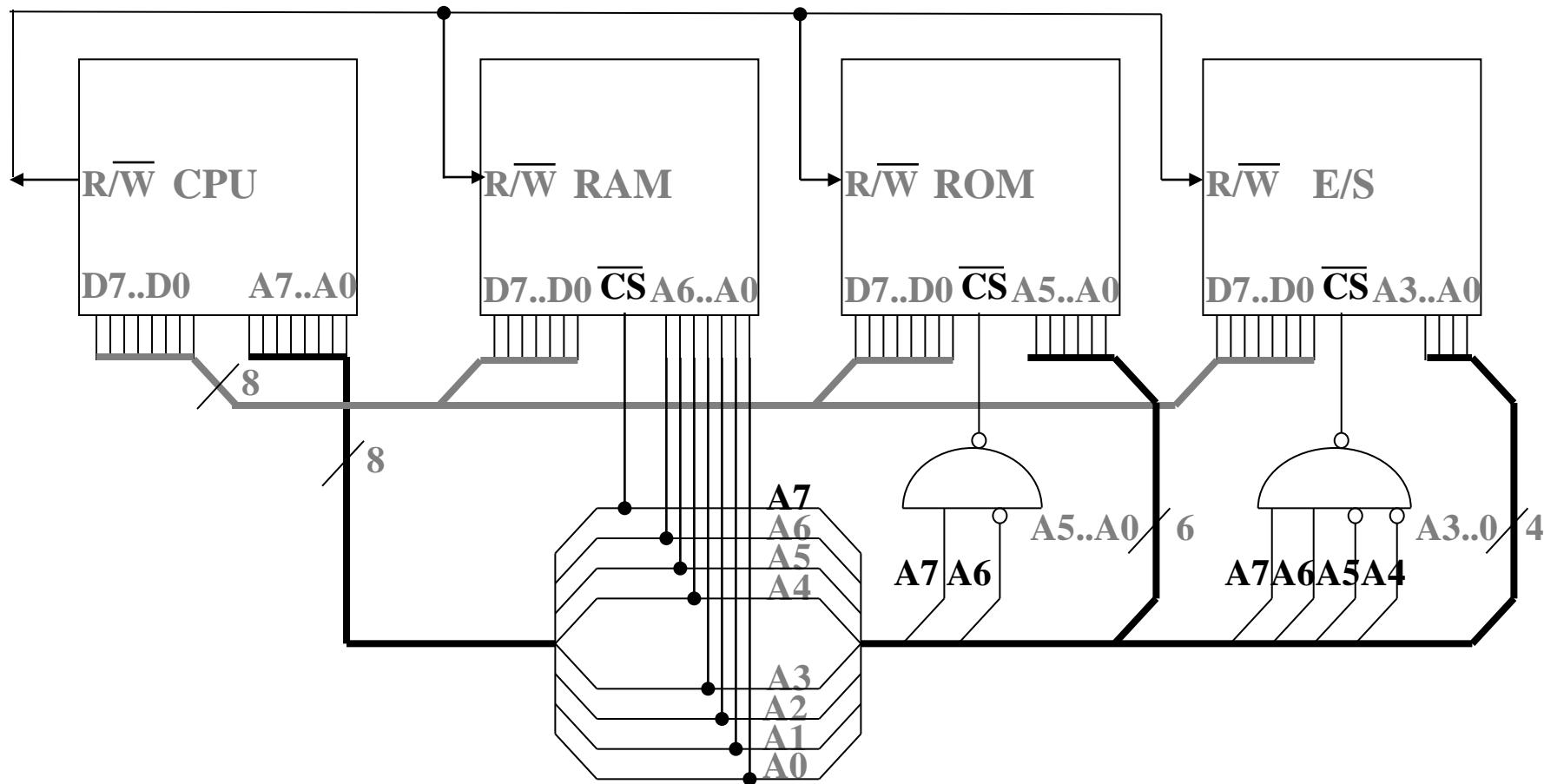
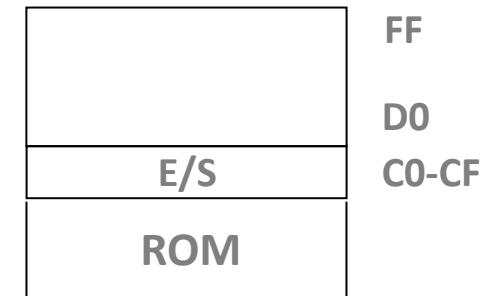


Mapa de memoria

(ejemplo académico, realmente no existen tamaños tan pequeños)

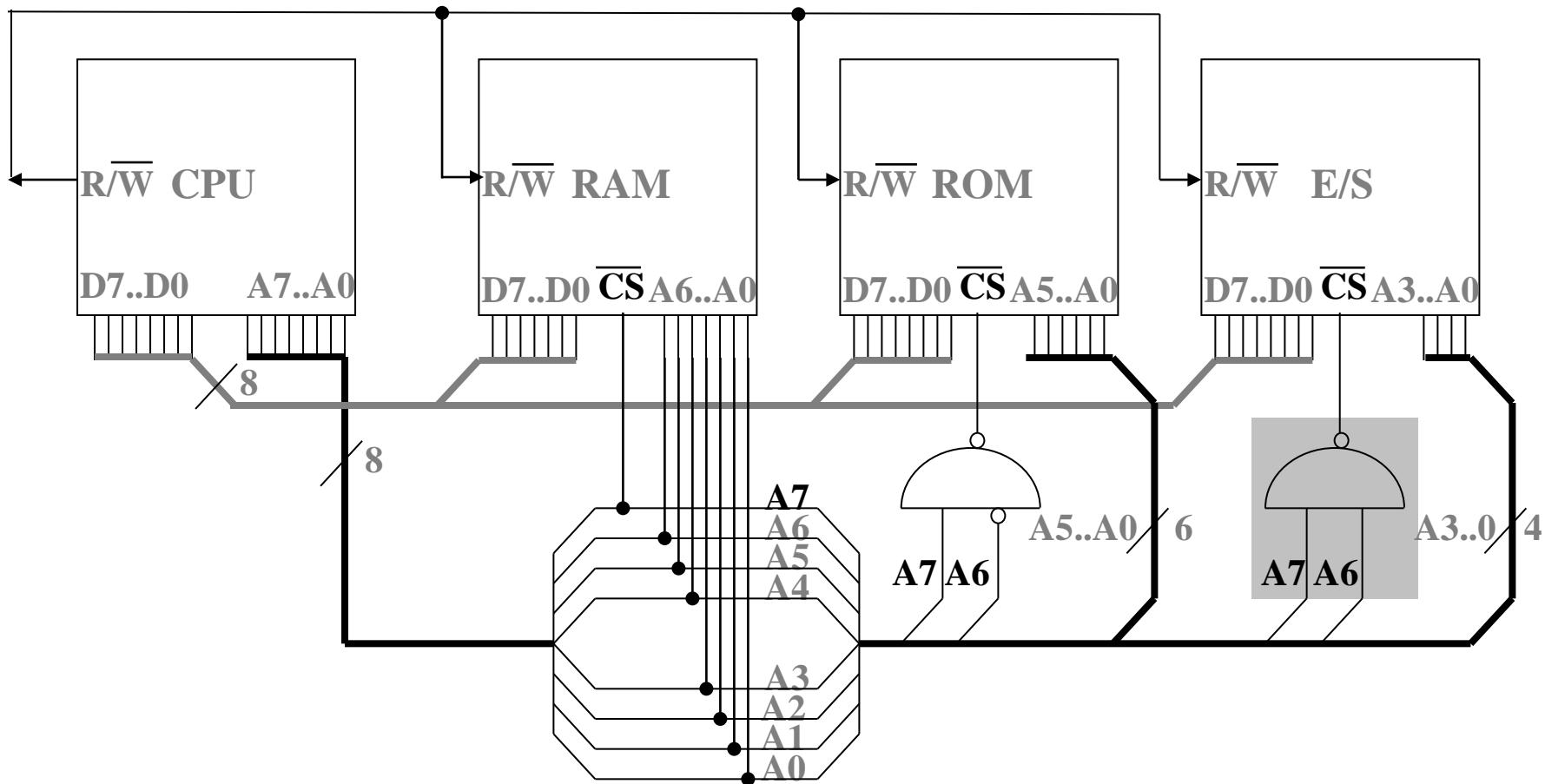
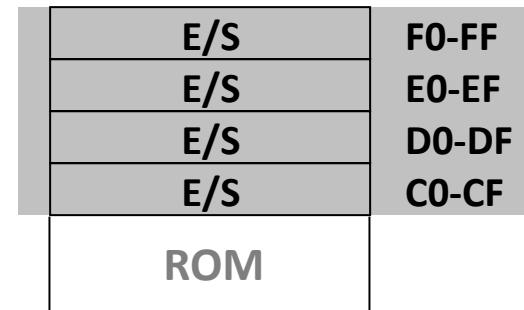
Decodificación completa

- se usan **todos los bits** Addr.
 - MSB decodifican el dispositivo/módulo (CS)
 - LSB direccionan dentro del dispositivo (Addr)



Decodificación parcial

- algunos (m) bits Addr. sin usar
 - El dispositivo aparece repetido 2^m veces en Memoria



Software de sistema

■ Cómo conseguir crear programa → MP → ejecutar

- Software de sistema implicado:
 - Shell (intérprete comandos): recibe órdenes usuario
 - EXEC: llamada para cargar y ejecutar aplicación
 - Editor: permite crear código fuente (y archivar!)
 - Compilador / Enlazador: código objeto / ejecutable
 - Sistema de ficheros (crear, copiar, abrir, leer)
 - desde Shell / desde programa usuario
 - Sistema E/S

■ Cómo se consigue encender → arrancar SO

- soporte hardware: dirección de bootstrap
- [Boot-P]ROM en espacio memoria apuntado
- Bootloader primario, carga arranque HD/FD/CD...
- Bootloader secundario (menú escoger SO, etc)...

Software de sistema

■ Llamadas al sistema (ej: aplicación lee fichero/calcula/imprime)

- usuario teclea nombre aplicación → **EXEC**
 - el shell invoca EXEC, proporcionando nombre fich.
- EXEC carga aplicación HD → M, pasa control
 - el propio SO proporciona zona M cargar aplicación
 - EXEC retorna a aplicación, y ella retornará a shell
- aplicación invoca **OPEN/READ/CLOSE**
 - proporciona zona memoria donde leer contenido
- aplicación calcula resultado, invoca **PRINT/EXIT**
 - proporciona datos a imprimir / código retorno

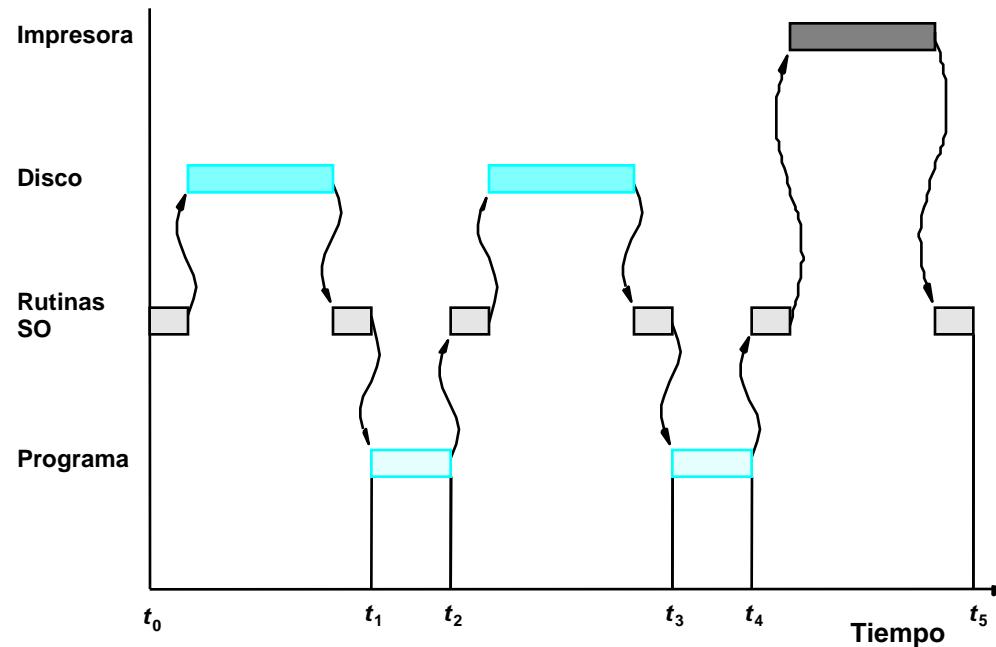
■ SO gestiona recursos (especialmente multiuser/multitask)

- ej: solapar E/S final con carga siguiente tarea
- ej: conmutar proceso en cuanto haga E/S

Software de sistema

■ Pensar tareas realizadas por SO para ejecutar aplicación

- Por ejemplo: leer datos HD, cálculos, imprimir resultados
- Pensar entonces cómo solapar varias de esas aplicaciones
- Detalles en [HAM03] Cap-1.5



Introducción

- Unidades funcionales
- Conceptos básicos de funcionamiento
- Estructuras de bus
- Rendimiento
- Perspectiva histórica

Rendimiento

■ Medida definitiva: **tiempo ejecución programa**

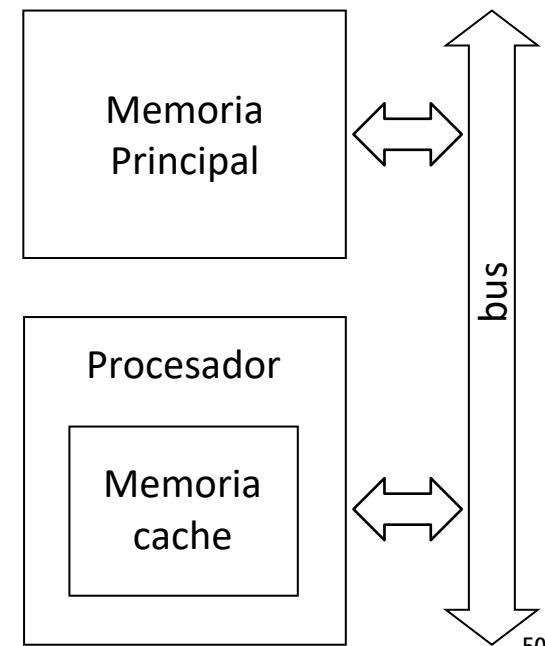
- Problema: ¿Cuál programa? Acordar benchmarks
- Depende de diseño CPU, repertorio instrucciones...
 - y del **compilador!!!** (benchmarks en lenguaje alto nivel)
 - y versión **SO, librerías**, etc.

■ Ejemplo anterior: **t5-t0 incluye HD, LPR**

- Tiempo transcurrido (wall-clock time)
- Mide rendimiento sistema completo
 - Influido por prestaciones CPU, HD, LPR, etc

■ **Benchmarks CPU ejercitan sólo CPU**

- Tiempo de procesamiento (CPU time)
- Influido por prestaciones **CPU, M, caches, buses**
 - cache conserva lo accedido recientemente/más rápida
 - ventaja en ejecución bucles, p.ej.



Rendimiento

■ Reloj del procesador

- UC emplea **varios ciclos** de reloj en ejecutar una instrucción
- pasos básicos 1 ciclo (comutar señales control)
- Frecuencia $R = 1/P$
 - $500\text{MHz} = 1 / 2\text{ns}$
 - $1.25\text{GHz} = 1 / 0.8\text{ns}$

■ Ecuación básica de rendimiento

- T tiempo para ejecutar programa benchmark
- **N instrucciones** (recuento dinámico bucles/subrutinas)
 - N no necesariamente igual a #instr. progr. objeto.
- **S ciclos/instr.** (“pasos básicos” de media)

$$T = \frac{N \times S}{R} \quad \begin{matrix} \text{ciclos} \\ \text{ciclos/s} \end{matrix}$$

Rendimiento

■ Ecuación básica de rendimiento

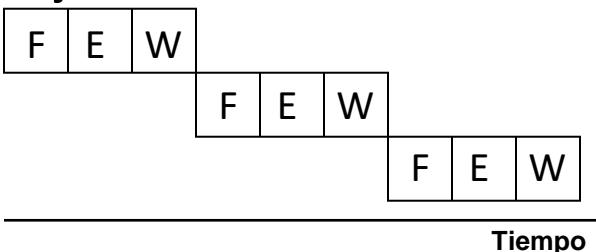
$$T = \frac{N \times S}{R} \quad \begin{matrix} \text{ciclos} \\ \text{ciclos/s} \end{matrix}$$

- Ideal: N y S ↓↓, R ↑↑
 - N (instrucciones) depende de compilador/repertorio
 - S (ciclos/instr) depende de implementación CPU
 - R (MHz - GHz) depende de tecnología (y diseño CPU)
- alterar uno modifica los otros
 - aumentar R puede ser a costa de aumentar S
 - lo importante es que al final T↓

Segmentación de cauce (intenta que $S \approx 1$)

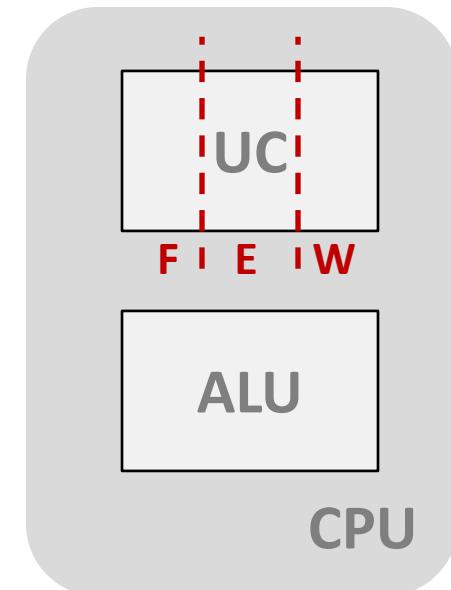
- NxS es suponiendo ejecución individual instrucciones

ADD R1,R2, R3



MUL R4,R5, R5

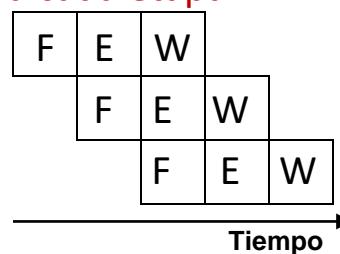
SUB R3,R5, R5



- Pero las distintas etapas hacen tareas distintas

- UC puede tener **circuitería separada para cada etapa:**

- Fetch: captación
- Exec: ejecución
- Write: actualización registro



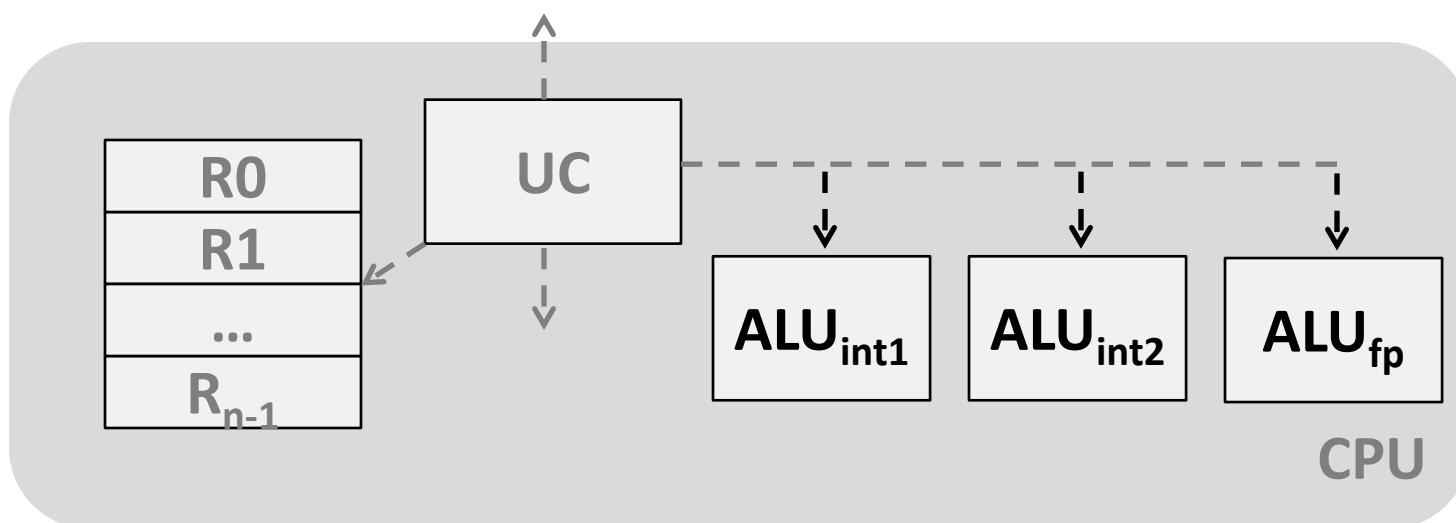
- una vez lleno el cauce, valor efectivo $S=1$ ciclo/instr**

- dependencias datos
- competición recursos
- saltos
- $S \geq 1, S \approx 1$

- (ej: MUL-SUB arriba, dependencia R5)
- (ej: almacenar resultado M/fetch instrucción+2)
(pipeline flush)

Funcionamiento superescalar (que S<1)

- Conseguir paralelismo a base de **reduplicar UFs** (unidades funcionales)
 - ej: 2 ALU enteros, 2 ALU FP
 - emitir hasta 4 operaciones simultáneas (2int+2fp)
 - si orden apropiado instrucciones (en secuencia programa)
 - combinado con segmentación, puede hacer S<1
 - se completa más de 1 instrucción por ciclo
- común en CPUs actuales. Dificultades:
 - **emisión desordenada**
 - **corrección** (mismo resultado que ejecución escalar)



Otras formas de reducir T

■ Velocidad del reloj (R↑, S/R)

- Tecnología $\uparrow \Rightarrow R\uparrow$
 - Si no cambia nada más, Rx2 $\Rightarrow T/2?$ ($T = NS / R$)
 - Falso: Memoria también Rx2 !!! o mejorar cache L1-L2
- Alternativamente, S $\uparrow \Rightarrow R\uparrow$
 - “supersegmentación”, reducir tarea por ciclo reloj
 - difícil predecir ganancia, puede incluso empeorar

■ Repertorio RISC/CISC (N·S)

- RISC: instr. simples para $R\uparrow\uparrow$, pero $S\downarrow \Rightarrow N\uparrow$
- CISC: instr. complejas para $N\downarrow\downarrow$, pero $S\uparrow$
 - corregir $S\uparrow$ con segmentación \Rightarrow competición recursos
- actualmente técnicas híbridas RISC/CISC

Otras formas de reducir T

■ Compilador $(N \downarrow)$

- optimizador espacial ($N \downarrow$) o temporal ($N \times S \downarrow$)
 - usualmente contrapuestos
- espacial: requiere conocimiento **arquitectura**
 - repertorio, modos direccionamiento, alternativas traducción...
- temporal: requiere conocimiento detallado **organización**
 - reordenación instrucciones para ahorrar ciclos
 - evitar competición recursos
- optimización debe ser **correcta** (mismo resultado)

Medida del rendimiento

■ Interesante para:

- diseñadores CPUs: evaluar mejoras introducidas
- fabricantes: marketing
- **compradores:** prestaciones/precio

■ Benchmark: 1 único programa acordado ?!?

- programas **sintéticos** no predicen bien T_{app}
- programas **reales** muy específicos
- colección programas considerados “**frecuentes**” (representativos)
- reducir a un único número usando **media geométrica**
 - evitar influencia computador referencia

Medida del rendimiento

■ SPEC: System Performance Evaluation Corporation

- tests: CPU92, CPU95, CPU2000, CPU2006 (CPUv6)
- CPU2006:
 - Referencia: WS UltraSPARC II 300MHz
 - CINT2006: gzip, bzip2, gcc, chess, gene seq., Perl... (12)
 - CFP2006: CFD, chromodynamics, ray-tracing, meteo... (17)

$$\text{vel}_{\text{gzip}} = T_{\text{ref}}_{\text{gzip}} / T_{\text{gzip}} \quad (\text{vel}=50 \Rightarrow 50x \text{ uSPARC II})$$

$$\text{vel}_{\text{SPEC}} = \sqrt[n]{\prod_i \text{vel}_{\text{prgi}}} \quad (n=12/17, \text{ media geom.})$$

■ mide efecto combinado

- CPU, M, SO, compilador
- <http://www.spec.org> (no es gratuito)

Introducción

- Unidades funcionales
- Conceptos básicos de funcionamiento
- Estructuras de bus
- Rendimiento
- Perspectiva histórica

Perspectiva histórica

■ 2ª Guerra Mundial

- Tecnología **relés** electromagnéticos $T_{\text{conmut.}} = O(s)$
 - Previamente: engranajes, palancas, poleas
- Tablas logaritmos, aprox. func. trigonométricas
- Generaciones 1-2-3-4ª 1945-55-65-75-etc

■ 1ª Generación (45-55): tubos de vacío

- von Neumann: concepto **prog. almacenado**
- tubos vacío 100-1000x $T_{\text{conmut.}} = O(ms)$
- M: líneas retardo mercurio, **núcleos magn.**
- E/S: lect/perf. tarjetas, cintas magnéticas
- software: lenguaje máquina / ensamblador
 - 1946-47 **ENIAC** UNIVAC
 - 1952-57 EDVAC UNIVAC II
 - 1953-55 IBM 701 702



Perspectiva histórica

■ 2ª Generación (55-65): transistores

- invento Bell AT&T 1947 $T_{\text{conmut.}} = O(\mu\text{s})$
- E/S: procesadores E/S (cintas) en paralelo con CPU
- software: compilador FORTRAN
 - 1955-57 IBM704 DEC PDP-1
 - 1964 IBM 7094



■ 3ª Generación (1965-75): Circuito Integrado

- velocidad CPU/M ↑ $T_{\text{conmut.}} = O(\text{ns})$
- arquitectura: μProgr, segm. cauce, M cache
- software: SO multiusuario, memoria virtual
 - 1965 IBM S/360
 - 1971-77 DEC PDP-8



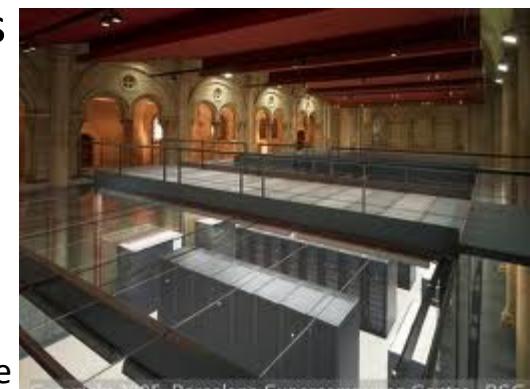
Perspectiva histórica

■ 4^a Generación (75-...): VLSI

- μProcesador: procesador completo en 1 chip
 - MP completa en uno o pocos chips
 - Intel, Motorola, AMD, TI, NS
- arquitectura: mejoras segm. cauce, cache, M virtual
- hardware: **portátiles**, PCs, WS, **redes**
- mainframes siguen sólo en grandes empresas
 - 1972-74-78 i8008 i8080 i8086
 - 1982-85-89 i80286 i80386 i80486

■ Actualidad

- Computadores sobremesa potentes/asequibles
- Internet
- Paralelismo masivo (Top500, **MareNostrum**, **Magerit**)
 - 1995-97-99-01 Pentium PII PIII P4
 - 2004-06-08 Pentium 4F, Core 2 Duo, Core i7
 - 2011-15-20 Core i7 2nd-6thgen, Kaby/Coffee/Cannon/Ice Lake



Introducción

■ Unidades funcionales

- E/S, M, CPU (ALU+UC)
- Memoria de bytes, alineamiento, ordenamiento
- Clasificación arq. m/n, pila, acumulador, RPG (R/R, R/M, M/M)
- Repertorios RISC/CISC, modos de direccionamiento

■ Conceptos básicos de funcionamiento

- Ciclo de ejecución

■ Estructuras de bus

- Bus único, buses múltiples, decodificación parcial/completa

■ Rendimiento

- Software de sistema, ecuación básica rendimiento, benchmarks
- Segmentación, funcionamiento superescalar, SPEC

■ Perspectiva histórica

- generaciones

Guía de trabajo autónomo (4h/s)

■ Estudio

- Cap.1 Hamacher (incluye problemas)

■ Lectura

- Guión de la Práctica 2
- Cap.3 CS:APP (Bryant/O'Hallaron)

■ Para los entusiastas: Ubuntu en el portátil (Ubuntu LTS 18.04 en ETSIIT)

- Posibilidades de usar Ubuntu en portátil:
 - Instalación directa (además de, o en lugar de, Windows)
 - VirtualBox + Ubuntu LTS (es +complicado, pero +ventajoso)
 - » no requiere rebotar, no toca MBR, se puede usar Windows a la vez
 - instalar paquetes **g++/make/ghex** (usar apt o Synaptic), y **default-jre** (para eclipse)
 - instalar snap **eclipse 4.8** (usar snap o UbuntuSoftware) (evitar paquete **eclipse 3.8**)
 - comprobar firewall con “`sudo ufw [status | enable]`”
- Instálarselo e intentar Ejercicios 1-4 del guión P2

Programación a Nivel-Máquina I: Conceptos Básicos y Aritmética

Estructura de Computadores
Semana 3

Bibliografía:

[BRY16] Cap.3 Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016
 Signatura ESIIT/[C.1 BRY com](#)

Transparencias del libro CS:APP, Cap.3

Introduction to Computer Systems: a Programmer's Perspective

Autores: Randal E. Bryant y David R. O'Hallaron

<http://www.cs.cmu.edu/afs/cs/academic/class/15213-f15/www/schedule.html>

Guía de trabajo autónomo (4h/s)

■ Lectura: del Cap.3 CS:APP (Bryant/O'Hallaron)

- Historical perspective, Program Encodings
 - § 3.1 – 3.2 pp.199-213
- Data Formats, Accessing Info.
 - § 3.3 – 3.4 pp.213-227
- Arithmetic and Logical Operations
 - § 3.5 pp.227-236

■ Ejercicios: del Cap.3 CS:APP (Bryant/O'Hallaron)

- Probl. 3.1 – 3.5 § 3.4, pp.218, 221, 222, 223, 225
- Probl. 3.6 – 3.12 § 3.5, pp.228, 229, 230, 231, 232, 233, 236

Bibliografía:

[BRY16] Cap.3

Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016

Signatura ESIIT/[C.1 BRY com](#)

Programación Máquina I: Conceptos Básicos

- Historia de los procesadores y arquitecturas de Intel
- Lenguaje C, ensamblador, código máquina
- Conceptos básicos asm: Registros, operandos, move
- Operaciones aritméticas y lógicas

Procesadores Intel x86

- **Dominan el mercado portátil/sobremesa/servidor**
- **Diseño evolutivo**
 - Compatible ascendentemente hasta el 8086, introducido en 1978
 - Va añadiendo características conforme pasa el tiempo
- **Computador con repertorio instrucciones complejo (CISC)**
 - Muchas instrucciones diferentes, con muchos formatos distintos
 - Pero sólo un pequeño subconjunto aparece en programas Linux
 - Difícil igualar prestaciones Computadores Repertorio Instr. Reducido (RISC)
 - Sin embargo, ¡Intel ha conseguido justo eso!
 - En lo que a velocidad se refiere. No tanto en (bajo) consumo.

Evolución Intel x86: Hitos significativos

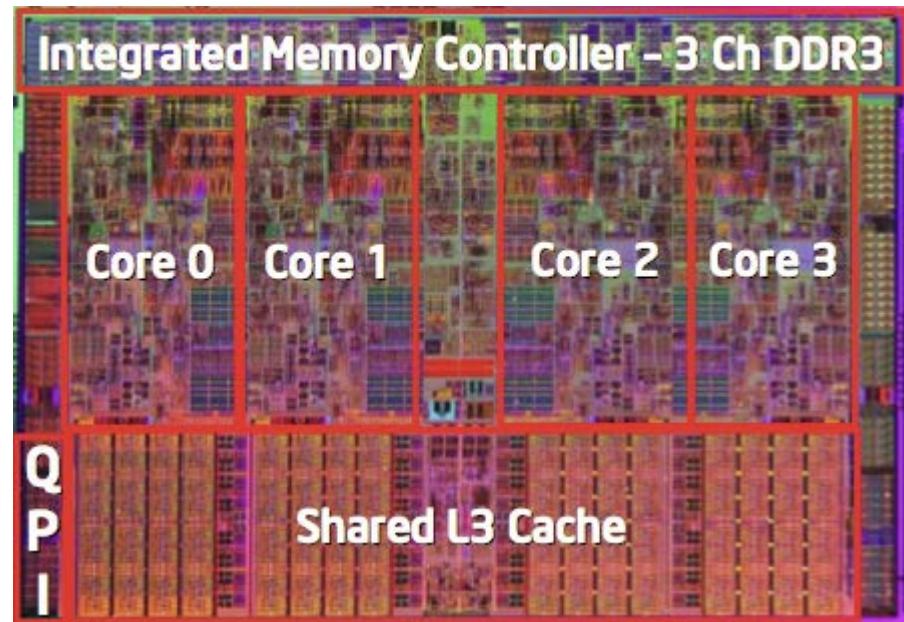
| <i>Nombre</i> | <i>Fecha</i> | <i>Transistores</i> | <i>MHz</i> |
|---------------------|--------------|---------------------|---|
| ■ 8086 | 1978 | 29K | 5-10 |
| | | | <ul style="list-style-type: none">■ Primer procesador Intel 16-bit. Base para el IBM PC & MS-DOS■ Espacio direccionamiento 1MB |
| ■ 386 | 1985 | 275K | 16-33 |
| | | | <ul style="list-style-type: none">■ Primer procesador Intel 32-bit de la familia (x86 luego llamada) IA32■ Añadió “direcciónamiento plano”[†], capaz de arrancar Unix |
| ■ Pentium 4E | 2004 | 125M | 2800-3800 |
| | | | <ul style="list-style-type: none">■ 1^{er} proc. Intel 64-bit de la familia (x86, llamada x86-64, EM64t) Intel 64 |
| ■ Core 2 | 2006 | 291M | 1060-3500 |
| | | | <ul style="list-style-type: none">■ Primer procesador Intel multi-core |
| ■ Core i7 | 2008 | 731M | 1700-3900 |
| | | | <ul style="list-style-type: none">■ Cuatro cores, hyperthreading (2 vías) |

[†] “flat addressing” 5

Procesadores Intel x86: Visión general

■ Evolución de las máquinas

| | | |
|-------------------|------|------|
| ■ 386 | 1985 | 0.3M |
| ■ Pentium | 1993 | 3.1M |
| ■ Pentium/MMX | 1997 | 4.5M |
| ■ PentiumPro | 1995 | 6.5M |
| ■ Pentium III | 1999 | 8.2M |
| ■ Pentium 4 | 2001 | 42M |
| ■ Core 2 Duo | 2006 | 291M |
| ■ Core i7 | 2008 | 731M |
| ■ Core i7 Skylake | 2015 | 1.9B |



Microfotografía de un dado Core i7

■ Características añadidas

- Instrucciones de soporte para operación multimedia (ops. en paralelo)
- Instrucciones para posibilitar operaciones condicionales más eficientes
- Transición de 32 bits a 64 bits
- Más núcleos (cores)

Procesadores Intel x86

■ Generaciones pasadas Tecnología del proceso

- 1st Pentium Pro 1995 600 nm
- 1st Pentium III 1999 250 nm
- 1st Pentium 4 2000 180 nm
- 1st Core 2 Duo 2006 65 nm

Tamaño tecnología proceso
= anchura mínima trazo
(10 nm ≈ 100 átomos ancho)

■ Generaciones recientes y venideras

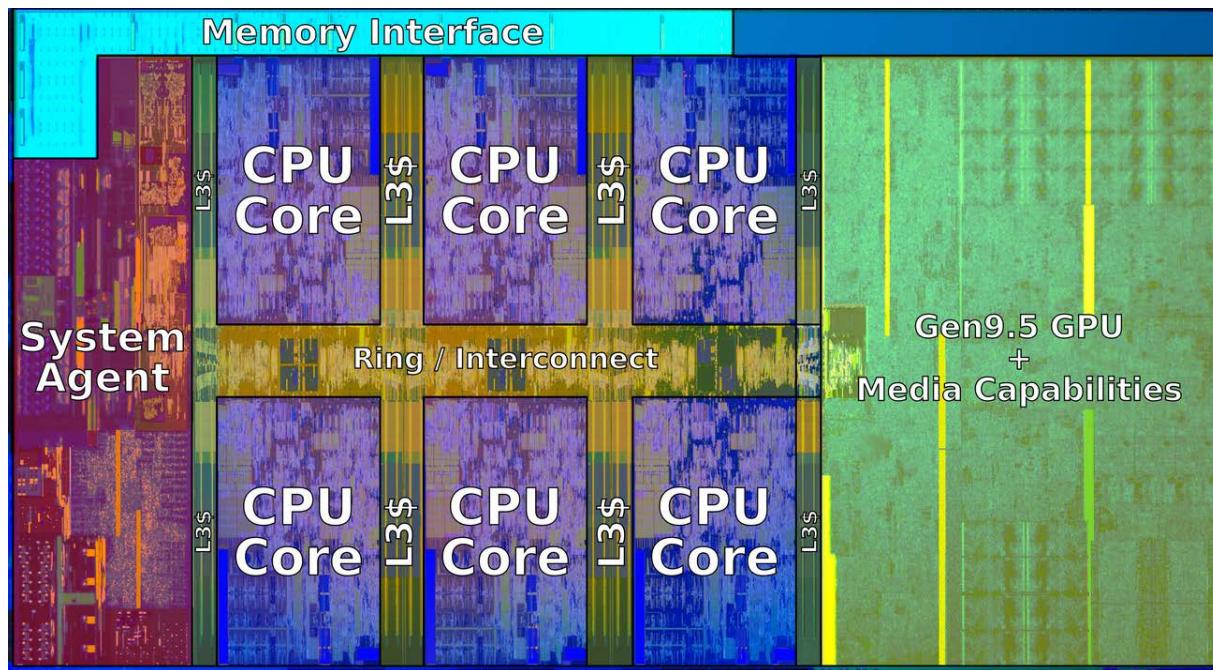
1. Nehalem 2008 45 nm
2. Sandy Bridge 2011 32 nm
3. Ivy Bridge 2012 22 nm
4. Haswell 2013 22 nm
5. Broadwell 2014 14 nm
6. Skylake 2015 14 nm
7. Kaby Lake 2016 14 nm
8. Coffee Lake 2017 14 nm
- Cannon Lake 2019? 10 nm

Lo último[†] en 2018

- Core i7 CoffeeLake 2018 (SkyLake 6xxx, KabyLake 7xxx, CoffeeLake 8xxx)

■ Modelo móvil: Core i7

- 2.2-3.2 GHz
- 45 W



■ Sobremesa: Core i7

- Gráficos integrados
- 2.4-4.0 GHz
- 35-95 W

■ Modelo servidor: Xeon E

- Gráficos integrados
- Habilitado para multi-zócalo[#]
- 3.3-3.8 GHz
- 80-95 W

[†] "state of the art"

[#] "multi-socket" 8

Clones x86: Advanced Micro Devices (AMD)

■ Históricamente

- AMD ha ido siguiendo a Intel en todo
- CPUs un poco más lentas, mucho más baratas

■ Y entonces

- Reclutaron los mejores diseñadores de circuitos de Digital Equipment Corp. y otras compañías con tendencia descendente
- Construyeron el Opteron: duro competidor para el Pentium 4
- Desarrollaron x86-64, su propia extensión a 64 bits

■ En años recientes

- Intel ha empezado a organizarse para ser más efectiva
 - Lidera el mundo de tecnologías de semiconductores
- AMD se ha quedado rezagada
 - Recurre a fabricante de semiconductores externalizado

La historia de los 64-bit de Intel

- **2001: Intel intenta un cambio radical de IA32 a IA64**
 - Arquitectura totalmente diferente (Itanium)
 - Ejecuta código IA32 sólo como herencia[†]
 - Prestaciones decepcionantes
- **2003: AMD interviene con una solución evolutiva**
 - x86-64 (ahora llamado “AMD64”)
- **Intel se sintió obligada a concentrarse en IA64**
 - Difícil admitir error, o admitir que AMD es mejor
- **2004: Intel anuncia extensión EM64T[‡] de la IA32** (ahora llamada Intel64)
 - Extended Memory 64-bit Technology
 - ¡Casi idéntica a x86-64!
- **Todos los procesadores x86 salvo gama baja soportan x86-64**
 - Pero gran cantidad de código se ejecuta aún en modo 32-bits

[†] “legacy” = herencia de características

[‡] Intel usa ahora “IA32” e “Intel64” para

distinguir IA32 de EM64T y evitar confusión con IA64 10

Nosotros cubrimos:

■ IA32

- El ~~x86 tradicional~~
- Para EC: ~~RIP, verano 2018~~

■ x86-64 / Intel64

- El **estándar**
- ubuntu_18> gcc hello.c
- ubuntu_18> gcc -m64 hello.c

■ Presentación

- El libro cubre x86-64. Transparencias, prácticas, ejercicios... todo en x86-64.
- En el libro hay un “añadido Web”[†] sobre IA32
- En SWAD puede quedar material (tests/exámenes/...) sobre IA32
- Sólo algunos **detalles** querremos recordar de IA32 (alineamiento, pila...)

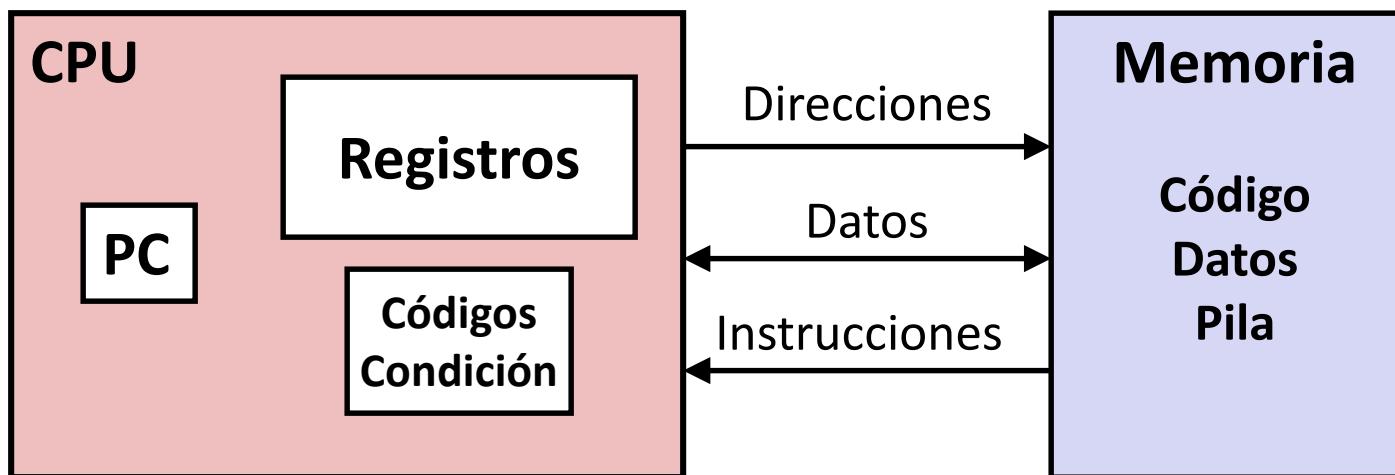
Programación Máquina I: Conceptos Básicos

- Historia de los procesadores y arquitecturas de Intel
- Lenguaje C, ensamblador, código máquina
- Conceptos básicos asm: Registros, operandos, move
- Operaciones aritméticas y lógicas

Definiciones

- **Arquitectura:** (también arquitectura del repertorio de instrucciones: ISA) Las partes del diseño de un procesador que se necesitan entender para escribir **código ensamblador**.
 - Ejemplos: especificación del repertorio de instrucciones, registros.
- **Formas del código:**
 - **Código máquina:** Programas (codops, bytes) que ejecuta el procesador
 - **Código ensamblador:** Representación textual del código máquina
- **Microarquitectura: Implementación de la arquitectura.**
 - Ejemplos: tamaño de las caches y frecuencia de los cores.
- **Ejemplos de ISAs:**
 - Intel: (x86 =) IA32, Itanium (= IA64 = IPF), x86-64 (= Intel 64 = EM64t)
 - ARM: Usado en casi todos los teléfonos móviles
 - RISC V: nueva ISA open-source

Perspectiva Código Ensamblador/Máquina

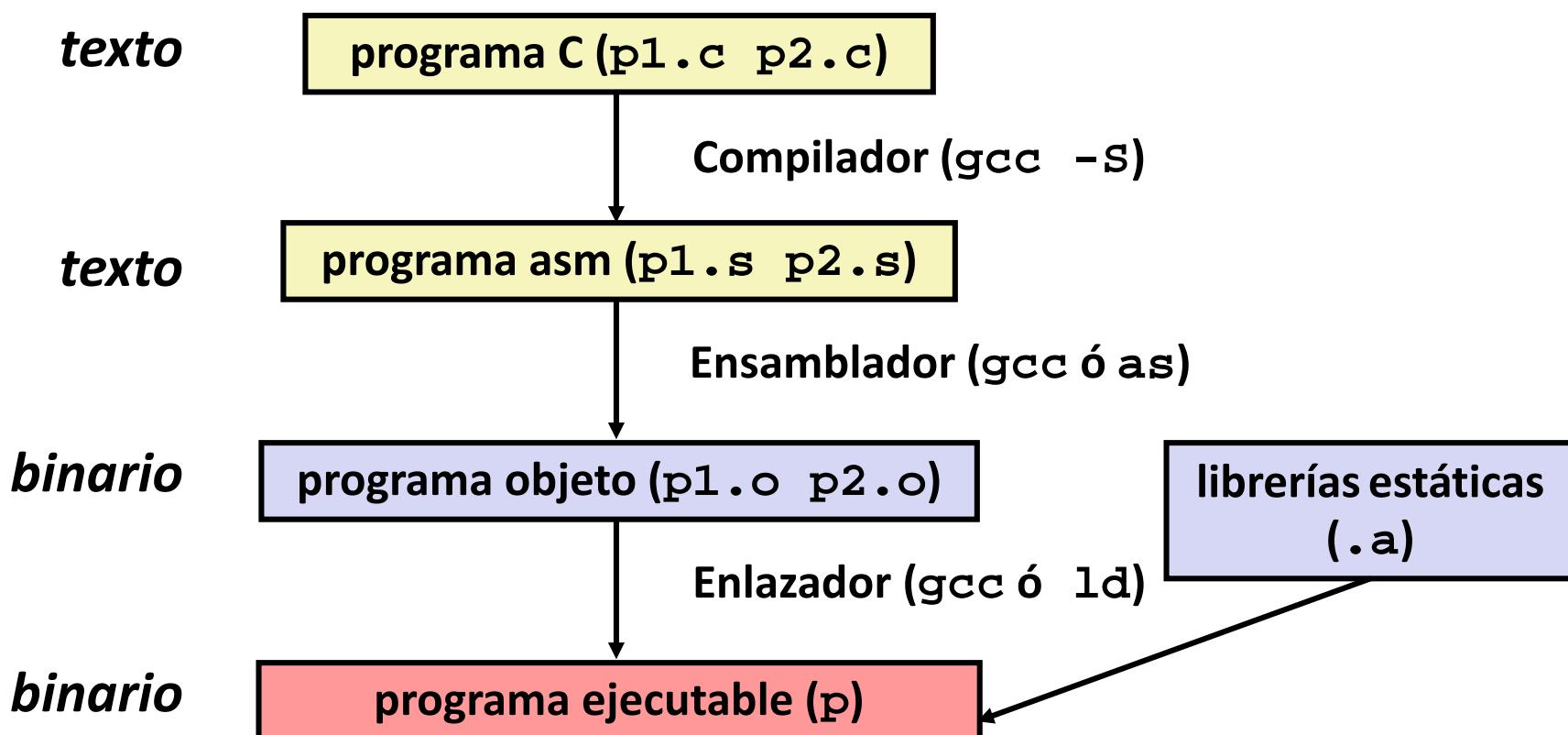


Estado visible al programador

- **PC: Contador de programa**
 - Dirección de la próxima instrucción
 - Llamado “RIP” (x86-64)
- **Archivo de registros**
 - Datos del programa muy utilizados
- **Códigos de condición / flags de estado**
 - Almacenan información estado sobre la operación aritmética/lógica más reciente
 - Usados para bifurcación condicional
- **Memoria**
 - Array direccionable por bytes
 - Código y datos usuario
 - Pila soporte a procedimientos

Convertir C en Código Objeto

- Código en ficheros `p1.c p2.c`
- Compilar con el comando: `gcc -Og p1.c p2.c -o p`
 - Usar optimizaciones básicas (`-Og`) [versiones recientes de GCC[†]]
 - Poner binario resultante en fichero `p`



Compilar a ensamblador

Código C (sum.c)

```
long plus(long x, long y);

void sumstore(long x, long y,
              long *dest)
{
    long t = plus(x, y);
    *dest = t;
}
```

Ensamblador x86-64 generado[†]

sumstore:

| | |
|-------|--------------|
| pushq | %rbx |
| movq | %rdx, %rbx |
| call | plus |
| movq | %rax, (%rbx) |
| popq | %rbx |
| ret | |

Obtenerlo con el comando

```
gcc -Og -S sum.c
```

Produce el fichero sum.s

Aviso: Se obtendrán resultados diferentes en cuanto se usen diferentes versiones de gcc y diferentes ajustes del compilador[†]

[†] Añadir *-fno-asynchronous-unwind-tables*
en GCC 7.3 Ubuntu 18.04 16

Representación Datos C, IA32, x86-64

■ Tamaño de Objetos C (en Bytes)

| <i>Tipo de Datos C</i> | <i>Normal 32-bit</i> | <i>Intel IA32</i> | <i>x86-64</i> |
|------------------------|----------------------|-------------------|---------------|
| ▪ unsigned | 4 | 4 | 4 |
| ▪ int | 4 | 4 | 4 |
| ▪ long int | 4 | 4 | 8 |
| ▪ char | 1 | 1 | 1 |
| ▪ short | 2 | 2 | 2 |
| ▪ float | 4 | 4 | 4 |
| ▪ double | 8 | 8 | 8 |
| ▪ long double | 8 | 10/12 | 16 |
| ▪ char * | 4 | 4 | 8 |

– o cualquier otro puntero

Características Ensamblador: Tipos de Datos

- **Datos “enteros” de 1, 2, 4 u 8 bytes**
 - Valores de datos
 - Direcciones (punteros sin tipo)
- **Datos en punto flotante de 4, 8 ó 10 bytes**
- **Código: secuencias de bytes codificando serie de instrucciones**
- **No hay tipos compuestos como arrays o estructuras**
 - Tan sólo bytes ubicados contiguamente (uno tras otro) en memoria

Características Ensamblador: Instrucciones

■ Realizan función aritmética sobre datos en registros o memoria

- “Operaciones” = Instrucciones aritmético/lógicas

■ Transfieren datos entre memoria y registros

- Cargar datos de memoria a un registro
- Almacenar datos de un registro en memoria
- “Instrucciones de transferencia”

■ Transferencia de control

- Incondicionales: saltos, llamadas a procedimientos, retornos desde procs.
- Saltos condicionales
- “Instrucciones de control”

Código Objeto

Código de sumstore

0x0400595:

0x53

0x48

0x89

0xd3

0xe8

0xf2

0xff

0xff

0xff

- 14 bytes total

- Cada instrucción

- 1, 3, ó 5 bytes

- Empieza en direcc.

0xc3 0x0400595

■ Ensamblador

- Traduce .s pasándolo a .o
- Instrucciones codificadas en binario
- Imagen casi completa del código ejecutable
- Le faltan enlaces entre código de ficheros diferentes

■ Enlazador

- Resuelve referencias entre ficheros
- Combina con libs. de tiempo ejec. estáticas[†]
 - P.ej., código para **malloc**, **printf**
- Algunas libs. son *dinámicamente enlazadas*[#]
 - El enlace ocurre cuando el programa empieza a ejecutarse

[†] “static run-time libraries” = bibliotecas estáticas para soporte en tiempo de ejecución
[#] “dynamically linked libraries”, o también “shared libs”

Ejemplo de Instrucción Máquina

```
*dest = t;
```

■ Código C

- Almacenar valor **t** adonde indica (apunta) **dest**

```
movq %rax, (%rbx)
```

■ Ensamblador

- Mover un valor de 8-byte a memoria
 - “Palabra Quad”[†] en jerga x86-64
- Operандos:

t: Registro **%rax**

dest: Registro **%rbx**

***dest:** Memoria **M[%rbx]**

```
0x40059e: 48 89 03
```

■ Código Objeto

- Instrucción de 3-byte
- Almacenada en dir. **0x40059e**

Desensamblar Código Objeto

Desensamblado

```
0000000000400595 <sumstore>:  
 400595: 53          push    %rbx  
 400596: 48 89 d3    mov      %rdx,%rbx  
 400599: e8 f2 ff ff ff    callq   400590 <plus>  
 40059e: 48 89 03    mov      %rax,(%rbx)  
 4005a1: 5b          pop     %rbx  
 4005a2: c3          retq
```

■ Desensamblador

`objdump -d sum`

- Herramienta útil para examinar código objeto
- Analiza el patrón de bits de series de instrucciones
- Produce versión aproximada del código ensamblador (correspondiente)
- Puede ejecutarse sobre el fich. a.out (ejecutable completo) ó el .o

Desensamblado Alternativo

Objeto

```
0x0400595:  
 0x53  
 0x48  
 0x89  
 0xd3  
 0xe8  
 0xf2  
 0xff  
 0xff  
 0xff  
 0x48  
 0x89  
 0x03  
 0x5b  
 0xc3
```

Desensamblado

```
Dump of assembler code for function sumstore:  
 0x0000000000400595 <+0>: push    %rbx  
 0x0000000000400596 <+1>: mov     %rdx,%rbx  
 0x0000000000400599 <+4>: callq   0x400590 <plus>  
 0x000000000040059e <+9>: mov     %rax,(%rbx)  
 0x00000000004005a1 <+12>:pop    %rbx  
 0x00000000004005a2 <+13>:retq
```

■ Desde el Depurador gdb

```
gdb sum
```

```
disassemble sumstore
```

- Desensamblar procedimiento

```
x/14xb sumstore
```

- Examinar 14 bytes a partir de sumstore



¿Qué se puede Desensamblar?

```
% objdump -d WINWORD.EXE  
  
WINWORD.EXE:      file format pei-i386  
  
No symbols in "WINWORD.EXE".  
Disassembly of section .text:  
  
30001000 <.text>:  
30001000:  
30001001:  
30001003:  
30001005:  
3000100a:  Ingeniería inversa prohibida por licencia  
            Microsoft End User License Agreement  
            (EULA)
```

- Cualquier cosa que se pueda interpretar como código ejecutable
- El desensamblador examina bytes y reconstruye el fuente asm.

Programación Máquina I: Conceptos Básicos

- Historia de los procesadores y arquitecturas de Intel
- Lenguaje C, ensamblador, código máquina
- **Conceptos básicos asm: Registros, operandos, move**
- Operaciones aritméticas y lógicas

Registros enteros x86-64

| | | | |
|-------------|-------------|------------|------------|
| %rax | %eax | %ax | %al |
|-------------|-------------|------------|------------|

| | | | |
|------------|-------------|-------------|-------------|
| %r8 | %r8d | %r8w | %r8b |
|------------|-------------|-------------|-------------|

| | |
|-------------|-------------|
| %rbx | %ebx |
|-------------|-------------|

| | |
|------------|-------------|
| %r9 | %r9d |
|------------|-------------|

| | |
|-------------|-------------|
| %rcx | %ecx |
|-------------|-------------|

| | |
|-------------|--------------|
| %r10 | %r10d |
|-------------|--------------|

| | |
|-------------|-------------|
| %rdx | %edx |
|-------------|-------------|

| | |
|-------------|--------------|
| %r11 | %r11d |
|-------------|--------------|

| | | | |
|-------------|-------------|------------|-------------|
| %rsi | %esi | %si | %sil |
|-------------|-------------|------------|-------------|

| | |
|-------------|--------------|
| %r12 | %r12d |
|-------------|--------------|

| | |
|-------------|-------------|
| %rdi | %edi |
|-------------|-------------|

| | |
|-------------|--------------|
| %r13 | %r13d |
|-------------|--------------|

| | |
|-------------|-------------|
| %rsp | %esp |
|-------------|-------------|

| | |
|-------------|--------------|
| %r14 | %r14d |
|-------------|--------------|

| | |
|-------------|-------------|
| %rbp | %ebp |
|-------------|-------------|

| | |
|-------------|--------------|
| %r15 | %r15d |
|-------------|--------------|

- Pueden referenciarse los 4 bytes de menor peso[†] (los 4 LSBs)
 - (también los 2 LSB y el 1 LSB)

[†] “low-order 4 bytes” 26

Un poco de historia: registros IA32

propósito general

| | | | |
|------|-----|-----|-----|
| %eax | %ax | %ah | %al |
| %ecx | %cx | %ch | %cl |
| %edx | %dx | %dh | %dl |
| %ebx | %bx | %bh | %bl |
| %esi | %si | | |
| %edi | %di | | |
| %esp | %sp | | |
| %ebp | %bp | | |

Motivos nombre
(mayoría obsoletos)

acumulador

contador

datos

base

*índice
fuente*

*índice
destino*

*puntero
de pila*

*puntero
base*

registros virtuales 16-bit
(compatibilidad ascendente)

Mover Datos

■ Mover Datos

`movq Source, Dest†`

■ Tipo de Operandos

- **Inmediato:** Datos enteros constantes
 - Ejemplo: `$0x400, $-533`
 - Como constante C, pero con prefijo '`$`'
 - Codificado mediante 1, 2, ó 4 bytes[‡]
- **Registro:** Alguno de los 16 registros enteros
 - Ejemplo: `%rax, %r13`
 - Pero `%rsp` reservado para uso especial
 - Otros tienen usos especiales con instrucciones particulares
- **Memoria:** 8 bytes consecutivos mem. en dirección dada por un registro
 - Ejemplo más sencillo: (`%rax`)
 - Hay otros diversos “modos de direccionamiento”

`%rax`

`%rcx`

`%rdx`

`%rbx`

`%rsi`

`%rdi`

`%rsp`

`%rbp`

`%rN`

[†] luego veremos `movabsq` para literales 8B

[‡] “source/destination” = fuente/destino 28

Combinaciones de Operandos movq

| | Source | Dest | Src,Dest | Análogo C |
|------|-------------------------|------------------|--------------------|----------------|
| movq | <i>Imm</i> ^t | <i>Reg</i> | movq \$0x4,%rax | temp = 0x4; |
| | | <i>Mem</i> | movq \$-147,(%rax) | *p = -147; |
| | <i>Reg</i> | <i>Reg</i> | movq %rax,%rdx | temp2 = temp1; |
| | <i>Mem</i> | movq %rax,(%rdx) | *p = temp; | |
| | <i>Mem</i> | <i>Reg</i> | movq (%rax),%rdx | temp = *p; |

Ver resto instrucciones transferencia (incluyendo pila) en el libro

No se puede transferir Mem-Mem con sólo una instrucción

Modos Direcccionamiento a memoria sencillos

■ Normal[†]

(R)

Mem[Reg[R]]

- El registro R indica la dirección de memoria
- ¡Exacto! Como seguir (*desreferenciar[#]*) un puntero en C

movq (%rcx), %rax

■ Desplazamiento D(R)

Mem[Reg[R]+D]

- El registro R indica el inicio de una región de memoria
- La constante de desplazamiento D indica el *offset[#]*

movq 8(%rbp), %rdx

[#] “offset”=compensación, para nosotros “desplazamiento”

[†] “indirecto a través de registro” según otros autores

[#] “dereferencing” en el original

Ejemplo Modos Direcccionamiento sencillos

```
void adiv(<tipo> a, <tipo> b)
{
    ????
    ?? = ???;
    ????
    ?? = ???;
    ???
    ?? = ??;
    ???
    ?? = ??;
}
```

%rdi

%rsi

adiv:

| | |
|------|--------------|
| movq | (%rdi), %rax |
| movq | (%rsi), %rdx |
| movq | %rdx, (%rdi) |
| movq | %rax, (%rsi) |
| ret | |

Ejemplo Modos Direcccionamiento sencillos

```
void swap(long *xp, long *yp)
{
    long t0 = *xp;
    long t1 = *yp;
    *xp = t1;
    *yp = t0;
}
```

swap:

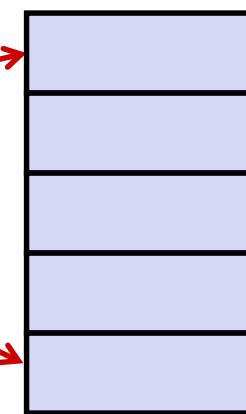
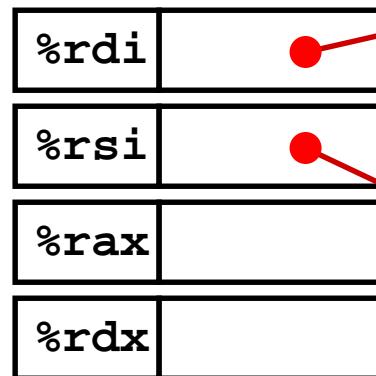
| | |
|------|--------------|
| movq | (%rdi), %rax |
| movq | (%rsi), %rdx |
| movq | %rdx, (%rdi) |
| movq | %rax, (%rsi) |
| ret | |

Comprendiendo swap()

```
void swap(long *xp, long *yp)
{
    long t0 = *xp;
    long t1 = *yp;
    *xp = t1;
    *yp = t0;
}
```

Memoria

Registros



| Registro | Valor |
|----------|-------|
| %rdi | xp |
| %rsi | yp |
| %rax | t0 |
| %rdx | t1 |

swap:

```
        movq    (%rdi), %rax    # t0 = *xp
        movq    (%rsi), %rdx    # t1 = *yp
        movq    %rdx, (%rdi)    # *xp = t1
        movq    %rax, (%rsi)    # *yp = t0
        ret
```

Comprendiendo swap()

Registers

| | |
|------|-------|
| %rdi | 0x120 |
| %rsi | 0x100 |
| %rax | |
| %rdx | |

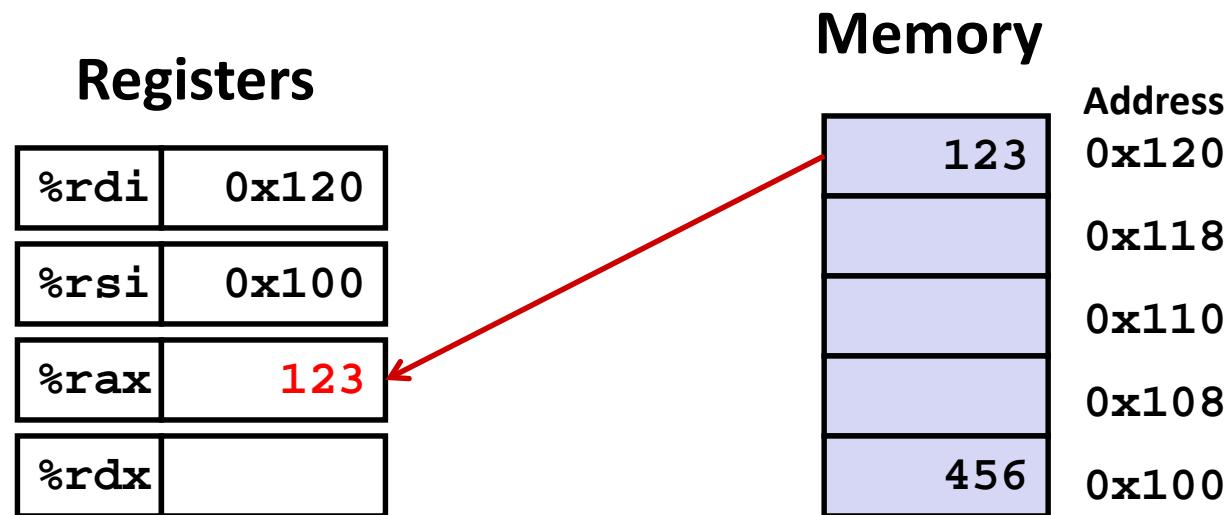
Memory

| | |
|-----|------------------|
| 123 | Address 0x120 |
| | 0x118 |
| | 0x110 |
| | 0x108 |
| 456 | 0x100 |

swap:

```
    movq    (%rdi), %rax    # t0 = *xp
    movq    (%rsi), %rdx    # t1 = *yp
    movq    %rdx, (%rdi)    # *xp = t1
    movq    %rax, (%rsi)    # *yp = t0
    ret
```

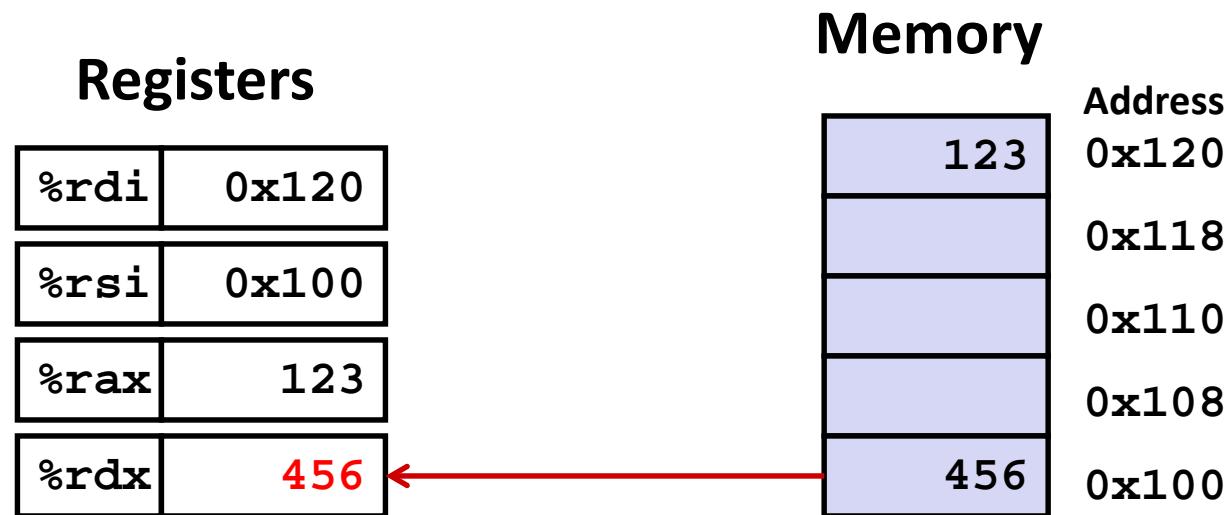
Comprendiendo swap()



swap:

```
    movq    (%rdi), %rax    # t0 = *xp
    movq    (%rsi), %rdx    # t1 = *yp
    movq    %rdx, (%rdi)    # *xp = t1
    movq    %rax, (%rsi)    # *yp = t0
    ret
```

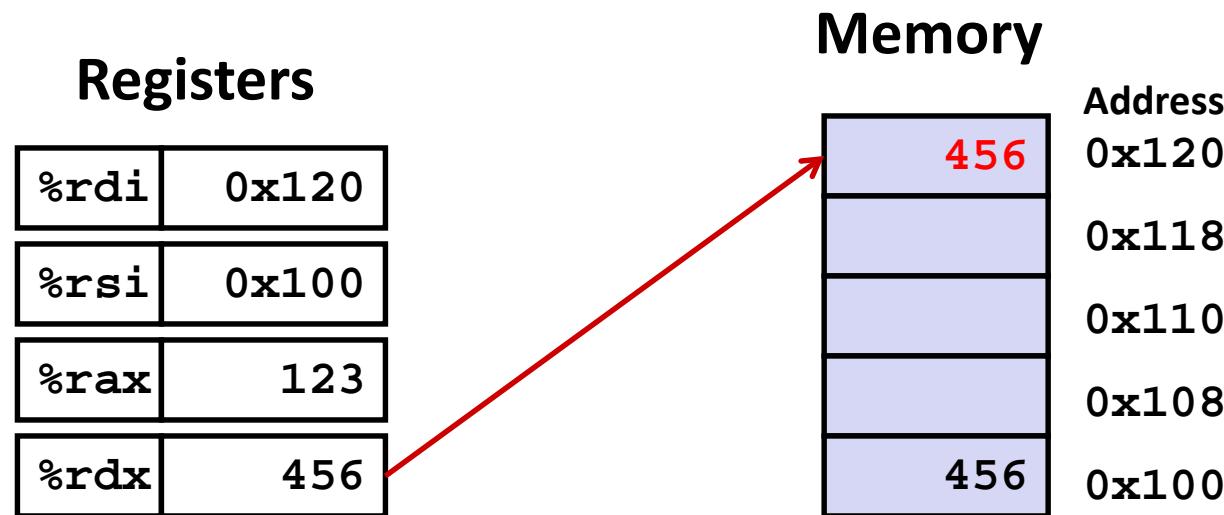
Comprendiendo swap()



swap:

```
    movq    (%rdi), %rax    # t0 = *xp
    movq    (%rsi), %rdx    # t1 = *yp
    movq    %rdx, (%rdi)    # *xp = t1
    movq    %rax, (%rsi)    # *yp = t0
    ret
```

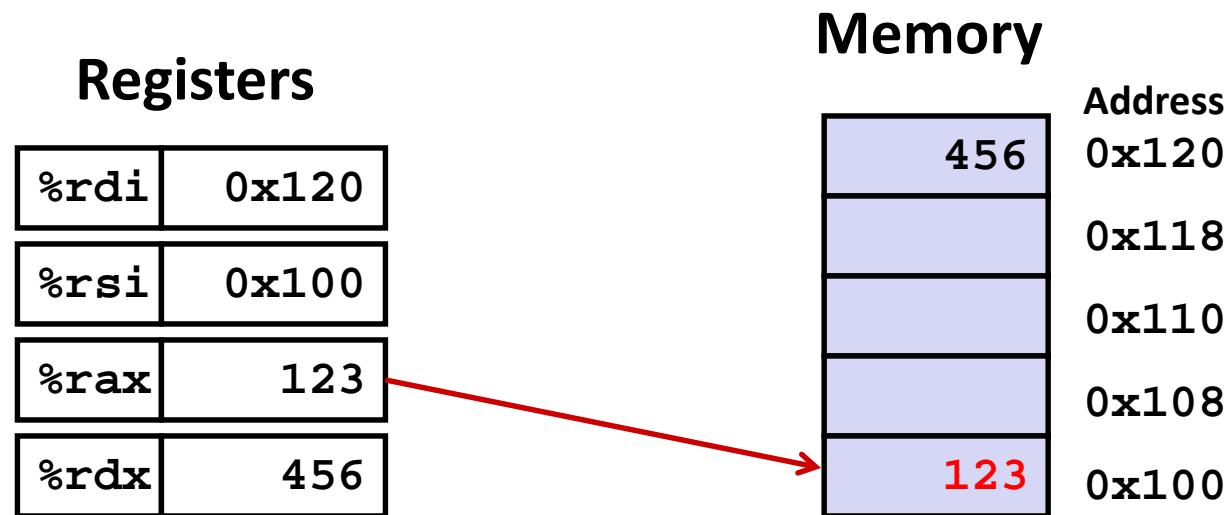
Comprendiendo swap()



swap:

```
    movq    (%rdi), %rax    # t0 = *xp
    movq    (%rsi), %rdx    # t1 = *yp
    movq    %rdx, (%rdi)    # *xp = t1
    movq    %rax, (%rsi)    # *yp = t0
    ret
```

Comprendiendo swap()



swap:

```
    movq    (%rdi), %rax    # t0 = *xp
    movq    (%rsi), %rdx    # t1 = *yp
    movq    %rdx, (%rdi)    # *xp = t1
    movq    %rax, (%rsi)    # *yp = t0
    ret
```

Modos Direcccionamiento a memoria sencillos

■ Normal[‡]

(R)

Mem[Reg[R]]

- El registro R indica la dirección de memoria
- ¡Exacto! Como seguir (*desreferenciar*[†]) un puntero en C

`movq (%rcx), %rax`

■ Desplazamiento D(R)

Mem[Reg[R]+D]

- El registro R indica el inicio de una región de memoria
- La constante de desplazamiento D indica el *offset*^{*}

`movq 8(%rbp), %rdx`

Modos Direcccionamiento a memoria completos

■ Forma más general

$$D(Rb, Ri, S) \quad \text{Mem}[Reg[Rb]+S*Reg[Ri]+ D]$$

- D: “Desplazamiento” constante 1, 2, ó 4 bytes
- Rb: Registro base: Cualquiera de los 16 registros enteros
- Ri: Registro índice: Cualquiera, excepto %rsp
- S: Factor de escala: 1, 2, 4, ú 8 (*¿por qué esos números?*)

■ Casos Especiales

$$(Rb, Ri) \quad \text{Mem}[Reg[Rb]+Reg[Ri]]$$

$$D(Rb, Ri) \quad \text{Mem}[Reg[Rb]+Reg[Ri]+D]$$

$$(Rb, Ri, S) \quad \text{Mem}[Reg[Rb]+S*Reg[Ri]]$$

Ejemplos de Cálculo de Direcciones

| | |
|------|--------|
| %rdx | 0xf000 |
| %rcx | 0x0100 |

D(Rb,Ri,S)

- D: “Desplazamiento” constante 1, 2, ó 4 bytes
- Rb: Registro base: Cualquiera de los 16 registros enteros
- Ri: Registro índice: Cualquiera, excepto %rsp
- S: Factor de escala: 1, 2, 4, ó 8 (*¿por qué esos números?*)

Mem[Reg[Rb]+S*Reg[Ri]+ D]

| Expresión | Cálculo de Dirección | Dirección |
|---------------|----------------------|-----------|
| 0x8(%rdx) | 0xf000 + 0x8 | 0xf008 |
| (%rdx,%rcx) | 0xf000 + 0x100 | 0xf100 |
| (%rdx,%rcx,4) | 0xf000 + 4*0x100 | 0xf400 |
| 0x80(,%rdx,2) | 2*0xf000 + 0x80 | 0x1e080 |

Programación Máquina I: Conceptos Básicos

- Historia de los procesadores y arquitecturas de Intel
- Lenguaje C, ensamblador, código máquina
- Conceptos básicos asm: Registros, operandos, move
- Operaciones aritméticas y lógicas

Instrucción para el Cálculo de Direcciones

■ **leaq Src, Dest^t**

- *Src* es cualquier expresión de modo direccionamiento (a memoria)
- Ajusta *Dest* a la dirección indicada por la expresión

■ **Usos**

- Calcular direcciones sin hacer referencias a memoria
 - P.ej., traducción de $p = \&x[i]$;
- Calcular expresiones aritméticas de la forma $x + k*y$
 - $k = 1, 2, 4 \text{ ú } 8$

■ **Ejemplo**

```
long m12(long x)
{
    return x*12;
}
```

Traducción a ASM por el compilador:

```
leaq (%rdi,%rdi,2), %rax # t <- x+x*2
salq $2, %rax           # return t<<2
```

Algunas Operaciones Aritméticas

■ Instrucciones de Dos Operandos:

| <i>Formato</i> | <i>Operación[†]</i> | |
|-----------------------|------------------------------|-----------------------------|
| addq <i>Src,Dest</i> | Dest = Dest + Src | |
| subq <i>Src,Dest</i> | Dest = Dest – Src | |
| imulq <i>Src,Dest</i> | Dest = Dest * Src | |
| salq <i>Src,Dest</i> | Dest = Dest << Src | <i>También llamada shlq</i> |
| sarq <i>Src,Dest</i> | Dest = Dest >> Src | <i>Aritméticas</i> |
| shrq <i>Src,Dest</i> | Dest = Dest >> Src | <i>Lógicas</i> |
| xorq <i>Src,Dest</i> | Dest = Dest ^ Src | |
| andq <i>Src,Dest</i> | Dest = Dest & Src | |
| orq <i>Src,Dest</i> | Dest = Dest Src | |

- ¡Cuidado con el orden de los argumentos! (Intel vs. AT&T)
- No se distingue entre enteros con/sin signo (*¿por qué?*)

Hora de hacer un test!

Conectarse a:

<https://swad.ugr.es/es?crs=5101>

Algunas Operaciones Aritméticas

■ Instrucciones de Un Operando:

| <i>Formato</i> | | <i>Operación</i> |
|----------------|-------------|--------------------|
| incq | <i>Dest</i> | $Dest = Dest + 1$ |
| decq | <i>Dest</i> | $Dest = Dest - 1$ |
| negq | <i>Dest</i> | $Dest = - Dest$ |
| notq | <i>Dest</i> | $Dest = \sim Dest$ |

■ Consultar más instrucciones en el libro†

- Aritméticas: [i]mulq Src, [i]divq Src, cqto
- Transferencia: movX (bwlq), movabsq,
movzXX (bw,bl,bq,wl,wq),‡
movsXX (bw,bl,bq,wl,wq,lq), cltq,
pushq, popq

† Según cómo se cuenten, hay entre 2000-3700 instrucciones en el manual

‡ luego veremos que movlq %eax, %rbx
sería lo mismo que mov %eax, %ebx 46

Ejemplo de Expresiones Aritméticas

```
long arith  
(long x, long y, long z)  
{  
    long t1 = x+y;  
    long t2 = z+t1;  
    long t3 = x+4;  
    long t4 = y * 48;  
    long t5 = t3 + t4;  
    long rval = t2 * t5;  
    return rval;  
}
```

arith:

| | |
|-------|---------------------|
| leaq | (%rdi,%rsi), %rax |
| addq | %rdx, %rax |
| leaq | (%rsi,%rsi,2), %rdx |
| salq | \$4, %rdx |
| leaq | 4(%rdi,%rdx), %rcx |
| imulq | %rcx, %rax |
| ret | |

Instrucciones interesantes

- **leaq**: cálculo de direcciones
- **salq**: desplazamiento aritmético
- **imulq**: multiplicación
 - pero sólo se usa una vez

Comprendiendo arith()

```
long arith
(long x, long y, long z)
{
    long t1 = x+y;
    long t2 = z+t1;
    long t3 = x+4;
    long t4 = y * 48;
    long t5 = t3 + t4;
    long rval = t2 * t5;
    return rval;
}
```

arith:

| | | |
|--------------|----------------------------|---------------|
| leaq | (%rdi,%rsi), %rax | # t1 |
| addq | %rdx, %rax | # t2 |
| leaq | (%rsi,%rsi,2), %rdx | |
| salq | \$4, %rdx | # t4 |
| leaq | 4(%rdi,%rdx), %rcx | # t5 |
| imulq | %rcx, %rax | # rval |
| ret | | |

| Registro | Uso(s) |
|----------|---------------------|
| %rdi | Argumento x |
| %rsi | Argumento y |
| %rdx | Argumento z |
| %rax | t1, t2, rval |
| %rdx | t4 |
| %rcx | t5 |

Programación a Nivel-Máquina I: Resumen

- **Historia de los procesadores y arquitecturas de Intel**
 - Diseño evolutivo lleva a demasiados artefactos y peculiaridades
- **Lenguaje C, ensamblador, código máquina**
 - Nuevas formas de estado visible[†]: contador de programa, registros, ...
 - El compilador debe transformar sentencias, expresiones, procedimientos, en secuencias de instrucciones a bajo nivel
- **Conceptos básicos asm: Registros, operandos, move**
 - Las instrucciones x86-64 `mov` cubren un amplio rango de variedades de movimientos de datos (transferencia)
- **Operaciones aritméticas y lógicas**
 - El compilador C saldrá con diversas combinaciones de instrucciones para realizar los cálculos

Guía de trabajo autónomo (4h/s)

■ **Estudio:** del Cap.3 CS:APP (Bryant/O'Hallaron)

- Historical perspective, Program Encodings
 - § 3.1 – 3.2 pp.199-213
- Data Formats, Accessing Info.
 - § 3.3 – 3.4 pp.213-227
- Arithmetic and Logical Operations
 - § 3.5 pp.227-236

■ **Ejercicios:** del Cap.3 CS:APP (Bryant/O'Hallaron)

- Probl. 3.1 – 3.5 § 3.4, pp.218, 221, 222, 223, 225
- Probl. 3.6 – 3.12 § 3.5, pp.228, 229, 230, 231, 232, 233, 236

Bibliografía:

[BRY16] Cap.3

Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016

Signatura ESIIT/[C.1 BRY com](#)

Programación a Nivel-Máquina II: Control

Estructura de Computadores
Semana 4

Bibliografía:

[BRY16] Cap.3 Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016
 Signatura ESIIT/[C.1 BRY com](#)

Transparencias del libro CS:APP, Cap.3

Introduction to Computer Systems: a Programmer's Perspective

Autores: Randal E. Bryant y David R. O'Hallaron

<http://www.cs.cmu.edu/afs/cs/academic/class/15213-f15/www/schedule.html>

Guía de trabajo autónomo (4h/s)

■ Lectura: del Cap.3 CS:APP (Bryant/O'Hallaron)

- Control
 - § 3.6 pp.236-274

■ Ejercicios: del Cap.3 CS:APP (Bryant/O'Hallaron)

- Probl. 3.13 – 3.14 § 3.6.2, pp.240, 241
- Probl. 3.15 § 3.6.4, pp.245[†]
- Probl. 3.16 – 3.18 § 3.6.5, pp.248₂, 249
- Probl. 3.19 – 3.21 § 3.6.6, pp.252[‡], 255₂
- Probl. 3.22 – 3.29 § 3.6.7, pp.257, 258, 260, 262, 264, 267₂, 268
- Probl. 3.30 – 3.31 § 3.6.8, pp.272, 273

Bibliografía:

[BRY16] Cap.3

Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016

Signatura ESIIT/C.1 BRY com

[†] direccionamiento relativo a contador de programa, “PC-relative”

[‡] penalización por predicción saltos 2

Programación Máquina II: Control

- **Control: Códigos de condición**
- **Saltos condicionales**
- **Bucles**
- **Sentencias switch**

Estado del Procesador (x86-64, Parcial)

■ Información sobre el programa ejecutándose actualmente

- Datos temporales (`%rax`, ...)
- Situación de la pila en tiempo de ejecución[†] (`%rsp`)
- Situación actual del contador de programa (`%rip`)
- Estado de comparaciones recientes (`CF, ZF, SF, OF`)

Tope de pila actual

Registros de propósito general

| | |
|-------------------|-------------------|
| <code>%rax</code> | <code>%r8</code> |
| <code>%rbx</code> | <code>%r9</code> |
| <code>%rcx</code> | <code>%r10</code> |
| <code>%rdx</code> | <code>%r11</code> |
| <code>%rsi</code> | <code>%r12</code> |
| <code>%rdi</code> | <code>%r13</code> |
| <code>%rsp</code> | <code>%r14</code> |
| <code>%rbp</code> | <code>%r15</code> |

`%rip` Puntero de instrucción[‡]

`CF` `ZF` `SF` `OF` Códigos de condición

[‡] “instruction pointer”, de ahí `%rip`

[†] “runtime stack”

Códigos de Condición (su ajuste implícito)

■ Registros de un solo bit[†]

- Ajustados implícitamente por las operaciones aritméticas
(interpretarlo como efecto colateral)

Ejemplo: $\text{addq } Src, Dest \leftrightarrow t = a+b$

CF puesto a 1 si sale acarreo del bit más significativo (desbord. op. sin signo)

ZF a1 **sii t == 0**

SF a 1 sii $t < 0$ (como número con signo)

OF a 1 sii desbord. en complemento a dos (desbord. op. con signo)
$$(a>0 \&\& b>0 \&\& t<0) \mid\mid (a<0 \&\& b<0 \&\& t>=0)$$

■ No afectados por la instrucción `lea`

[†] “flag” = “bandera”, deberíamos traducir “flag” por “indicador”, pero se suele dejar así, “optimization flags” debería ser “modificador/comutador”, también se suele dejar así.

*# “overflow” = “desbordamiento”, y los otros
“carry/zero/sign” = “acarreo/cero/signo” 5*

Códigos de Condición (ajuste explícito: Compare)

■ Ajuste Explícito mediante la Instrucción Compare

- `cmpq Src2, Src1`
- `cmpq b,a` equivale a restar $a-b$ pero sin ajustar el destino

- **CF a 1** sii sale acarreo del MSB[†] (c_n) (hacer caso cuando comp. sin signo)
- **ZF a 1** sii $a == b$
- **SF a 1** sii $(a-b) < 0$ (como número con signo)
- **OF a 1** sii overflow[‡] en complemento a dos (atender si comp. con signo)
 $(a>0 \&\& b<0 \&\& (a-b)<0) \mid\mid (a<0 \&\& b>0 \&\& (a-b)>0)$
definición overflow OF = $(c_n \wedge c_{n-1})$ mientras que acarreo CF = c_n

[†] “MSB” = bit más significativo

[‡] dejar sin traducir “overflow” ayuda didácticamente a distinguirlo
del acarreo (desbordamiento sin signo) al recordar los flags OF/CF 6

Códigos de Condición (ajuste explícito: Test)

■ Ajuste Explícito mediante la Instrucción Test

- `testq Src2, Src1`
- `testq b,a` equivale a hacer `a&b` pero sin ajustar el destino

- **ZF a 1** si $(a \& b) == 0$
- **SF a 1** si $(a \& b) < 0$

- Ajusta los códigos de condición según el valor de *Src1* & *Src2*
- Útil cuando uno de los operandos es una máscara
- Para comprobar si un valor es 0, gcc usa `testq`, no `cmpq`

```
    cmpq $0, %rax  
    testq %rax, %rax
```

Consultando Códigos de Condición

■ Instrucciones SetCC Dest

- Ajustar el byte destino a 0/1 según el código de condición indicado con CC[†] (combinación de flags deseada)
- *Dst registro* debe ser tamaño byte, *Dst memoria* sólo se modifica 1^{er} LSByte[†]

| SetCC | Condición | Descripción |
|-------|--------------|--------------------------|
| sete | ZF | Equal / Zero |
| setne | ~ZF | Not Equal / Not Zero |
| sets | SF | Sign (negativo) |
| setns | ~SF | Not Sign |
| setg | ~(SF^OF)&~ZF | Greater (signo) |
| setge | ~(SF^OF) | Greater or Equal (signo) |
| setl | (SF^OF) | Less (signo) |
| setle | (SF^OF) ZF | Less or Equal (signo) |
| seta | ~CF&~ZF | Above (sin signo) |
| setb | CF | Below (sin signo) |

"CC" = "condition code"

† "LSByte" = byte menos significativo, 8

Registros enteros x86-64

| | | | |
|------|------|------|-------|
| %rax | %al | %r8 | %r8b |
| %rbx | %bl | %r9 | %r9b |
| %rcx | %cl | %r10 | %r10b |
| %rdx | %dl | %r11 | %r11b |
| %rsi | %sil | %r12 | %r12b |
| %rdi | %dil | %r13 | %r13b |
| %rsp | %spl | %r14 | %r14b |
| %rbp | %bpl | %r15 | %r15b |

- Se puede referenciar el LSByte, tiene nombre de registro propio

Consultando Códigos de Condición (Cont.)

■ Instrucciones SetCC Dest

- Ajustar un byte suelto *Dest* según el código de condición

■ Uno de los registros byte direccionables

- No se alteran los restantes bytes
- Típicamente se usa `movzbl†` para terminar trabajo
 - las instrucciones de 32-bit también ponen los 32 MSB a 0

```
int gt (long x, long y)
{
    return x > y;
}
```

| Registro | Uso(s) |
|----------|--------------------|
| %rdi | Argumento x |
| %rsi | Argumento y |
| %eax | Valor de retorno |

```
gt:
    cmpq    %rsi, %rdi  # Comparar x:y
    setg    %al           # Poner a 1 si >
†   movzbl  %al, %eax # Resto %rax a cero
    ret
```

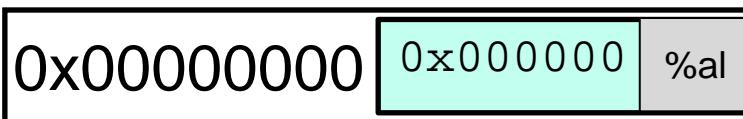
[†] "Move with Zero-extend Byte to Long" mnemotécnico MOVZX según Intel 10

Consultando Códigos de Condición (Cont.)

■ Instrucciones SetCC Dest

- A Las operaciones que modifican un registro de 32 bits, ponen a 0 el resto hasta 64 bits
- Un
- N
- T

movzbl %al, %eax



gt:

Puesto todo a 0

† **movzbl %al, %eax # Resto %rax a cero**
ret

+ "Move with Zero-extend Byte to Long"
mnemotécnico MOVZX según Intel 11

Programación Máquina II: Control

- Control: Códigos de condición
- Saltos condicionales
- Bucles
- Sentencias switch

Saltos

■ Instrucciones jCC

- Saltar a otro lugar del código si se cumple el código de condición CC

| jCC | Condición | Descripción |
|-----|--------------|--------------------------|
| jmp | 1 | Incondicional |
| je | ZF | Equal / Zero |
| jne | ~ZF | Not Equal / Not Zero |
| js | SF | Sign (negativo) |
| jns | ~SF | Not Sign |
| jg | ~(SF^OF)&~ZF | Greater (signo) |
| jge | ~(SF^OF) | Greater or Equal (signo) |
| jl | (SF^OF) | Less (signo) |
| jle | (SF^OF) ZF | Less or Equal (signo) |
| ja | ~CF&~ZF | Above (sin signo) |
| jb | CF | Below (sin signo) |

Ejemplo de Salto Condicional (al viejo estilo)

■ Generación[†]

```
shark> gcc -Og -S -fno-if-conversion control.c
```

```
long absdiff(long x, long y)
{
    long result;
    if (x > y)
        result = x-y;
    else
        result = y-x;
    return result;
}
```

`absdiff:`

```

    cmpq    %rsi, %rdi  # x:y
    jle     .L4
    movq    %rdi, %rax
    subq    %rsi, %rax
    ret
.L4:                                # x <= y
    movq    %rsi, %rax
    subq    %rdi, %rax
    ret

```

| Registro | Uso(s) |
|----------|--------------------|
| %rdi | Argumento x |
| %rsi | Argumento y |
| %rax | Valor de retorno |

[†] en el gcc-7.3.0 que viene con Ubuntu-18.04

-fif-conversion activada para -O123s pero no para -Og 14

Expresándolo con código Goto

- C permite la sentencia goto
- Salta a la posición indicada por la etiqueta

```
long absdiff(long x, long y)
{
    long result;
    if (x > y)
        result = x-y;
    else
        result = y-x;
    return result;
}
```

```
long absdiff_j(long x, long y)
{
    long result;
    int ntest = x <= y;
    if (ntest) goto Else;
    result = x-y;
    goto Done;
Else:
    result = y-x;
Done:
    return result;
}
```

Traducción en General Expresión Condicional (usando saltos)

Código C

```
val = Test ? Then_Expr : Else_Expr;
```

```
val = x>y ? x-y : y-x;
```

Versión Goto

```
ntest = !Test;  
if (ntest) goto Else;  
val = Then_Expr;  
goto Done;  
Else:  
    val = Else_Expr;  
Done:  
    . . .
```

- Crear regiones de código separadas para las expresiones Then y Else
- Ejecutar sólo la adecuada

Usando Movimientos Condicionales

■ Instrucciones Mvmt. Condicional

- Las instrucciones implementan:
 $\text{if } (\text{Test}) \text{ Dest} \leftarrow \text{Src}$
- En procesadores x86 posteriores a 1995
(Pentium Pro/II)
- GCC intenta utilizarlas
 - Pero sólo cuando sepa que es seguro

■ ¿Por qué?

- Ramificaciones muy perjudiciales para flujo instrucciones en cauces[†]
- Movimiento condicional no requiere transferencia de control

Código C

```
val = Test
    ? Then_Expr
    : Else_Expr;
```

Versión Goto

```
result = Then_Expr;
eval = Else_Expr;
nt = !Test;
if (nt) result = eval;
return result;
```

Ejemplo de Movimiento Condicional[†]

```
long absdiff(long x, long y)
{
    long result;
    if (x > y)
        result = x-y;
    else
        result = y-x;
    return result;
}
```

| Registro | Uso(s) |
|----------|--------------------|
| %rdi | Argumento x |
| %rsi | Argumento y |
| %rax | Valor de retorno |

absdiff:

| | | |
|---------|------------|------------------------|
| movq | %rdi, %rax | # x |
| subq | %rsi, %rax | # result = x-y |
| movq | %rsi, %rdx | |
| subq | %rdi, %rdx | # eval = y-x |
| cmpq | %rsi, %rdi | # x:y |
| cmovele | %rdx, %rax | # if <=, result = eval |
| ret | | |

[†] generar con `gcc -fif-conversion -Og -S control.c`
 ó incluso con `gcc -O -S control.c`
 en el `gcc-7.3.0` que viene con `Ubuntu-18.04` 18

Malos Casos para Movimientos Condicionales

- Recordar que se calculan ambos valores

Cálculos costosos

```
val = Test(x) ? Hard1(x) : Hard2(x);
```

- Sólo tiene sentido cuando son cálculos muy sencillos

Cálculos arriesgados

```
val = p ? *p : 0;
```

- Pueden tener efectos no deseables

Cálculos con efectos colaterales

```
val = x > 0 ? x*=7 : x+=3;
```

- No deberían tener efectos colaterales

Ejercicio

`cmpq b,a` equiv. restar $t=a-b$ pero sin ajustar destino

- **CF a 1** si $c_n == 1$, porque $a < b$ sin signo
- **ZF a 1** si $t == 0$, porque $a == b$
- **SF a 1** si $t_n == 1$, porque $a < b$ con signo
- **OF a 1** si $(c_n \wedge c_{n-1}) == 1$, pq. $a-b$ (signo) mal hecha

| SetCC | Condición | Descripción |
|-------|-------------------------------------|--------------------------|
| sete | ZF | Equal / Zero |
| setne | $\sim ZF$ | Not Equal / Not Zero |
| sets | SF | Sign (negativo) |
| setns | $\sim SF$ | Not Sign |
| setg | $\sim(SF \wedge OF) \wedge \sim ZF$ | Greater (signo) |
| setge | $\sim(SF \wedge OF)$ | Greater or Equal (signo) |
| setl | $(SF \wedge OF)$ | Less (signo) |
| setle | $(SF \wedge OF) \mid ZF$ | Less or Equal (signo) |
| seta | $\sim CF \wedge \sim ZF$ | Above (sin signo) |
| setb | CF | Below (sin signo) |

| | |
|---------------------|-------------------------|
| <code>xorq</code> | <code>%rax, %rax</code> |
| <code>subq</code> | <code>\$1, %rax</code> |
| <code>cmpq</code> | <code>\$2, %rax</code> |
| <code>setl</code> | <code>%al</code> |
| <code>movzbl</code> | <code>%al, %eax</code> |

| %rax | SF | CF | OF | ZF |
|------|----|----|----|----|
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

Notar: `setl` y `movzblq` no ajustan códigos de condición

Ejercicio

`cmpq b,a` equiv. restar $t=a-b$ pero sin ajustar destino

- **CF a 1** si $c_n == 1$, porque $a < b$ sin signo
- **ZF a 1** si $t == 0$, porque $a == b$
- **SF a 1** si $t_n == 1$, porque $a < b$ con signo
- **OF a 1** si $(c_n \wedge c_{n-1}) == 1$, pq. $a-b$ (signo) mal hecha

| SetCC | Condición | Descripción | |
|-------|---------------------------------|----------------------|-------------|
| sete | ZF | Equal / Zero | |
| setne | $\sim ZF$ | Not Equal / Not Zero | |
| sets | SF | Sign (negativo) | |
| setns | $\sim SF$ | Not Sign | |
| setg | $\sim(SF \wedge OF) \& \sim ZF$ | Greater | (signo) |
| setge | $\sim(SF \wedge OF)$ | Greater or Equal | (signo) |
| setl | $(SF \wedge OF)$ | Less | (signo) |
| setle | $(SF \wedge OF) \mid ZF$ | Less or Equal | (signo) |
| seta | $\sim CF \& \sim ZF$ | Above | (sin signo) |
| setb | CF | Below | (sin signo) |

| | <code>xorq %rax, %rax</code> | <code>subq \$1, %rax</code> | <code>cmpq \$2, %rax</code> | <code>setl %al</code> | <code>movzbl %al, %eax</code> |
|--|------------------------------|-----------------------------|-----------------------------|-----------------------|-------------------------------|
| | | | | | |

| %rax | SF | CF | OF | ZF |
|-----------------------|----|----|----|----|
| 0x0000 0000 0000 0000 | 0 | 0 | 0 | 1 |
| 0xFFFF FFFF FFFF FFFF | 1 | 1 | 0 | 0 |
| 0xFFFF FFFF FFFF FFFF | 1 | 0 | 0 | 0 |
| 0xFFFF FFFF FFFF FF01 | 1 | 0 | 0 | 0 |
| 0x0000 0000 0000 0001 | 1 | 0 | 0 | 0 |

Notar: `setl` y `movzbl` no ajustan códigos de condición

Programación Máquina II: Control

- Control: Códigos de condición
- Saltos condicionales
- Bucles
- Sentencias switch

Ejemplo de bucle “Do-While”

Código C

```
long pcount_do
(unsigned long x) {
    long result = 0;
    do {
        result += x & 0x1;
        x >>= 1;
    } while (x);
    return result;
}
```

Versión Goto

```
long pcount_goto
(unsigned long x) {
    long result = 0;
loop:
    result += x & 0x1;
    x >>= 1;
    if(x) goto loop;
    return result;
}
```

- Contar el número de 1's en el argumento x (“popcount” [†])
- Usar salto condicional para seguir iterando o salir del bucle

[†] “population count”= peso Hamming,
distancia Hamming (al 0), suma lateral... 23

Compilación del bucle “Do-While”

Versión Goto

```
long pcount_goto
(unsigned long x) {
    long result = 0;
loop:
    result += x & 0x1;
    x >>= 1;
    if(x) goto loop;
    return result;
}
```

| Registro | Uso(s) |
|----------|--------------------|
| %rdi | Argumento x |
| %rax | result |

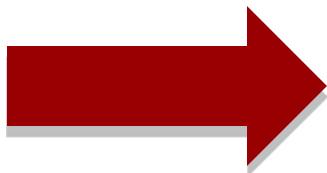
```
    movl    $0, %eax      # result = 0
.L2:                           # loop:
    movq    %rdi, %rdx
    andl    $1, %edx      # t = x & 0x1
    addq    %rdx, %rax    # result += t
    shrq    %rdi          # x >>= 1
    jne     .L2           # if (x) goto loop
    rep; ret
```

*+ problema predicción saltos Opteron y Athlon 64 (2003-2005) en RET 1B tras flowctrl.
Software Optimization Guide for AMD64 Family 10-12h (2010-2011) recomienda RET O*

Traducción en General de “Do-While”

Código C

```
do  
    Body  
    while ( Test );
```



Versión Goto

```
loop:  
    Body  
    if ( Test )  
        goto loop
```

■ **Body:** {
 Sentencia₁;
 Sentencia₂;
 ...
 Sentencia_n;
}

“body” = cuerpo,
“statement” = sentencia,
“test” = comprobación.

Traducción en General de “While” (#1)

Código C

```
while ( Test )  
    Body
```



Versión Goto

```
goto test;  
loop:  
    Body  
test:  
    if ( Test )  
        goto loop;  
done:
```

- Traducción tipo “salta-en-medio” †
- Usada con -O0 / -Og

Ejemplo de bucle “While” (#1)

Código C

```
long pcount_while  
  (unsigned long x) {  
    long result = 0;  
    while (x) {  
        result += x & 0x1;  
        x >>= 1;  
    }  
    return result;  
}
```

Salta-en-medio[†]

```
long pcount_goto_jtm  
  (unsigned long x) {  
    long result = 0;  
    goto test;  
loop:  
    result += x & 0x1;  
    x >>= 1;  
test:  
    if(x) goto loop;  
    return result;  
}
```

- Comparar con la versión do-while de la misma función
- El goto inicial empieza el bucle por **test** (“en medio”)

Traducción en General de “While” (#2)

Versión While

```
while ( Test )
    Body
```



Versión Do-While

```
if ( !Test )
    goto done;
do
    Body
    while( Test );
done:
```

- Traducción tipo “copia-test”
 - Conversión a “do-while”
- Usada con $-O1^t$

Versión Goto

```
if ( !Test )
    goto done;
loop:
    Body
    if ( Test )
        goto loop;
done:
```



^t evitar test al principio con gcc -O123 -fno-tree-ch

ó con gcc -Os -fno-reorder-blocks 28

Ejemplo de bucle “While” (#2)

Código C

```
long pcount_while
(unsigned long x) {
    long result = 0;
    while (x) {
        result += x & 0x1;
        x >>= 1;
    }
    return result;
}
```

Copia-test

```
long pcount_goto_ct
(unsigned long x) {
    long result = 0;
    if (!x) goto done;
loop:
    result += x & 0x1;
    x >>= 1;
    if(x) goto loop;
done:
    return result;
}
```

- Comparar con la versión do-while de la misma función
- El primer condicional guarda la entrada al bucle

Forma del bucle “For”

Forma General

```
for (Init; Test; Update )  
    Body
```

```
#define WSIZE 8*sizeof(int)  
long pcount_for(unsigned long x)  
{  
    size_t i;  
    long result = 0;  
    for (i = 0; i < WSIZE; i++)  
    {  
        unsigned bit =  
            (x >> i) & 0x1;  
        result += bit;  
    }  
    return result;  
}
```

Init

```
i = 0
```

Test

```
i < WSIZE
```

Update

```
i++
```

Body

```
{  
    unsigned bit =  
        (x >> i) & 0x1;  
    result += bit;  
}
```

“Init” = inicialización,
“test” = comprobación,
“update” = actualización,
“body” = cuerpo. 30

Bucle “For” → Bucle While

Versión For

```
for ( Init; Test; Update )  
    Body
```



Versión While

```
Init;  
  
while ( Test ) {  
    Body  
    Update;  
}
```

Conversión For-While

Init

```
i = 0
```

Test

```
i < WSIZE
```

Update

```
i++
```

Body

```
{  
    unsigned bit =  
        (x >> i) & 0x1;  
    result += bit;  
}
```

```
long pcount_for_while  
(unsigned long x)  
{  
    size_t i;  
    long result = 0;  
    i = 0;  
    while (i < WSIZE)  
    {  
        unsigned bit =  
            (x >> i) & 0x1;  
        result += bit;  
        i++;  
    }  
    return result;  
}
```

Conversión Bucle “For” a Do-While

Código C

```
long pcount_for
(unsigned long x)
{
    size_t i;
    long result = 0;
    for (i = 0; i < WSIZE; i++)
    {
        unsigned bit =
            (x >> i) & 0x1;
        result += bit;
    }
    return result;
}
```

Versión Goto

```
long pcount_for_goto_dw
(unsigned long x) {
    size_t i;
    long result = 0;
    i = 0;
if (! (i < WSIZE))
    goto done;
loop:
{
    unsigned bit =
        (x >> i) & 0x1;
    result += bit;
}
i++;
if (i < WSIZE)
    goto loop;
done:
return result;
}
```

Init

! Test

Body

Update

Test

- La comprobación inicial se puede optimizar (quitándola)

Programación Máquina II: Control

- Control: Códigos de condición
- Saltos condicionales
- Bucles
- Sentencias switch[†]

[†] “switch”=comutador,
tampoco traducimos
“for” o “while” 34

```
long switch_eg
    (long x, long y, long z)
{
    long w = 1;
    switch(x) {
        case 1:
            w = y*z;
            break;
        case 2:
            w = y/z;
            /* Fall Through */
        case 3:
            w += z;
            break;
        case 5:
        case 6:
            w -= z;
            break;
        default:
            w = 2;
    }
    return w;
}
```

Ejemplo de sentencia switch

- Múltiples etiquetas de caso
 - Aquí: 5 y 6
- Caídas en cascada[†]
 - Aquí: 2
- Casos ausentes[†]
 - Aquí: 4

[†] “fall-through cases”, “missing cases” 35

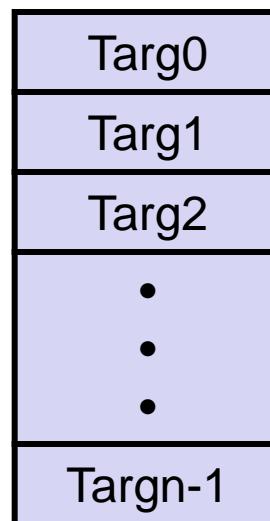
Estructura de una Tabla de Saltos

Forma switch

```
switch(x) {
    case val_0:
        Block 0
    case val_1:
        Block 1
    ...
    case val_n-1:
        Block n-1
}
```

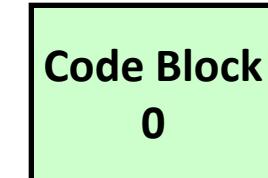
Tabla Saltos[†]

JTab:

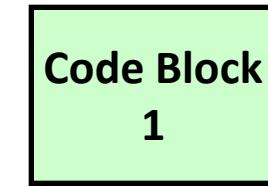


Destinos salto[†]

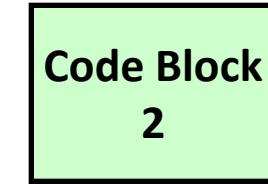
Targ0:



Targ1:

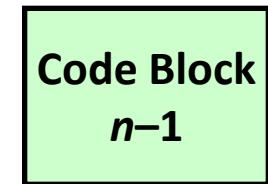


Targ2:



•
•
•

Targn-1:



Traducción aprox. (C ficticio)

```
goto *JTab[x];
```

[†] “jump table”, “jump targets” 36

Ejemplo de Sentencia Switch

```
long switch_eg
    (long x, long y, long z)
{
    long w = 1;
    switch(x) {
        . . .
    }
    return w;
}
```

| Registro | Uso(s) |
|----------|------------------|
| %rdi | Argumento x |
| %rsi | Argumento y |
| %rdx | Argumento z |
| %rax | Valor de retorno |

Inicialización:

```
switch_eg:
    movq %rdx, %rcx      # z → %rcx
    cmpq $6, %rdi         # x:6
    † ja .L8              # default: ←
    jmp * .L4(,%rdi,8)   # goto *Jtab[x]
```

Notar que w no se inicializa aquí

¿Qué rango de valores cubre default?

Ejemplo de Sentencia Switch

```
long switch_eg
    (long x, long y, long z)
{
    long w = 1;
    switch(x) {
        . . .
    }
    return w;
}
```

Tabla de saltos[†]

| | |
|----------|-------------|
| .section | .rodata |
| .align 8 | |
| .L4: | |
| .quad | .L8 # x = 0 |
| .quad | .L3 # x = 1 |
| .quad | .L5 # x = 2 |
| .quad | .L9 # x = 3 |
| .quad | .L8 # x = 4 |
| .quad | .L7 # x = 5 |
| .quad | .L7 # x = 6 |

Inicialización:

```
switch_eg:
    movq %rdx, %rcx      # z → %rcx
    cmpq $6, %rdi         # x:6
    † ja .L8              # default:
    jmp * .L4(,%rdi,8)   # goto *Jtab[x]
```

*Salto
indirecto*

Explicación Inicialización Ensamblador

■ Estructura de la Tabla

- Cada destino salto requiere 8 bytes
- Dirección base es .L4

■ Saltos

- **Directo:** `jmp .L8`
- Destino salto indicado por etiqueta .L8
- **Indirecto:** `jmp * .L4(,%rdi,8)`
- Inicio de la tabla de saltos: .L4
- Se debe escalar por un factor de 8 (direcciones ocupan 8 bytes)
- Captar destino salto desde la Dirección Efectiva $.L4 + x*8$
 - Sólo para $0 \leq x \leq 6$

Tabla de saltos

```
.section    .rodata
.align 8
.L4:
.quad      .L8    # x = 0
.quad      .L3    # x = 1
.quad      .L5    # x = 2
.quad      .L9    # x = 3
.quad      .L8    # x = 4
.quad      .L7    # x = 5
.quad      .L7    # x = 6
```

Tabla de Saltos

Tabla de saltos

```
.section .rodata
.align 8
.L4:
.quad .L8 # x = 0
.quad .L3 # x = 1
.quad .L5 # x = 2
.quad .L9 # x = 3
.quad .L8 # x = 4
.quad .L7 # x = 5
.quad .L7 # x = 6
```

```
switch(x) {
    case 1:          // .L3
        w = y*z;
        break;
    case 2:          // .L5
        w = y/z;
        /* Fall Through */
    case 3:          // .L9
        w += z;
        break;
    case 5:
    case 6:          // .L7
        w -= z;
        break;
    default:         // .L8
        w = 2;
}
```

Bloques de Código ($x == 1$)

```
switch(x) {  
    case 1:      // .L3  
        w = y*z;  
        break;  
    . . .  
}
```

.L3:

```
    movq    %rsi, %rax  # y  
    imulq   %rdx, %rax  # y*z  
    ret
```

| Registro | Uso(s) |
|----------|--------------------|
| %rdi | Argumento x |
| %rsi | Argumento y |
| %rdx | Argumento z |
| %rax | Valor de retorno |

Tratamiento de Caídas en Cascada

```
long w = 1;  
.  
.  
switch(x) {  
.  
. . .  
case 2:  
    w = y/z;  
    /* Fall Through */  
case 3:  
    w += z;  
    break;  
.  
.  
}
```

case 2:
 w = y/z;
 goto merge;

case 3:
 w = 1;

merge:
 w += z;

Bloques de Código ($x == 2$, $x == 3$)

```

long w = 1;
    . . .
switch(x) {
    . . .
case 2:
    w = y/z;
    /* Fall Through */
case 3:
    w += z;
    break;
    . . .
}

```

```

.L5:                                # Case 2
    movq    %rsi, %rax
    † cqto
    idivq   %rcx      # y/z
    jmp     .L6        # goto merge
.L9:                                # Case 3
    movl    $1, %eax    # w = 1
.L6:                                # merge:
    addq    %rcx, %rax # w += z
    ret

```

| Registro | Uso(s) |
|----------|------------------|
| %rdi | Argumento x |
| %rsi | Argumento y |
| %rdx | Argumento z |
| %rax | Valor de retorno |

mnemotécnico CQO según Intel

† “Convert Quad to Oct”

Bloques de Código ($x == 5$, $x == 6$, default)

```
switch(x) {  
    . . .  
    case 5: // .L7  
    case 6: // .L7  
        w -= z;  
        break;  
    default: // .L8  
        w = 2;  
}
```

```
.L7:                      # Case 5,6  
    movl $1, %eax      # w = 1  
    subq %rdx, %rax   # w -= z  
    ret  
.L8:                      # Default:  
    movl $2, %eax      # 2  
    ret
```

| Registro | Uso(s) |
|----------|------------------|
| %rdi | Argumento x |
| %rsi | Argumento y |
| %rdx | Argumento z |
| %rax | Valor de retorno |

Resumen

■ Control C

- if-then-else
- do-while
- while, for
- switch

■ Control Ensamblador

- Salto condicional
- Movimiento condicional
- Salto indirecto (mediante tablas de saltos)
- Compilador genera secuencia código p/implementar control más complejo

■ Técnicas estándar

- Bucles convertidos a forma do-while (ó salta-en medio ó copia-test)
- Sentencias switch grandes usan tablas de saltos
- Sentencias switch poco densas → árboles decisión (if-elseif-elseif-else)

Guía de trabajo autónomo (4h/s)

■ **Estudio:** del Cap.3 CS:APP (Bryant/O'Hallaron)

- Control
 - § 3.6 pp.236-274

■ **Ejercicios:** del Cap.3 CS:APP (Bryant/O'Hallaron)

- Probl. 3.13 – 3.14 § 3.6.2, pp.240, 241
- Probl. 3.15 § 3.6.4, pp.245[†]
- Probl. 3.16 – 3.18 § 3.6.5, pp.248₂, 249
- Probl. 3.19 – 3.21 § 3.6.6, pp.252[‡], 255₂
- Probl. 3.22 – 3.29 § 3.6.7, pp.257, 258, 260, 262, 264, 267₂, 268
- Probl. 3.30 – 3.31 § 3.6.8, pp.272, 273

Bibliografía:

[BRY16] Cap.3

Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016

Signatura ESIIT/C.1 BRY com

[†] direccionamiento relativo a contador de programa, “PC-relative”

[‡] penalización por predicción saltos

Programación a Nivel-Máquina III: Procedimientos

Estructura de Computadores
Semana 5

Bibliografía:

[BRY16] Cap.3 Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016
 Signatura ESIIT/[C.1 BRY com](#)

Transparencias del libro CS:APP, Cap.3

Introduction to Computer Systems: a Programmer's Perspective

Autores: Randal E. Bryant y David R. O'Hallaron

<http://www.cs.cmu.edu/afs/cs/academic/class/15213-f15/www/schedule.html>

Guía de trabajo autónomo (4h/s)

■ Lectura: del Cap.3 CS:APP (Bryant/O'Hallaron)

- Procedures
 - § 3.7 pp.274-291
- Understanding Pointers, Using GDB.
 - § 3.10.1-2 pp.312-316

■ Ejercicios: del Cap.3 CS:APP (Bryant/O'Hallaron)

- Probl. 3.32 § 3.7.2, pp.280
- Probl. 3.33 § 3.7.3, pp.282
- Probl. 3.34 § 3.7.5, pp.288
- Probl. 3.35 § 3.7.6, pp.290

Bibliografía:

[BRY16] Cap.3

Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016

Signatura ESIIT/[C.1 BRY com](#)

Programación Máquina III: Procedimientos

■ Procedimientos

- Mecanismos
- Estructura de la pila
- Convenciones de llamada
 - Pasando el control
 - Pasando los datos
 - Gestionando datos locales
- Ejemplos ilustrativos de Recursividad

Mecanismos en los Procedimientos

■ Transferencia de control

- al principio del código del procedimiento
- de vuelta al punto de retorno

■ Transferencia de datos

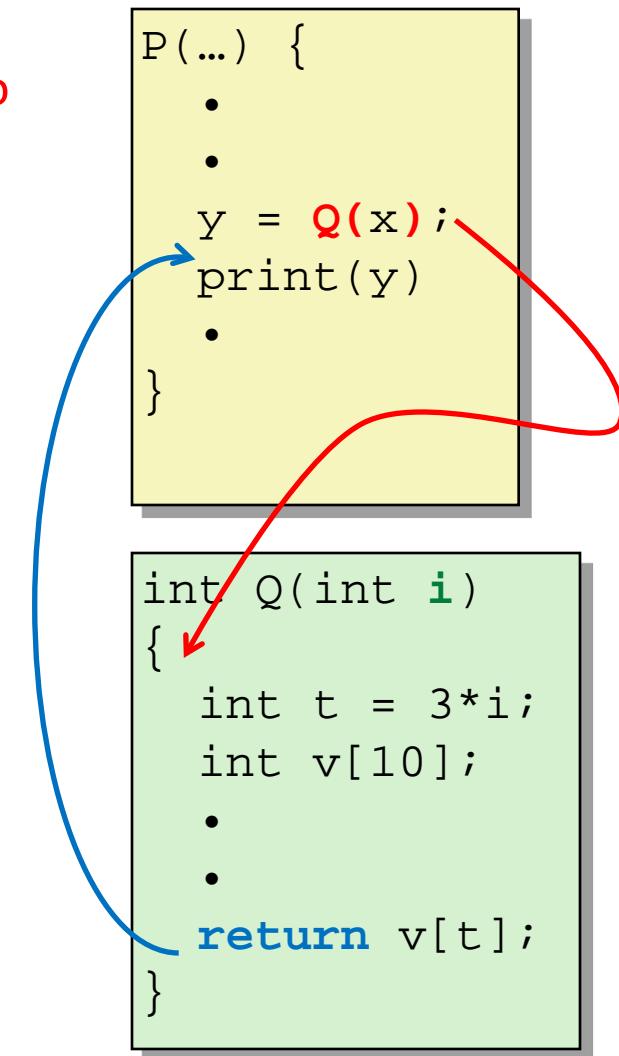
- Argumentos del procedimiento
- Valor de retorno

■ Gestión de memoria

- Reservar[†] durante ejecución procedimiento
- Liberar al retornar

■ Mecanismos todos implementados con instrucciones máquina

■ La implement. x86-64 de un proc. concreto usa sólo los mecanismos que éste requiera



Mecanismos en los Procedimientos

■ Transferencia de control

- al principio del código del procedimiento
- de vuelta al punto de retorno

■ Transferencia de datos

- Argumentos del procedimiento
- Valor de retorno

■ Gestión de memoria

- Reservar[†] durante ejecución procedimiento
- Liberar al retornar

■ Mecanismos todos implementados con instrucciones máquina

■ La implement. x86-64 de un proc. concreto usa sólo los mecanismos que éste requiera

```
P( ... ) {  
    .  
    .  
    y = Q(x);  
    print(y)  
    .  
}
```

```
int Q(int i)  
{  
    int t = 3*i;  
    int v[10];  
    .  
    .  
    return v[t];  
}
```

Mecanismos en los Procedimientos

■ Transferencia de control

- al principio del código del procedimiento
- de vuelta al punto de retorno

■ Transferencia de datos

- Argumentos del procedimiento
- Valor de retorno

■ Gestión de memoria

- Reservar[†] durante ejecución procedimiento
- Liberar al retornar

■ Mecanismos todos implementados con instrucciones máquina

■ La implement. x86-64 de un proc. concreto usa sólo los mecanismos que éste requiera

```
P(...) {  
    •  
    •  
    y = Q(x);  
    print(y)  
    •  
}
```

```
int Q(int i)  
{  
    int t = 3*i;  
    int v[10];  
    •  
    •  
    return v[t];  
}
```

Mecanismos en los Procedimientos

■ Transferencia de control

- al principio del código del procedimiento
- de vuelta al punto de retorno

■ Transferencia de datos

- Argumentos del procedimiento
- Valor de retorno

■ Gestión de memoria

- Reservar[†] durante ejecución procedimiento
- Liberar al retornar

■ Mecanismos todos implementados con instrucciones máquina

■ La implement. x86-64 de un proc. concreto usa sólo los mecanismos que éste requiera

```
P(...) {  
    .  
    .  
    y = Q(x);  
    print(y)  
    .  
}
```

```
int Q(int i)  
{  
    int t = 3*i;  
    int v[10];  
    .  
    .  
    return v[t];  
}
```

Programación Máquina III: Procedimientos

■ Procedimientos

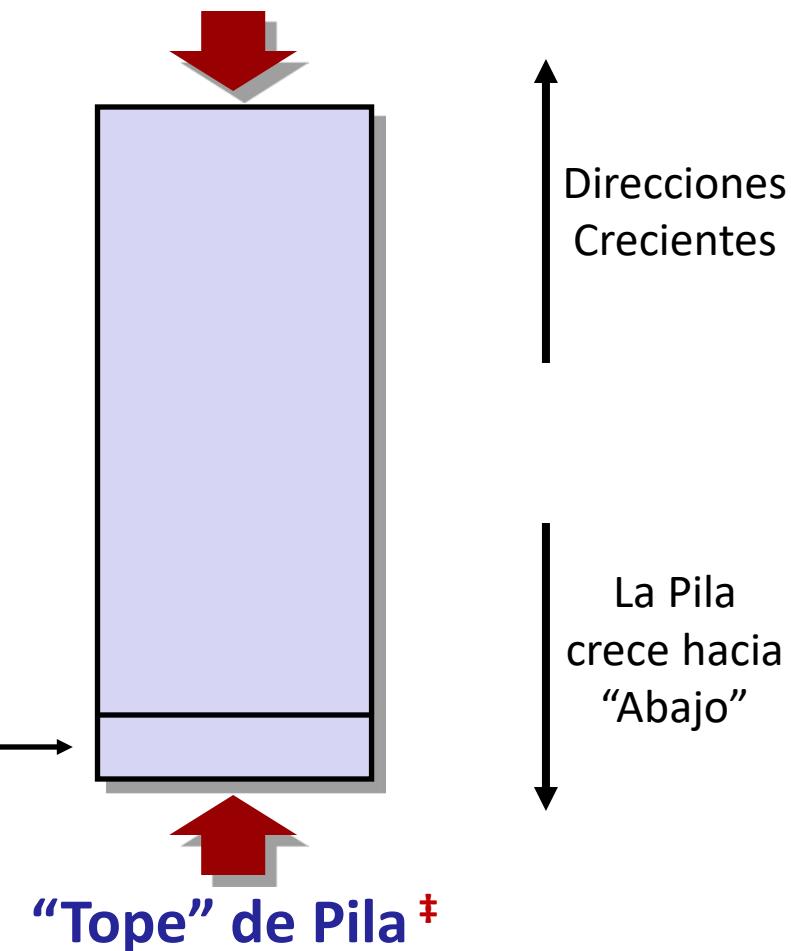
- Mecanismos
- Estructura de la pila
- Convenciones de llamada
 - Pasando el control
 - Pasando los datos
 - Gestionando datos locales
- Ejemplos ilustrativos de Recursividad

Pila x86-64

- Región de memoria gestionada con disciplina de pila
- Crece hacia posiciones inferiores
- El registro **%rsp** contiene la dirección más baja[†] de la pila
 - dirección del elemento “tope”

Puntero de Pila: **%rsp** →

“Fondo” de Pila[‡]



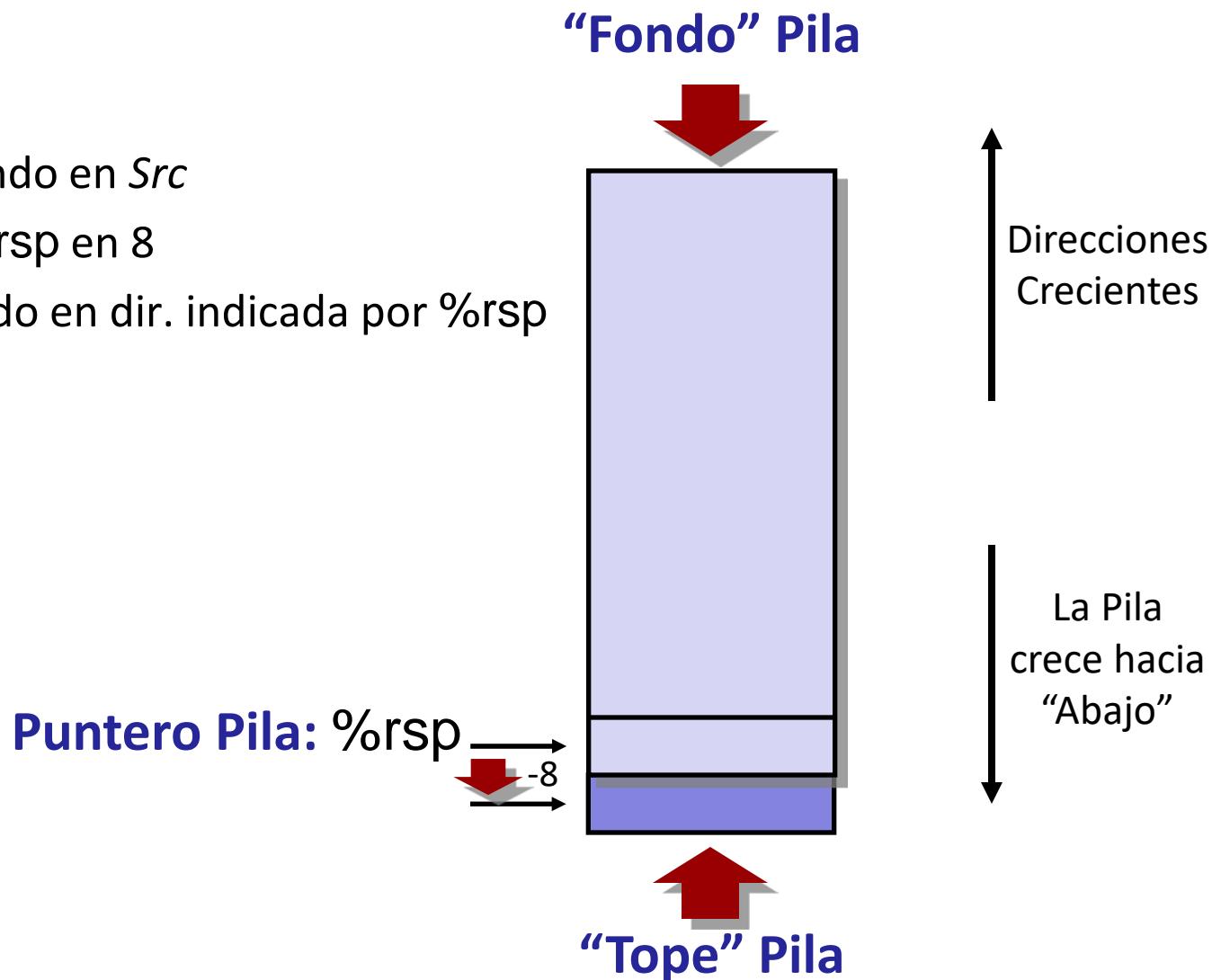
[†] “lowest” en el instante actual

[‡] “top” = tope, “bottom” = fondo 9

Pila x86-64: Push[†]

■ pushq Src

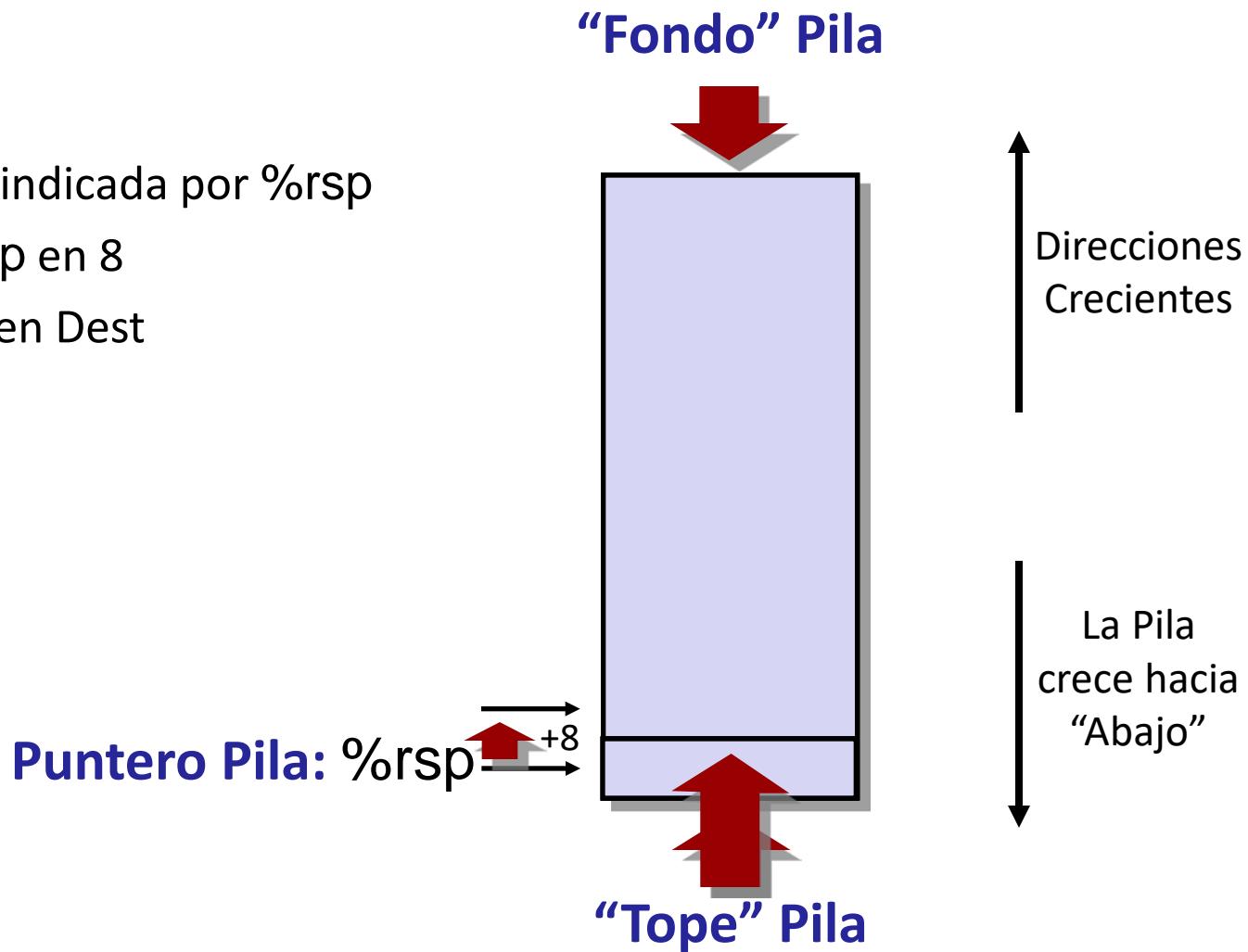
- Capta el operando en Src
- Decrementa %rsp en 8
- Escribe operando en dir. indicada por %rsp



Pila x86-64: Pop[†]

■ popq Dest

- Lee valor de dir. indicada por %rsp
- Incrementa %rsp en 8
- Almacena valor en Dest



Programación Máquina III: Procedimientos

■ Procedimientos

- Mecanismos
- Estructura de la pila
- Convenciones de llamada
 - **Pasando el control**
 - Pasando los datos
 - Gestionando datos locales
- Ejemplos ilustrativos de Recursividad

Código ejemplo

```
void multstore
    (long x, long y, long *dest)
{
    long t = mult2(x, y);
    *dest = t;
}
```

0000000000400540 <multstore>:

| | | |
|----------------|----------------------|------------------|
| 400540: | push %rbx | # preservar %rbx |
| 400541: | mov %rdx,%rbx | # conservar dest |
| 400544: | callq 400550 <mult2> | # mult2(x,y) |
| 400549: | mov %rax,(%rbx) | # salvar en dest |
| 40054c: | pop %rbx | # restaurar %rbx |
| 40054d: | retq | # retornar |

```
long mult2
    (long a, long b)
{
    long s = a * b;
    return s;
}
```

0000000000400550 <mult2>:

| | | |
|---------|----------------|------------|
| 400550: | mov %rdi,%rax | # a |
| 400553: | imul %rsi,%rax | # a * b |
| 400557: | retq | # retornar |

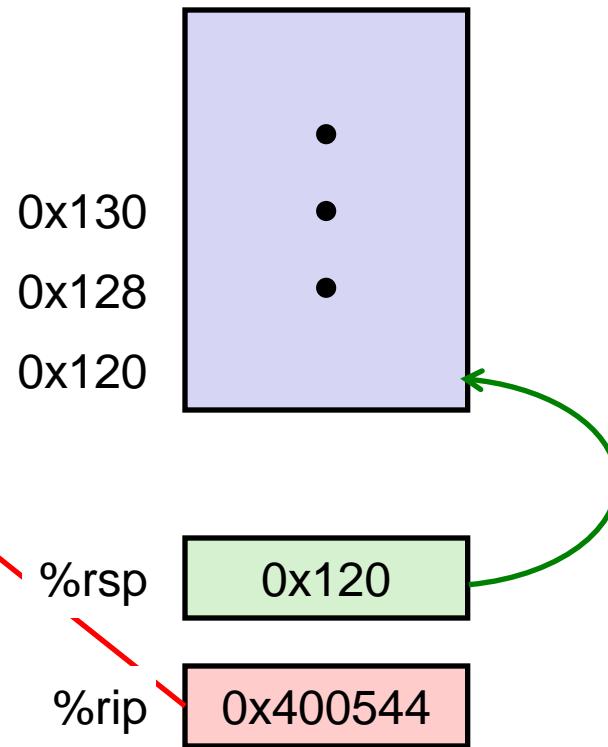
Flujo de Control en Procedimientos

- Usar la pila para soportar llamadas y retornos de procedimientos
- Llamada a procedimiento: **call label**
 - Recuerda[†] la dirección de retorno en la pila
 - Salta a etiqueta **label**
 - Codificada con *direcccionamiento relativo a IP*
- Dirección de retorno:
 - Dirección de la siguiente instrucción justo después de la llamada (call)
 - Ejemplo en el desensamblado anterior: **0x400549**
- Retorno de procedimiento: **ret**
 - Recupera[†] la dirección (de retorno) de la pila
 - Salta a dicha dirección

Ejemplo Flujo Control #1

```
0000000000400540 <multstore>:  
    .  
    .  
    .  
    400544: callq  400550 <mult2>  
400549: mov     %rax,(%rbx)  
    .  
    .
```

```
0000000000400550 <mult2>:  
    400550: mov     %rdi,%rax  
    .  
    .  
    400557: retq
```

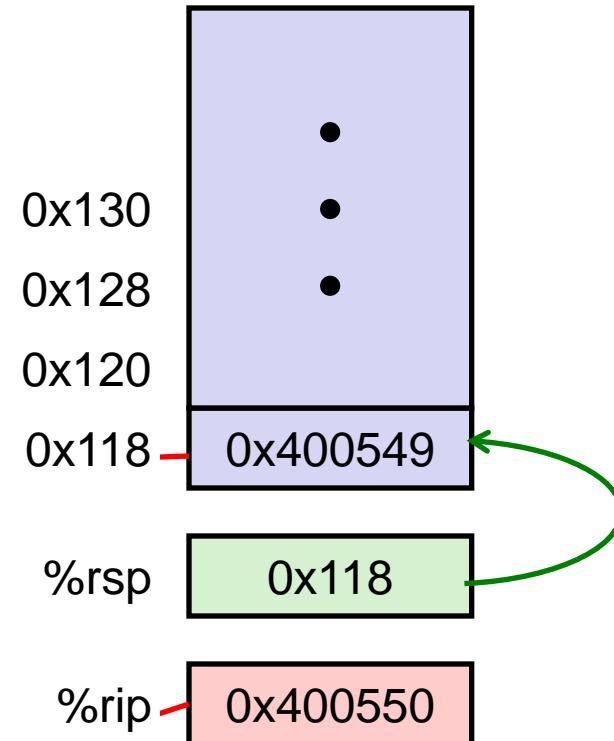


■ Direcccionamiento relativo a contador de programa (RIP)

$$\begin{array}{rcl} \text{RIP} & \textcolor{red}{0x00400549} & \text{(tras fetch)} \\ +\text{offs} & \underline{0x00000007} \\ =\text{Dst} & \textcolor{black}{0x00400550} \end{array}$$

Ejemplo Flujo Control #2

```
0000000000400540 <multstore>:
.
.
400544: callq  400550 <mult2>
400549: mov     %rax,(%rbx) ←
```

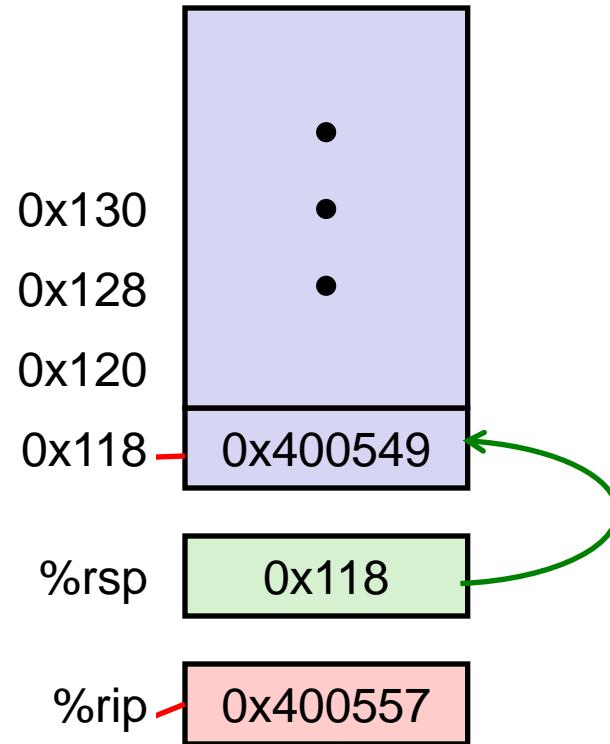


```
0000000000400550 <mult2>:
400550: mov     %rdi,%rax ←
.
.
400557: retq
```

| | |
|---------------------------|----------------------|
| 400544: e8 07 00 00 00 | callq 400550 <mult2> |
| 400549: 48 89 03 | mov %rax,(%rbx) |
| 40054c: 5b | pop %rbx |
| 40054d: c3 | retq |
| 40054e: 66 90 | nop |
| 0000000000400550 <mult2>: | |
| 400550: 48 89 f8 | mov %rdi,%rax |

Ejemplo Flujo Control #3

```
0000000000400540 <multstore>:  
    .  
    .  
400544: callq  400550 <mult2>  
400549: mov     %rax,(%rbx) ←
```

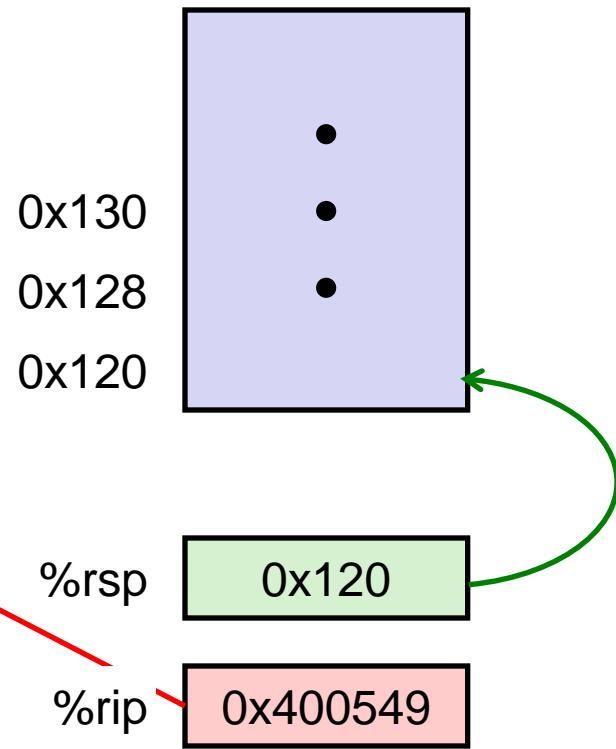


```
0000000000400550 <mult2>:  
400550: mov     %rdi,%rax  
    .  
    .  
400557: retq ←
```

Ejemplo Flujo Control #4

```
0000000000400540 <multstore>:  
    .  
    .  
    .  
    400544: callq  400550 <mult2>  
    400549: mov     %rax,(%rbx) ←
```

```
0000000000400550 <mult2>:  
    400550: mov     %rdi,%rax  
    .  
    .  
    400557: retq
```



Programación Máquina III: Procedimientos

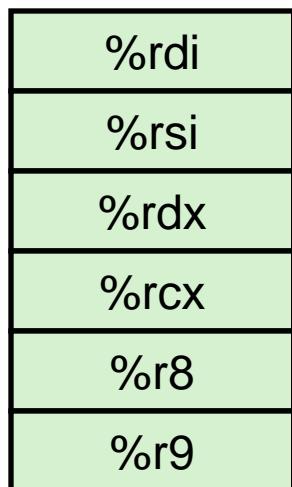
■ Procedimientos

- Mecanismos
- Estructura de la pila
- Convenciones de llamada
 - Pasando el control
 - **Pasando los datos**
 - Gestionando datos locales
- Ejemplos ilustrativos de Recursividad

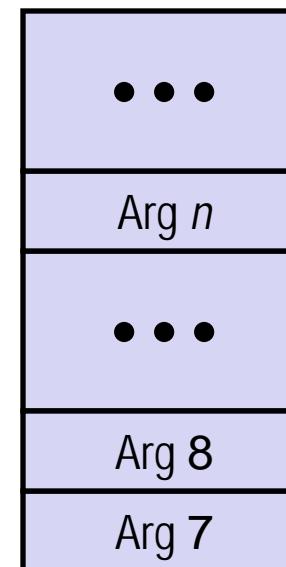
Flujo de Datos para Procedimientos

Registros

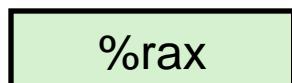
- **Primeros 6 argumentos**



Pila



- **Valor de retorno**



- **Sólo se reserva espacio en la pila cuando se necesita**

Ejemplo

Flujo Datos

```
void multstore
    (long x, long y, long *dest)
{
    long t = mult2(x, y);
    *dest = t;
}
```

0000000000400540 <multstore>:

x en %rdi, y en %rsi, dest en %rdx

```
400540: push    %rbx          # preservar %rbx
400541: mov     %rdx,%rbx      # conservar dest
400544: callq   400550 <mult2>  # mult2(x,y)
400549: mov     %rax,(%rbx)    # salvar en dest
40054c: pop     %rbx          # restaurar %rbx
# %rax libre (void), todavía conserva t=x*y
40054d: retq               # retornar
```

```
long mult2
    (long a, long b)
{
    long s = a * b;
    return s;
}
```

0000000000400550 <mult2>:

a en %rdi, b en %rsi

```
400550: mov     %rdi,%rax      # a
400553: imul   %rsi,%rax      # a * b
# s en %rax
400557: retq               # retornar
```

Programación Máquina III: Procedimientos

■ Procedimientos

- Mecanismos
- Estructura de la pila
- Convenciones de llamada
 - Pasando el control
 - Pasando los datos
 - **Gestionando datos locales**
- Ejemplos ilustrativos de Recursividad

Lenguajes basados en pila[†]

■ Lenguajes que soportan recursividad

- P.ej., C, Pascal, Java
- El código debe ser “*Reentrant*”
 - Múltiples instanciaciones[‡] simultáneas de un mismo procedimiento
- Se necesita algún lugar para guardar el estado de cada instanciaión
 - Argumentos
 - Variables locales
 - Puntero (dirección) de retorno

■ Disciplina de pila

- Estado para un procedimiento dado, necesario por tiempo limitado
 - Desde que se le llama hasta que retorna
- El invocado[‡] retorna antes de que lo haga el invocante[‡]

■ La pila se reserva en *Marcos*[‡]

- estado para una sola instanciaión de procedimiento

[†] “block structured” en terminología Intel

[‡] “callee/caller” en inglés

[#] “allocated in frames”

[‡] “instantiate” = crear nuevos ejemplares

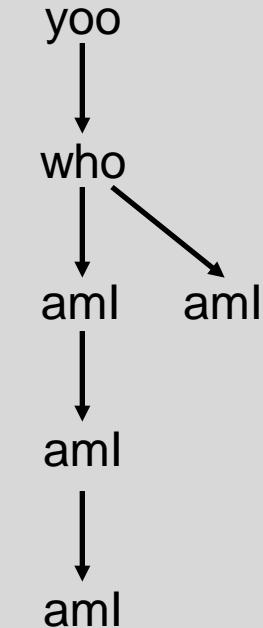
Ejemplo de secuencia[†] de llamadas

```
yoo( ... )
{
    .
    .
    who( ) ;
    .
    .
}
```

```
who( ... )
{
    .
    .
    amI( ) ;
    .
    .
    amI( ) ;
    .
    .
}
```

```
amI( ... )
{
    .
    .
    amI( ) ;
    .
    .
}
```

Ejemplo
Sec. Llamadas



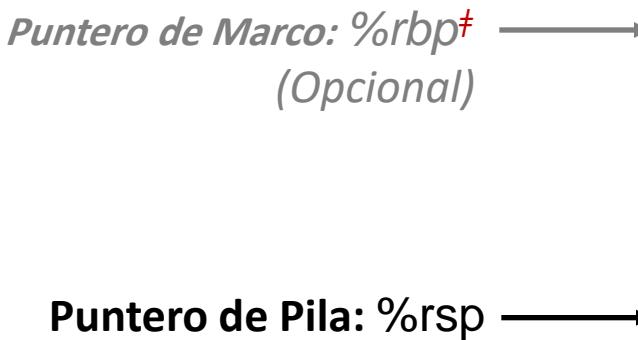
El procedimiento `amI()` es recursivo

[†] “call chain”.
“yoo/you” = tú,
“who” = quién,
“am I” = soy yo. 24

Marcos de Pila

■ Contenido

- Información de retorno
- Almacnmto[†] local (si necesario)
- Espacio temporal (si necesario)



■ Gestión

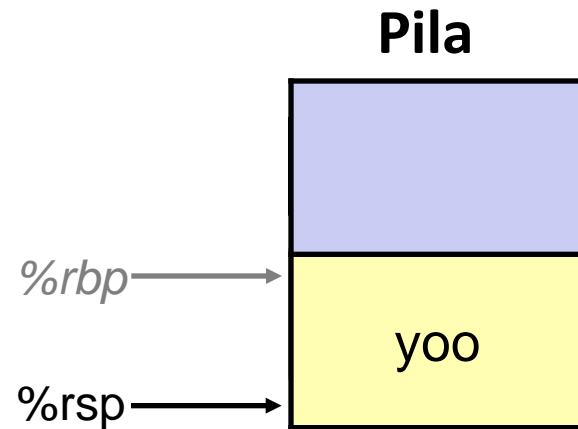
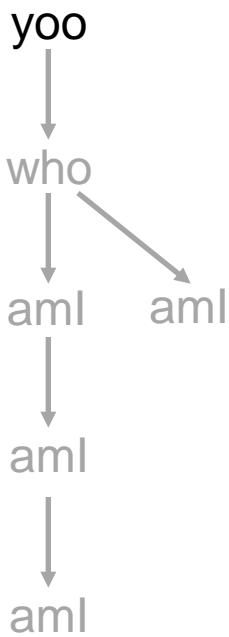
- Espacio se reserva al entrar el procedimiento
 - Código de “Inicialización” [‡]
 - Incluye el “push dir.ret.” de la instrucción **call**
- Se libera al retornar
 - Código de “Finalización” [‡]
 - Incluye el “pop cont.prog.” de la instrucción **ret**

“Tope” Pila

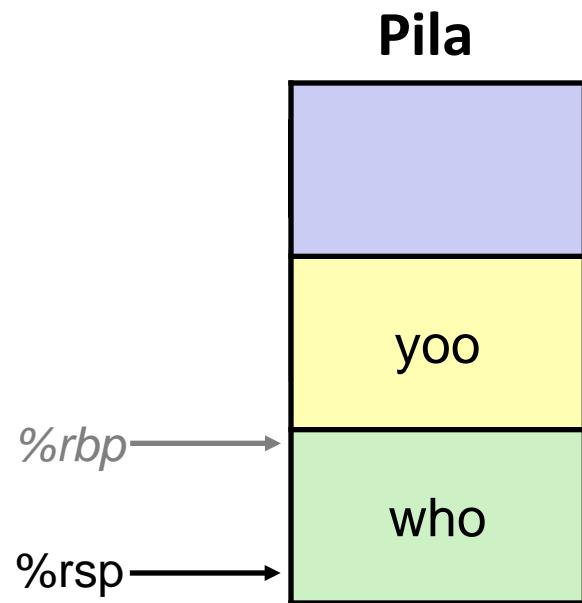
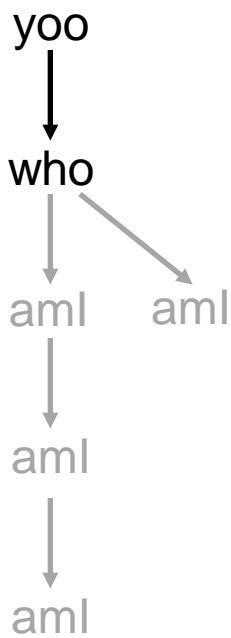
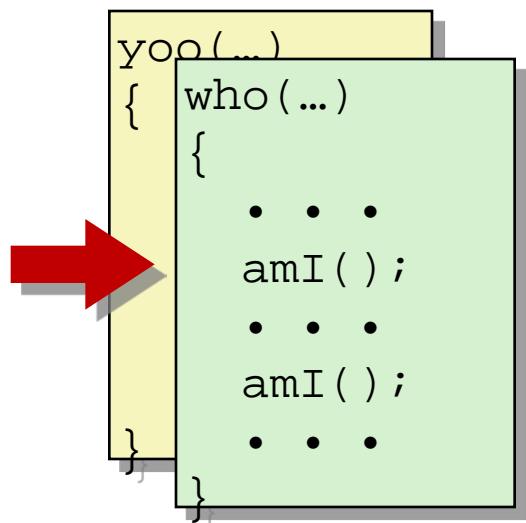
[#] si se usa *-fno-omit-frame-pointer*
[‡] “set-up/finish code”
† “local storage” 25

Ejemplo

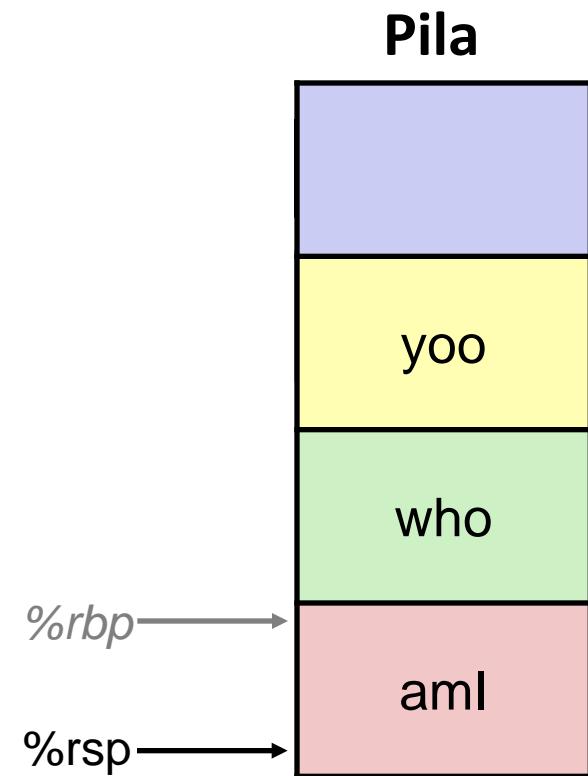
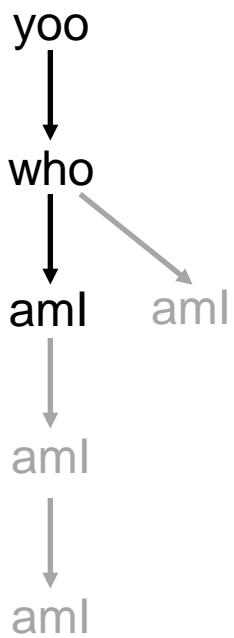
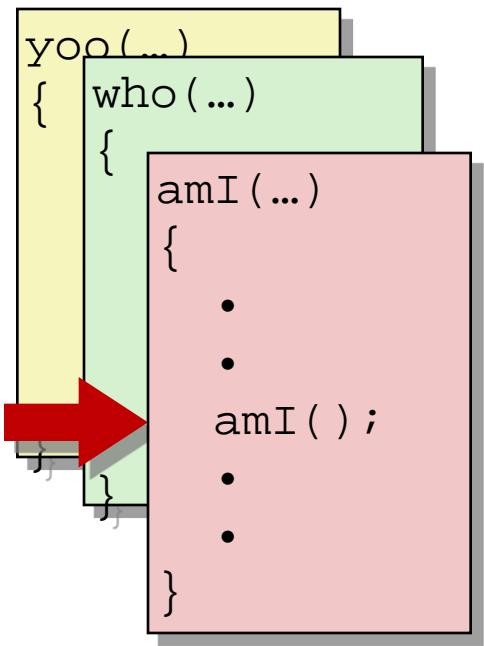
```
yoo( ... )  
{  
    •  
    •  
    who( ) ;  
    •  
    •  
}
```



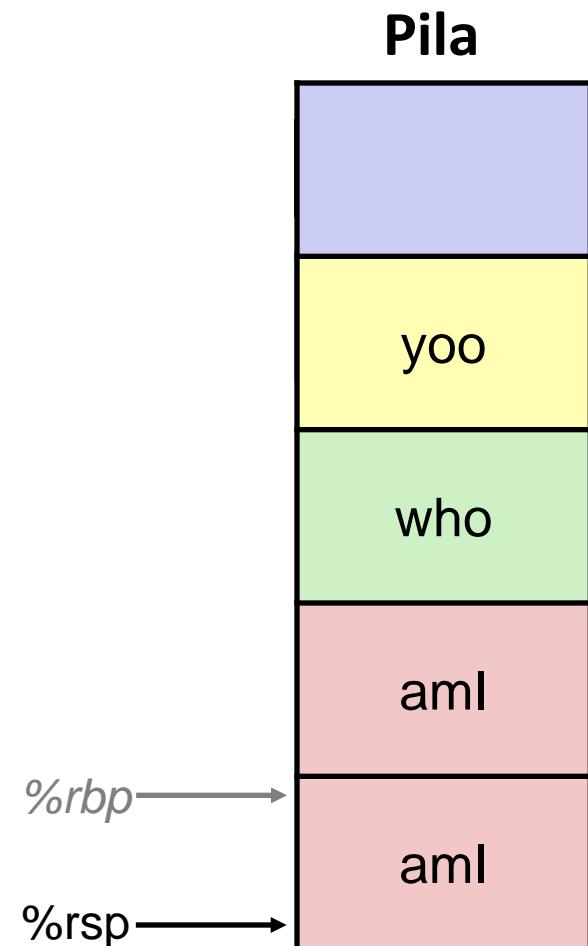
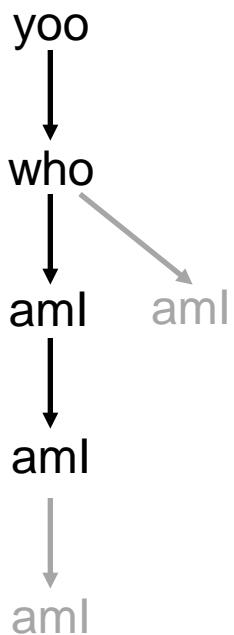
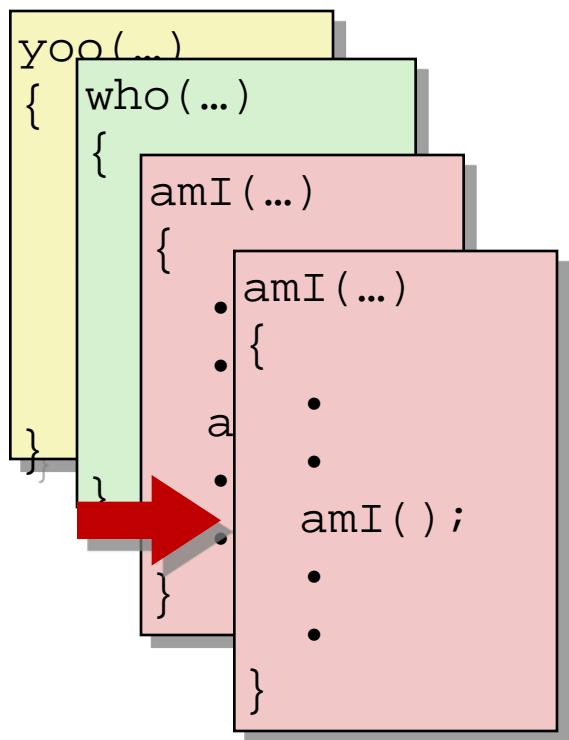
Ejemplo



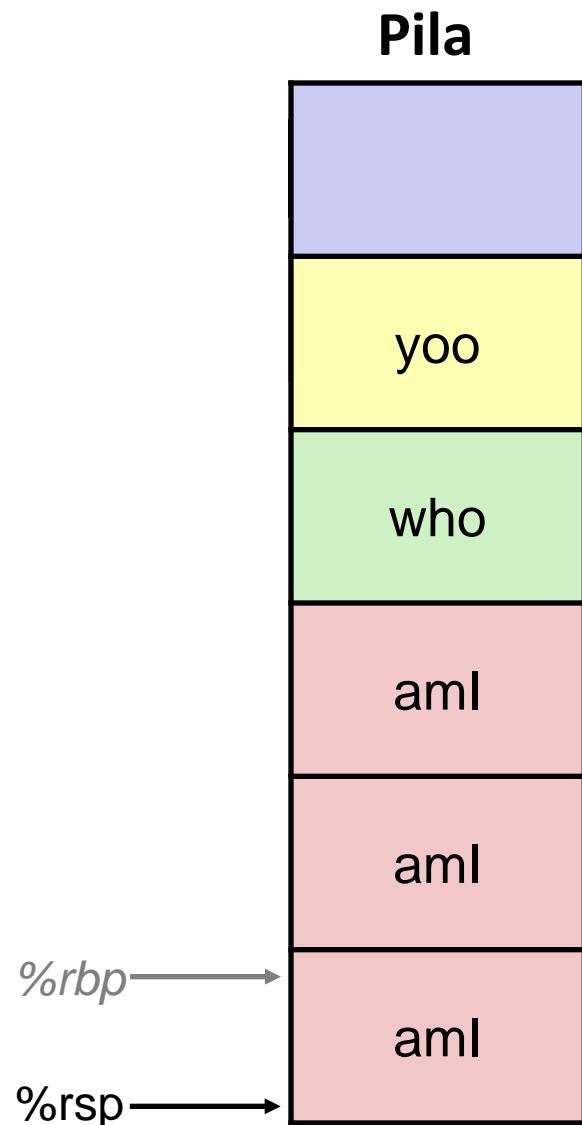
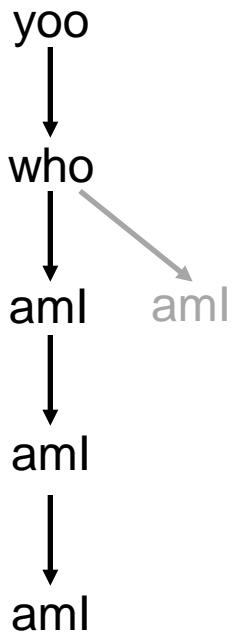
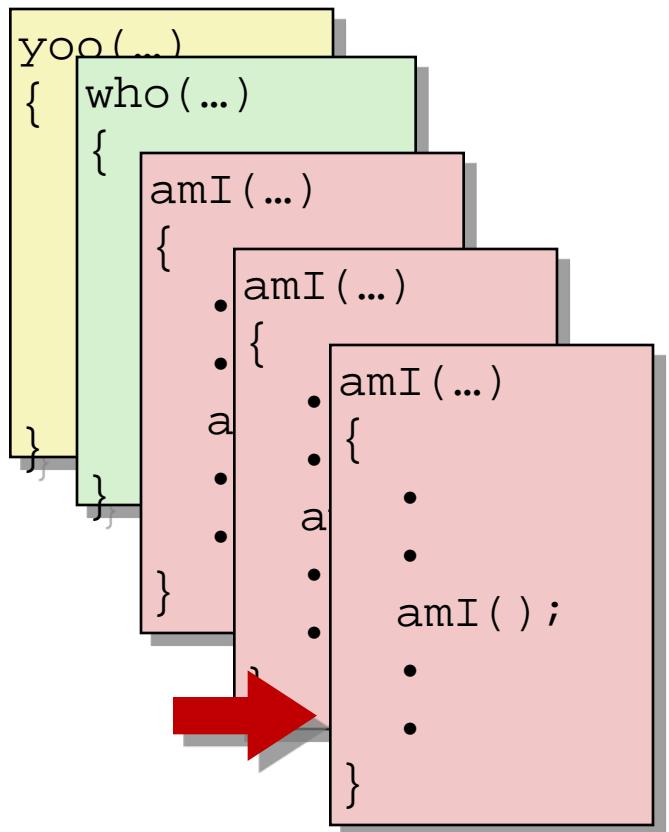
Ejemplo



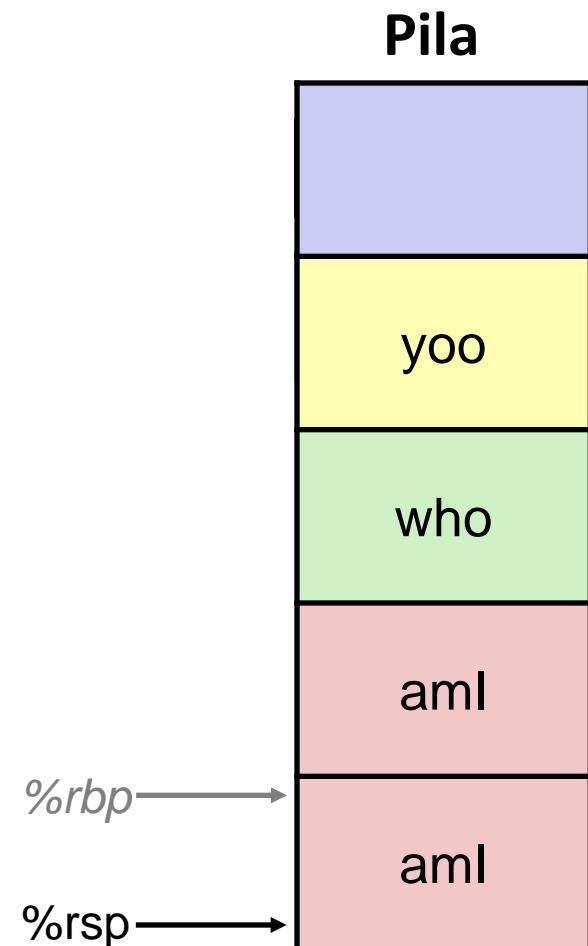
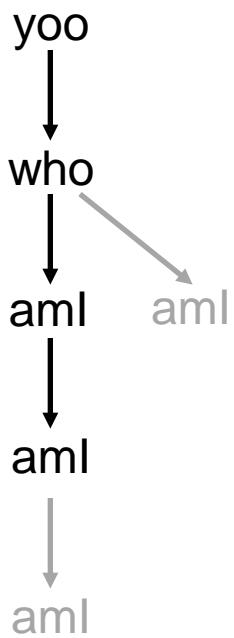
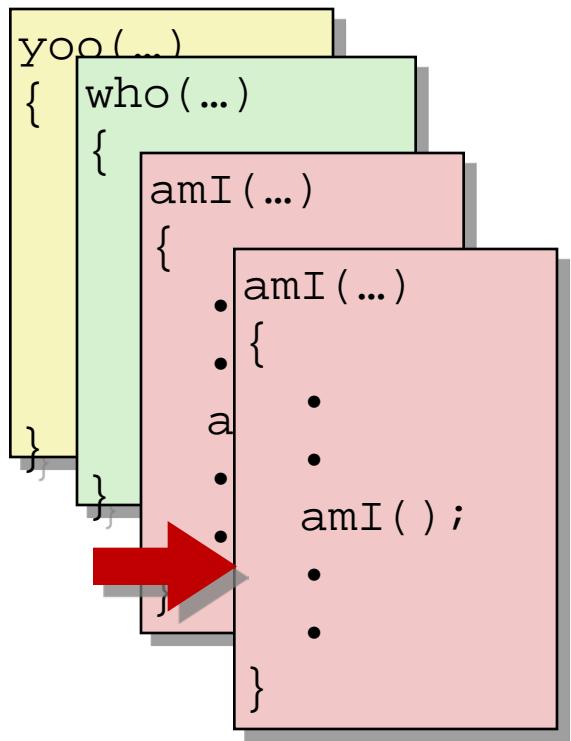
Ejemplo



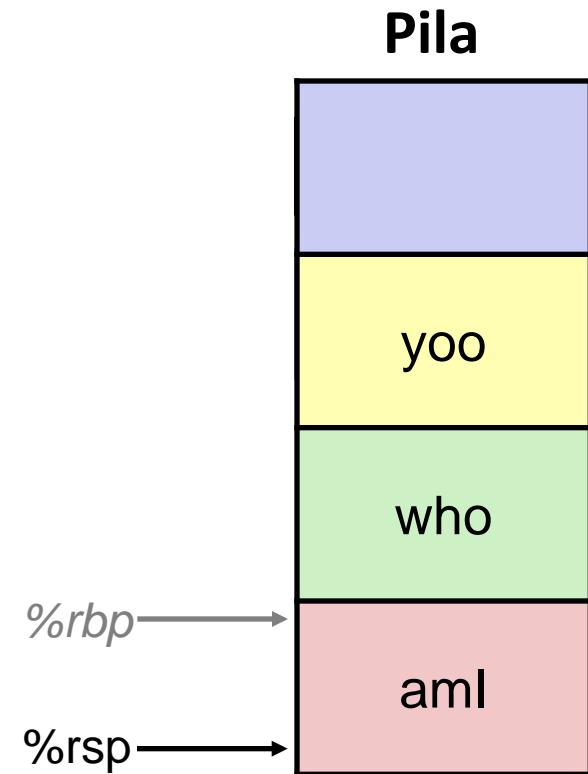
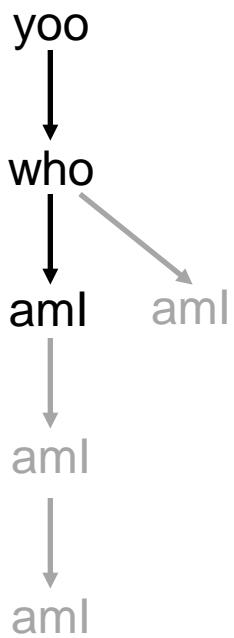
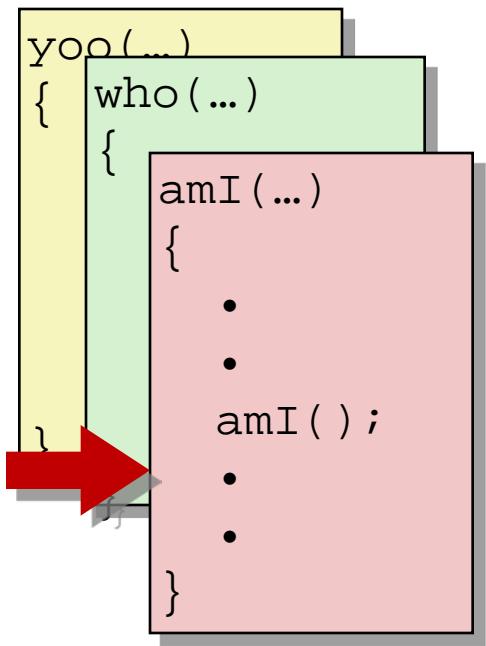
Ejemplo



Ejemplo

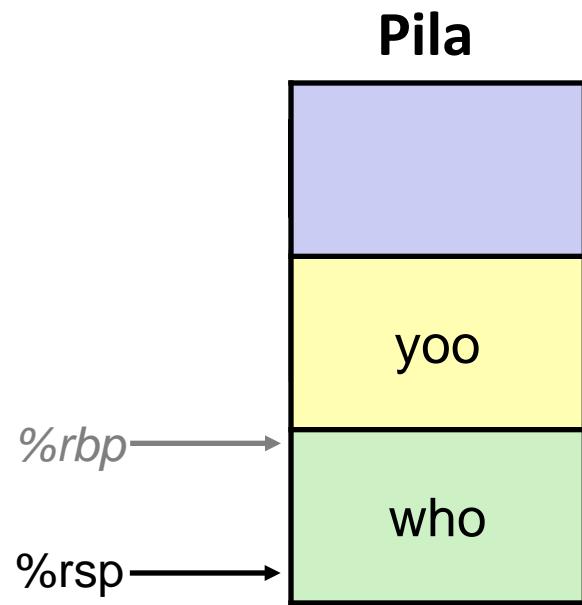
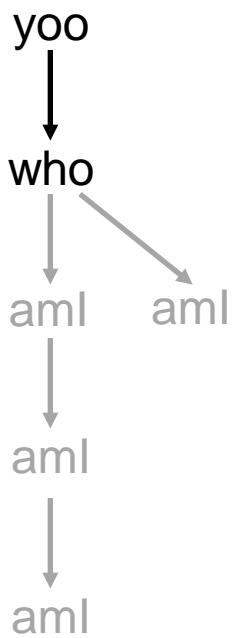


Ejemplo

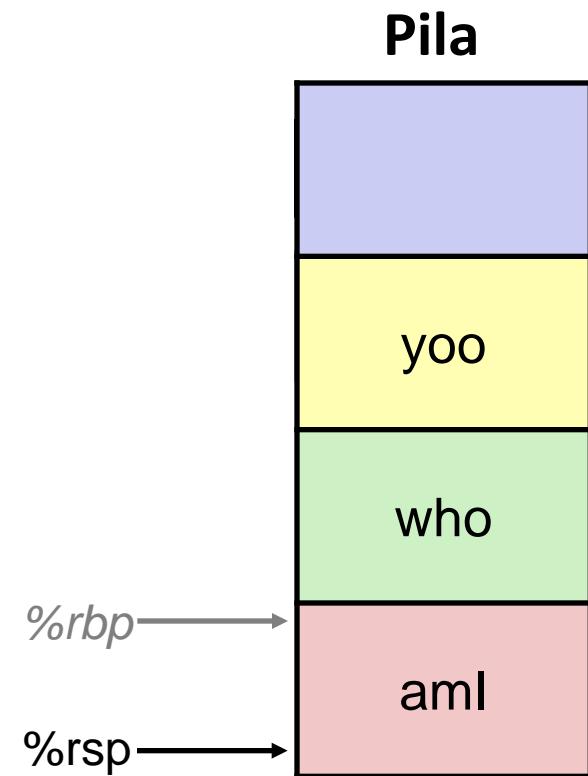
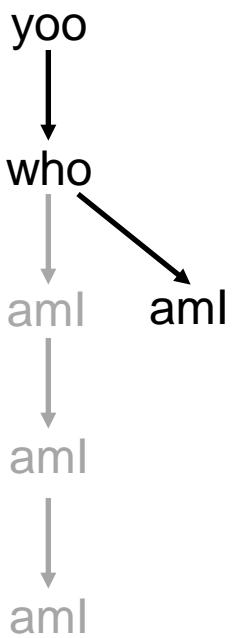
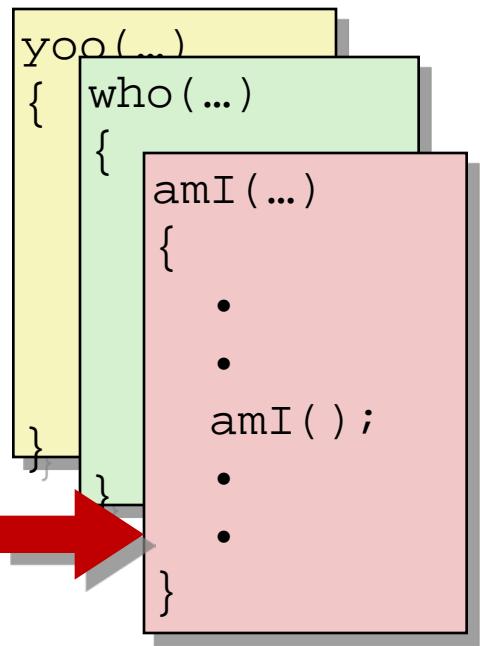


Ejemplo

```
yoo(...)  
{   who(...)  
{  
    . . .  
    amI();  
    . . .  
    amI();  
    . . .  
}
```

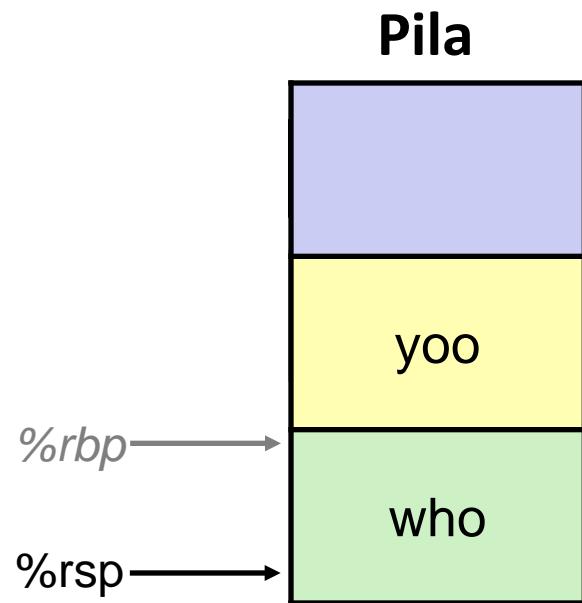
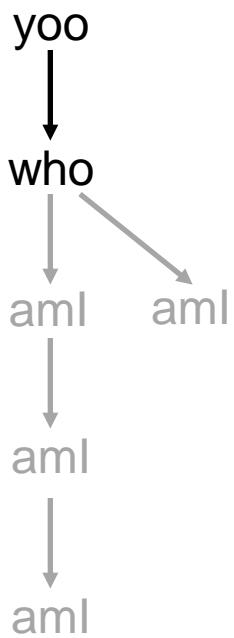


Ejemplo



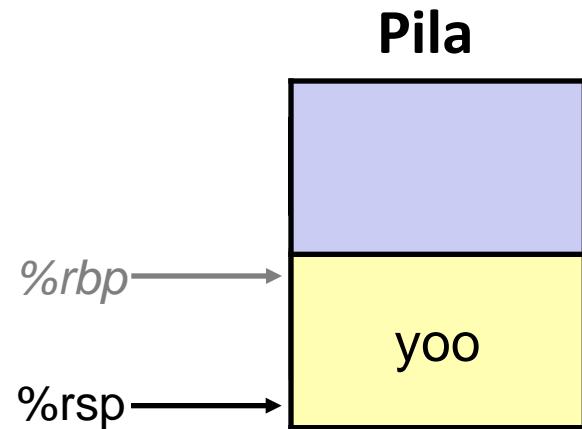
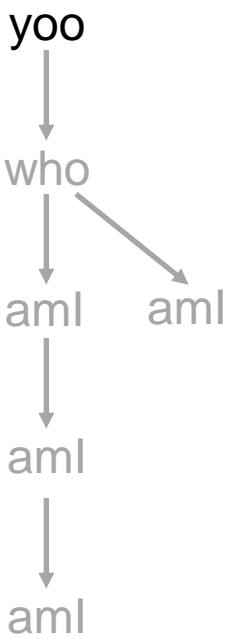
Ejemplo

```
yoo(...)  
{   who(...)  
{  
    . . .  
    amI();  
    . . .  
    amI();  
    . . .  
}
```



Ejemplo

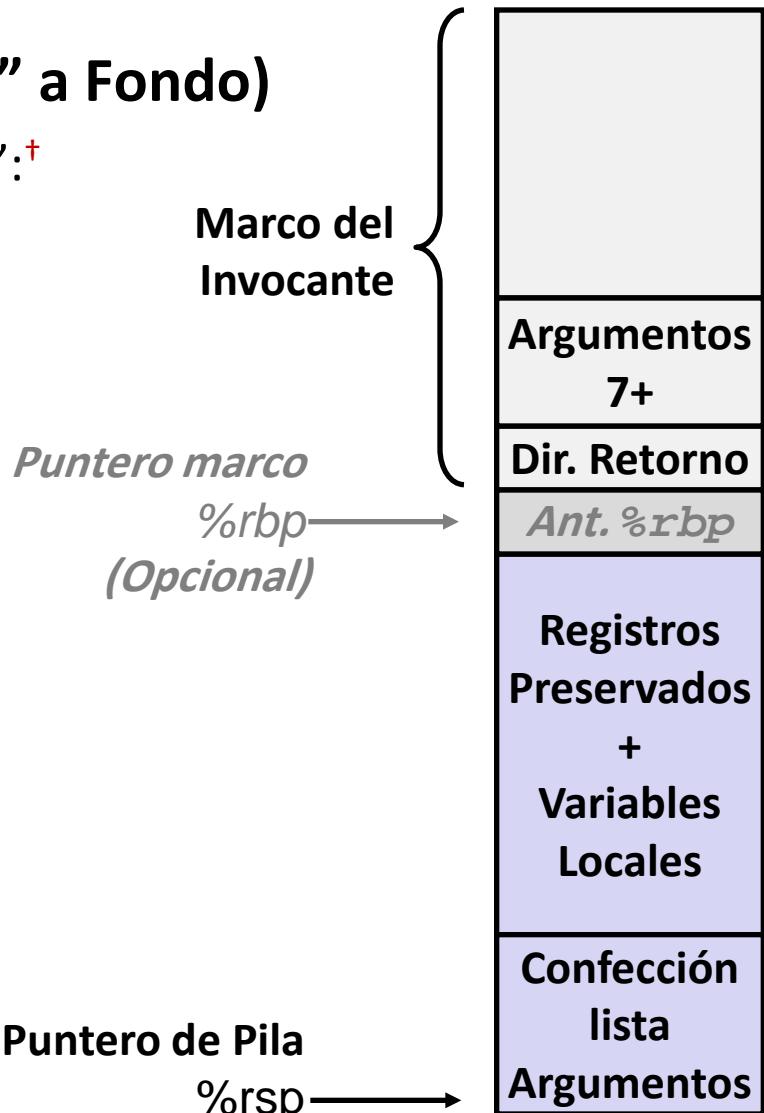
```
yoo( ... )  
{  
    •  
    •  
    who( ) ;  
    •  
    •  
}
```



Marco de Pila x86-64/Linux

■ Contenidos Marco Pila (de “Tope” a Fondo)

- “Confección de la lista de argumentos”:[†]
Parámetros (7+) función punto ser llamada
- Variables locales
Si no se pueden mantener en registros
- Contexto registros preservados
- *Antiguo puntero de marco (opcional)*
usando -fno-omit-frame-pointer (ó -O0)
 - *Dir.retorno pertenece marco anterior*



■ Marco de Pila del Invocante

- Dirección de retorno
 - Salvada por la instrucción call
- Argumentos (7+) para esta llamada

Puntero de Pila
%rsp →

[†] “Argument build” 37

Ejemplo: incr

```
long incr(long *p, long val) {  
    long x = *p;  
    long y = x + val;  
    *p = y;  
    return x;  
}
```

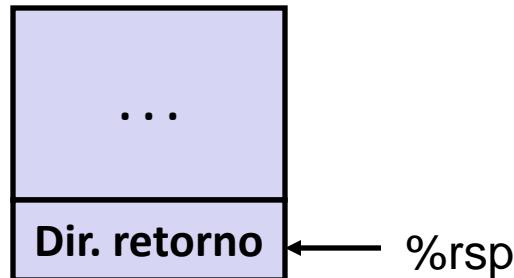
```
incr:  
    movq    (%rdi), %rax  
    addq    %rax, %rsi  
    movq    %rsi, (%rdi)  
    ret
```

| Registro | Uso(s) |
|----------|-----------------------------|
| %rdi | Argumento p |
| %rsi | Argumento val , y |
| %rax | x , Valor de retorno |

Ejemplo: llamar a incr #1

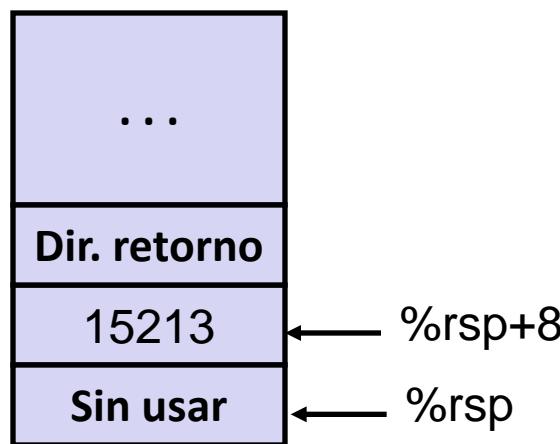
```
long call_incr() {
    long v1 = 15213;
    long v2 = incr(&v1, 3000);
    return v1+v2;
}
```

Estructura de Pila inicial



```
call_incr:
    subq    $16, %rsp
    movq    $15213, 8(%rsp)
    movl    $3000, %esi
    leaq    8(%rsp), %rdi
    call    incr
    addq    8(%rsp), %rax
    addq    $16, %rsp
    ret
```

Estructura de Pila resultante

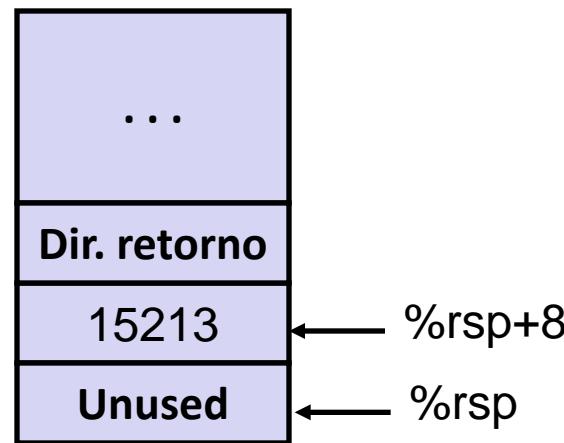


Ejemplo: llamar a incr #2

```
long call_incr() {
    long v1 = 15213;
    long v2 = incr(&v1, 3000);
    return v1+v2;
}
```

```
call_incr:
    subq    $16, %rsp
    movq    $15213, 8(%rsp)
    movl    $3000, %esi
    leaq    8(%rsp), %rdi
    call    incr
    addq    8(%rsp), %rax
    addq    $16, %rsp
    ret
```

Estructura de Pila



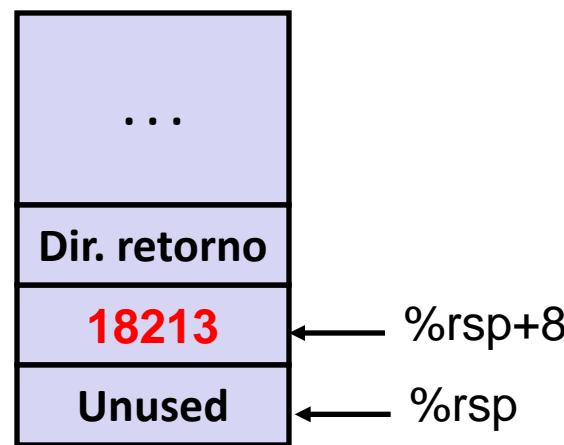
| Registro | Uso(s) |
|-------------------|----------------------|
| <code>%rdi</code> | <code>&v1</code> |
| <code>%rsi</code> | 3000 |

Ejemplo: llamar a incr #3

```
long call_incr() {
    long v1 = 15213;
    long v2 = incr(&v1, 3000);
    return v1+v2;
}
```

```
call_incr:
    subq    $16, %rsp
    movq    $15213, 8(%rsp)
    movl    $3000, %esi
    leaq    8(%rsp), %rdi
    call    incr
    addq    8(%rsp), %rax
    addq    $16, %rsp
    ret
```

Estructura de Pila

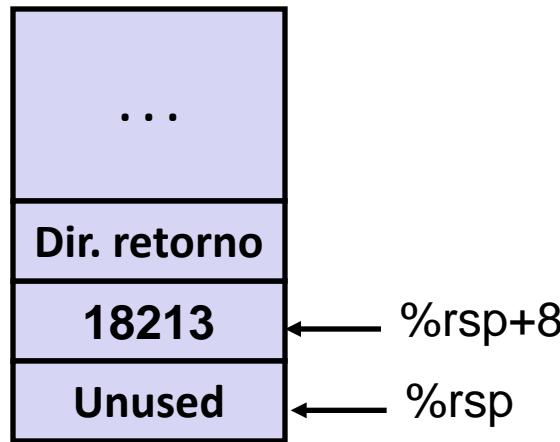


| Registro | Uso(s) |
|----------|--------|
| %rdi | &v1 |
| %rsi | 3000 |

Ejemplo: llamar a incr #4

Estructura de Pila

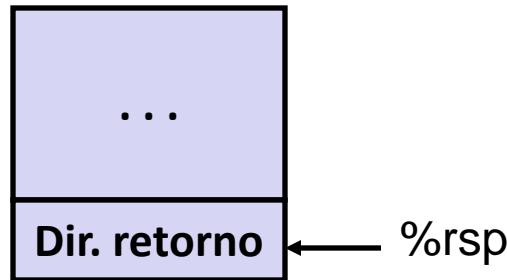
```
long call_incr() {
    long v1 = 15213;
    long v2 = incr(&v1, 3000);
    return v1+v2;
}
```



```
call_incr:
    subq    $16, %rsp
    movq    $15213, 8(%rsp)
    movl    $3000, %esi
    leaq    8(%rsp), %rdi
    call    incr
    addq    8(%rsp), %rax
    addq    $16, %rsp
    ret
```

| Registro | Uso(s) |
|----------|------------------|
| %rax | Valor de retorno |

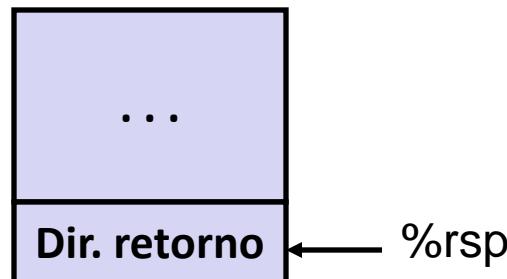
Estructura de Pila resultante



Ejemplo: llamar a incr #5

```
long call_incr() {
    long v1 = 15213;
    long v2 = incr(&v1, 3000);
    return v1+v2;
}
```

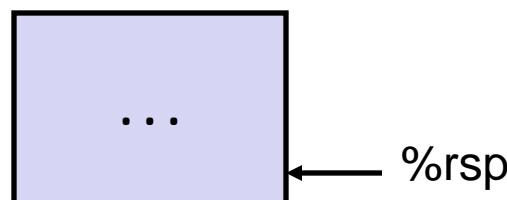
Estructura de Pila resultante



```
call_incr:
subq    $16, %rsp
movq    $15213, 8(%rsp)
movl    $3000, %esi
leaq    8(%rsp), %rdi
call    incr
addq    8(%rsp), %rax
addq    $16, %rsp
ret
```

| Registro | Uso(s) |
|----------|------------------|
| %rax | Valor de retorno |

Estructura de Pila final



Convenciones de Preservación de Registros[†]

- Cuando el procedimiento yoo llama a who:
 - yoo es el **que llama (invocante, llamante)**
 - who es el **llamado (invocado)**
- ¿Se puede usar un registro para almacenamiento temporal?

yoo:

```
• • •  
    movq $15213, %rdx  
    call who  
    addq %rdx, %rax  
• • •  
    ret
```

who:

```
• • •  
    subq $18213, %rdx  
• • •  
    ret
```

- Contenidos del registro %edx sobrescritos por who
- Podría causar problemas → ¡debería hacerse algo!
 - Necesita alguna coordinación

Convenciones de Preservación de Registros

- Cuando el procedimiento *yoo* llama a *who*:
 - *yoo* es el *que llama (invocante, llamante)*
 - *who* es el *llamado (invocado)*
- ¿Se puede usar un registro para almacenamiento temporal?
- Convenciones[†]
 - “*Salva Invocante*”
 - El que llama salva valores temporales en su marco antes de la llamada
 - “*Salva Invocado*”
 - El llamado salva valores temporales en su marco antes de usar (regs.)
 - ...y los restaura antes de retornar al que llama

Uso de Registros en Linux x86-64[†] #1

■ %rax

- Valor de retorno
- También salva-invocante
- *Puede ser modificado[‡]* por el proc.

■ %rdi, ..., %r9

- Argumentos[‡]
- También salva-invocante
- Pueden ser modificados[‡] por proc.

■ %r10, %r11

- Salva-invocante
- Pueden ser modificados[‡] por proc.

Val.retorno

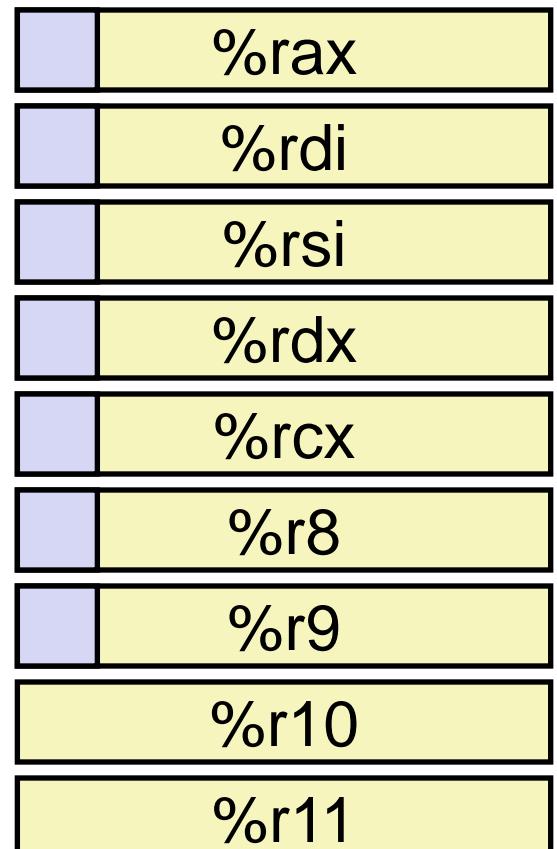
S-Invocante

Argumentos

S-Invocante

Temporales

S-Invocante



[†] Ver Wikipedia: "X86 calling conventions", Sys5 AMD64 ABI

[‡] regla mnemotécnica: invocado-cuidado, invocante-adelante

[‡] regla mnemotécnica: Diane's Silk Dress Costs \$89

Uso de Registros en Linux x86-64 #2

■ **%rbx, %r12 - %r15**

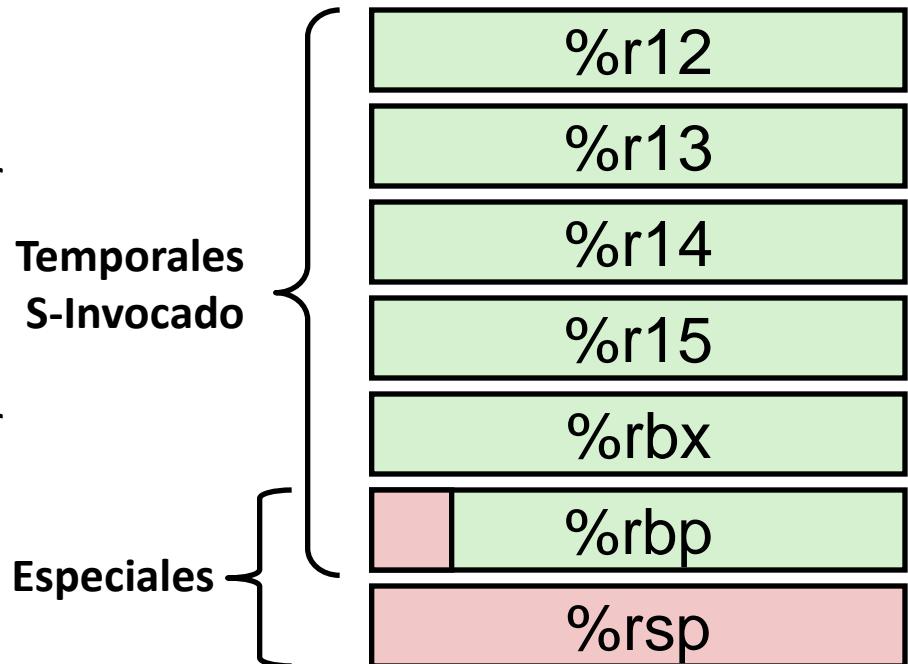
- Salva-invocado
- Invocado debe preservar y restaurar

■ **%rbp**

- Salva-invocado
- Invocado debe preservar y restaurar
- Puede que se use como marco pila
- Puede usarse intermezcladamente[†]

■ **%rsp**

- Forma especial de salva-invocado
- Restaurado a su valor original a la salida del procedimiento



[†] "mix & match", se refiere a que podrían linkarse procedimientos compilados con `gcc -fno-omit...` y `-fomit...` y no pasaría nada porque `%rbp` seguiría siendo s-invocado

Mini-Ejercicio

```

long add5(long b0, long b1, long b2, long b3, long b4) {
    return b0+b1+b2+b3+b4;
}

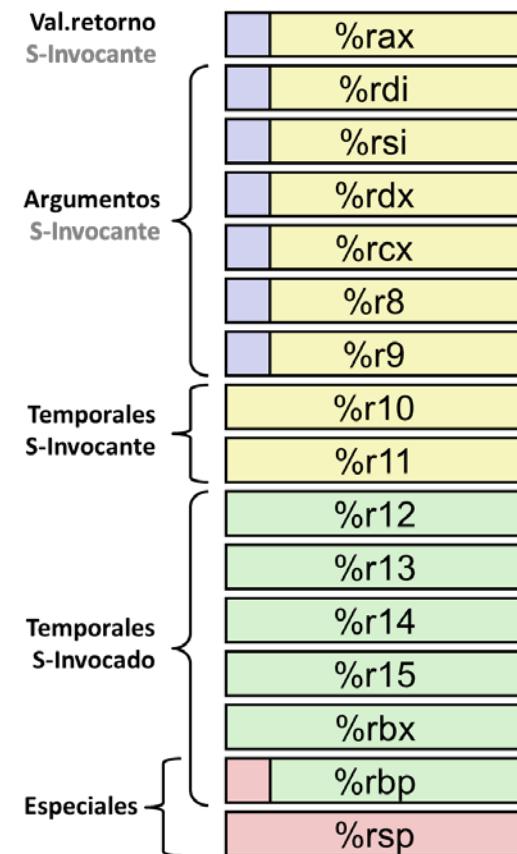
long add10(long a0, long a1, long a2, long a3, long a4,
           long a5, long a6, long a7, long a8, long a9) {
    return add5(a0, a1, a2, a3, a4) +
           add5(a5, a6, a7, a8, a9);
}

```

■ ¿Dónde se pasan **a0,..., a9?**
rdi, rsi, rdx, rcx, r8, r9, pila

■ ¿Dónde se pasan **b0,..., b4?**
rdi, rsi, rdx, rcx, r8

■ ¿Qué registros tenemos que preservar?
 Pregunta mal formulada. Requiere ver ASM.
rbx, rbp, r9 (durante 1ª llamada a **add5**)



Mini-Ejercicio

```
long add5(long b0, long b1, long b2, long b3, long b4) {
    return b0+b1+b2+b3+b4;
}

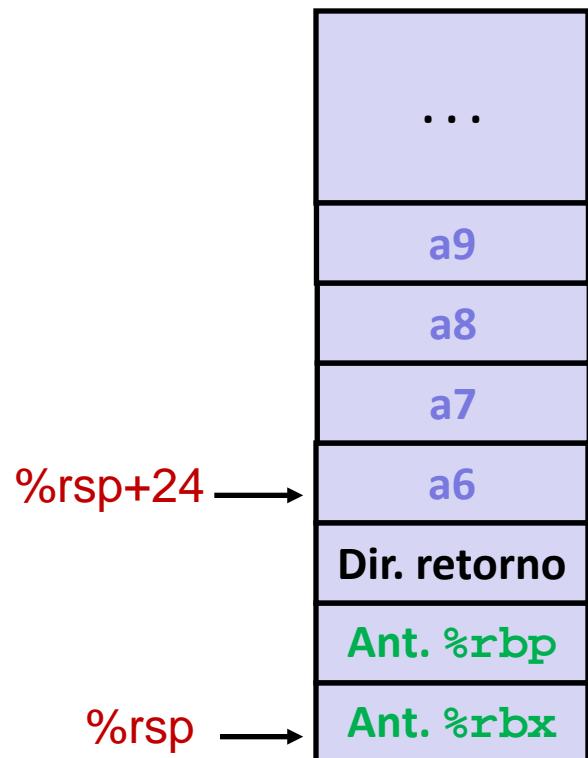
long add10(long a0, long a1, long a2, long a3, long a4,
           long a5, long a6, long a7, long a8, long a9) {
    return add5(a0, a1, a2, a3, a4)+  

           add5(a5, a6, a7, a8, a9);
}
```

```
add10:
    pushq  %rbp
    pushq  %rbx
    movq   %r9, %rbp
    call   add5
    movq   %rax, %rbx
    movq   48(%rsp), %r8
    movq   40(%rsp), %rcx
    movq   32(%rsp), %rdx
    movq   24(%rsp), %rsi
    movq   %rbp, %rdi
    call   add5
    addq   %rbx, %rax
    popq   %rbx
    popq   %rbp
    ret
```

```
add5:
    addq   %rsi, %rdi
    addq   %rdi, %rdx
    addq   %rdx, %rcx
    leaq   (%rcx,%r8), %rax
    ret
```

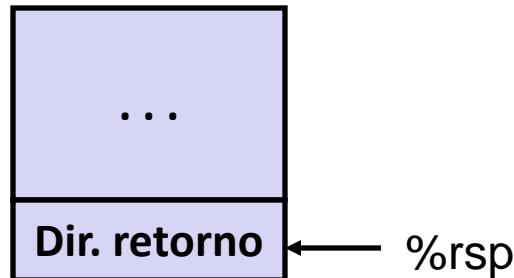
Pila durante la llamada a add10



Ejemplo Salva-invocado #1

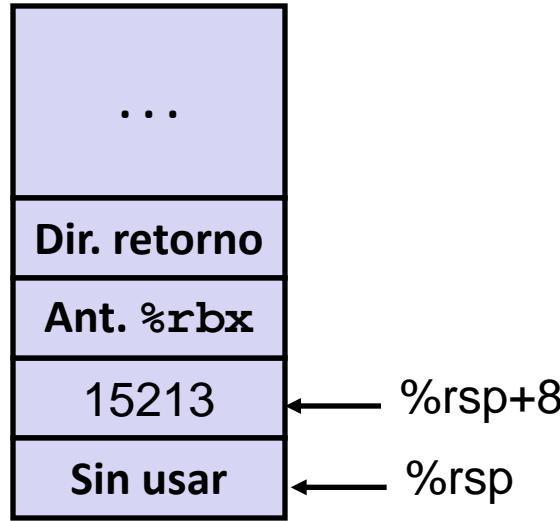
```
long call_incr2(long x) {
    long v1 = 15213;
    long v2 = incr(&v1, 3000);
    return x+v2;
}
```

Estructura de Pila inicial



```
call_incr2:
    pushq  %rbx
    subq   $16, %rsp
    movq   %rdi, %rbx
    movq   $15213, 8(%rsp)
    movl   $3000, %esi
    leaq   8(%rsp), %rdi
    call   incr
    addq   %rbx, %rax
    addq   $16, %rsp
    popq   %rbx
    ret
```

Estructura de Pila resultante

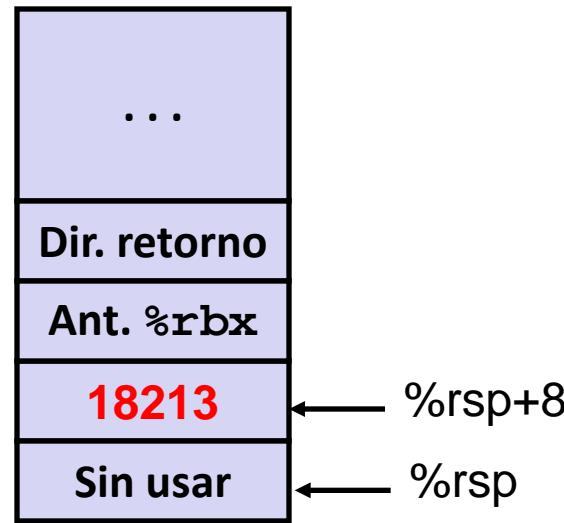


Ejemplo Salva-invocado #2

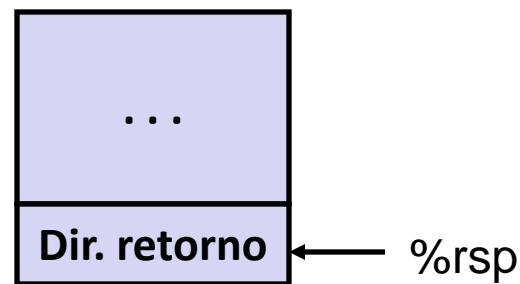
Estructura de Pila

```
long call_incr2(long x) {
    long v1 = 15213;
    long v2 = incr(&v1, 3000);
    return x+v2;
}
```

```
call_incr2:
    pushq %rbx
    subq $16, %rsp
    movq %rdi, %rbx
    movq $15213, 8(%rsp)
    movl $3000, %esi
    leaq 8(%rsp), %rdi
    call incr
    addq %rbx, %rax
    addq $16, %rsp
    popq %rbx
    ret
```



Estructura de Pila antes de ret



Programación Máquina III: Procedimientos

■ Procedimientos

- Mecanismos
- Estructura de la pila
- Convenciones de llamada
 - Pasando el control
 - Pasando los datos
 - Gestionando datos locales
- Ejemplos ilustrativos de Recursividad

Función Recursiva

```
/* Recursive popcount */
long pcount_r(unsigned long x) {
    if (x == 0)
        return 0;
    else
        return (x & 1)
            + pcount_r(x >> 1);
}
```

pcount_r:

```
    movl    $0, %eax
    testq   %rdi, %rdi
    je      .L6
    pushq   %rbx
    movq   %rdi, %rbx
    andl   $1, %ebx
    shrq   %rdi
    call   pcount_r
    addq   %rbx, %rax
    popq   %rbx
```

.L6:

```
    rep; ret
```

Condición Terminación Función Recursiva

```
/* Recursive popcount */
long pcount_r(unsigned long x) {
    if (x == 0)
        return 0;
    else
        return (x & 1)
            + pcount_r(x >> 1);
}
```

`pcount_r:`

| | |
|--------------------|-------------------------|
| <code>movl</code> | <code>\$0, %eax</code> |
| <code>testq</code> | <code>%rdi, %rdi</code> |
| <code>je</code> | <code>.L6</code> |
| <code>pushq</code> | <code>%rbx</code> |
| <code>movq</code> | <code>%rdi, %rbx</code> |
| <code>andl</code> | <code>\$1, %ebx</code> |
| <code>shrq</code> | <code>%rdi</code> |
| <code>call</code> | <code>pcount_r</code> |
| <code>addq</code> | <code>%rbx, %rax</code> |
| <code>popq</code> | <code>%rbx</code> |

`.L6:`

`rep; ret`

| Registro | Uso(s) | Tipo |
|-------------------|------------------|---|
| <code>%rdi</code> | <code>x</code> | Salva-invocante  |
| <code>%rax</code> | Valor de retorno | Salva-invocante  |

Preservar Registro para Llamada Recursiva

```
/* Recursive popcount */
long pcount_r(unsigned long x) {
    if (x == 0)
        return 0;
    else
        return (x & 1)
            + pcount_r(x >> 1);
}
```

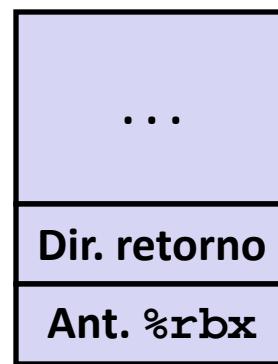
| Registro | Uso(s) | Tipo |
|----------|--------|----------------|
| %rbx | x & 1 | Salva-invocado |

pcount_r:

```
    movl    $0, %eax
    testq   %rdi, %rdi
    je      .L6
    pushq   %rbx
    movq   %rdi, %rbx
    andl   $1, %ebx
    shrq   %rdi
    call   pcount_r
    addq   %rbx, %rax
    popq   %rbx
```

.L6:

rep; ret



Preparar Llamada Función Recursiva

```
/* Recursive popcount */
long pcount_r(unsigned long x) {
    if (x == 0)
        return 0;
    else
        return (x & 1)
            + pcount_r(x >> 1);
}
```

pcount_r:

| | |
|-------|------------|
| movl | \$0, %eax |
| testq | %rdi, %rdi |
| je | .L6 |
| pushq | %rbx |
| movq | %rdi, %rbx |
| andl | \$1, %ebx |
| shrq | %rdi |
| call | pcount_r |
| addq | %rbx, %rax |
| popq | %rbx |

.L6:

rep; ret

| Registro | Uso(s) | Tipo | |
|----------|-----------------------------|-----------------|--|
| %rbx | x & 1 | Salva-invocado | |
| %rdi | x >> 1 Argumento recurs. | Salva-invocante | |

Llamada Función Recursiva

```
/* Recursive popcount */
long pcount_r(unsigned long x) {
    if (x == 0)
        return 0;
    else
        return (x & 1)
            + pcount_r(x >> 1);
}
```

pcount_r:

```
    movl    $0, %eax
    testq   %rdi, %rdi
    je      .L6
    pushq   %rbx
    movq    %rdi, %rbx
    andl    $1, %ebx
    shrq    %rdi
    call    pcount_r
    addq    %rbx, %rax
    popq    %rbx
```

.L6:

rep; ret

| Registro | Uso(s) | Tipo |
|----------|---------------------------------------|---|
| %rbx | x & 1 | Salva-invocado  |
| %rax | Valor de retorno de llamada recursiva | Salva-invocante  |

Resultado Función Recursiva

```
/* Recursive popcount */
long pcount_r(unsigned long x) {
    if (x == 0)
        return 0;
    else
        return (x & 1)
            + pcount_r(x >> 1);
}
```

pcount_r:

```
    movl    $0, %eax
    testq   %rdi, %rdi
    je      .L6
    pushq   %rbx
    movq   %rdi, %rbx
    andl   $1, %ebx
    shrq   %rdi
    call   pcount_r
    addq   %rbx, %rax
    popq   %rbx
```

.L6:

rep; ret

| Registro | Uso(s) | Tipo |
|----------|------------------|---|
| %rbx | x & 1 | Salva-invocado  |
| %rax | Valor de retorno | Salva-invocante  |

Terminar Función Recursiva

```
/* Recursive popcount */
long pcount_r(unsigned long x) {
    if (x == 0)
        return 0;
    else
        return (x & 1)
            + pcount_r(x >> 1);
}
```

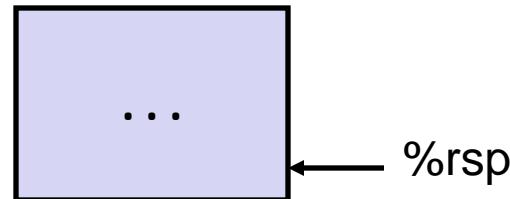
| Registro | Uso(s) | Tipo |
|----------|------------------|---|
| %rax | Valor de retorno | Salva-invocante  |

pcount_r:

```
    movl    $0, %eax
    testq   %rdi, %rdi
    je      .L6
    pushq   %rbx
    movq   %rdi, %rbx
    andl   $1, %ebx
    shrq   %rdi
    call   pcount_r
    addq   %rbx, %rax
    popq   %rbx
```

.L6:

rep; ret



Observaciones Sobre la Recursividad

■ Manejada sin Especiales Consideraciones

- Marcos pila implican que cada llamada a función tiene almacenamiento privado
 - Variables locales y registros preservados
 - Dirección de retorno salvada
- Convenciones preservación registros previenen que una llamada a función corrompa los datos de otra
 - A menos que el código C explícitamente lo haga (p.ej. buffer overflow)
- Disciplina de pila sigue el patrón de llamadas / retornos
 - Si P llama a Q, entonces Q retorna antes que P
 - Primero en entrar, último en salir (Last-In, First-Out)[†]

■ También funciona con recursividad mutua[‡]

- P llama a Q; Q llama a P

[‡] en general con reentrantia

[†] pila = lista LIFO

60

Resumen de Procedimientos en x86-64

■ Puntos Importantes

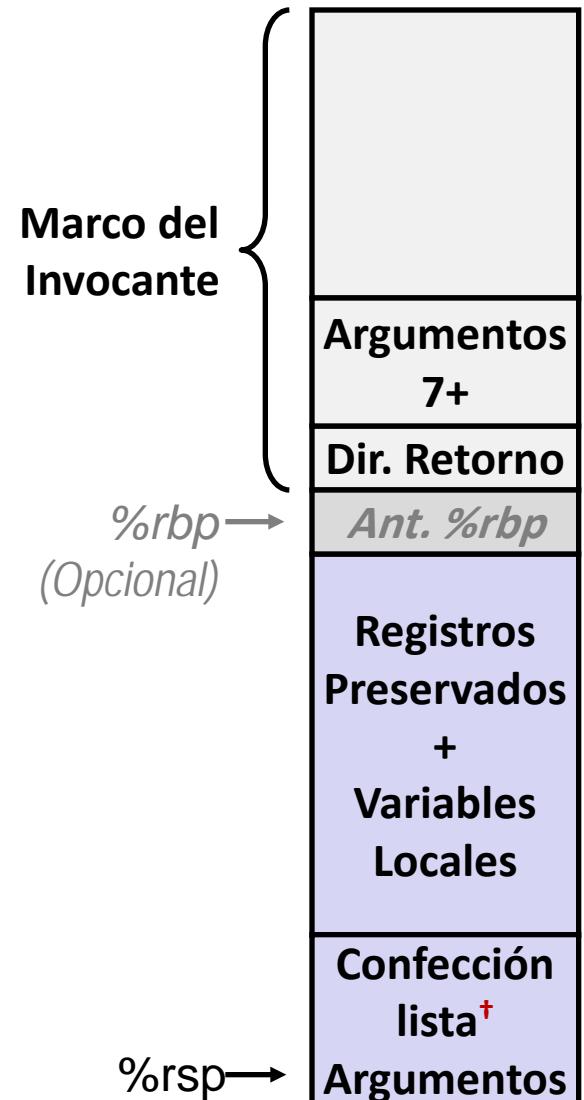
- Pila es la estructura de datos correcta para llamada / retorno procedimientos
 - P llama a Q, entonces Q retorna antes que P

■ Recursividad (y recursividad mutua) con mismas convenciones de llamada normales

- Se pueden almacenar valores tranquilamente en el marco de pila local y en registros salva-invocado
- Poner argumentos 7+ de la función en tope de pila
- Devolver resultado en %rax

■ Punteros son direcciones de valores

- Global o en pila



Guía de trabajo autónomo (4h/s)

■ **Estudio:** del Cap.3 CS:APP (Bryant/O'Hallaron)

- Procedures
 - § 3.7 pp.274-291
- Understanding Pointers, Using GDB.
 - § 3.10.1-2 pp.312-316

■ **Ejercicios:** del Cap.3 CS:APP (Bryant/O'Hallaron)

- Probl. 3.32 § 3.7.2, pp.280
- Probl. 3.33 § 3.7.3, pp.282
- Probl. 3.34 § 3.7.5, pp.288
- Probl. 3.35 § 3.7.6, pp.290

Bibliografía:

[BRY16] Cap.3

Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016

Signatura ESIIT/[C.1 BRY com](#)

Práctica 1: Entorno de desarrollo GNU

Estructura de Computadores

Gustavo Romero López

Updated: 24 de septiembre de 2020

Arquitectura y Tecnología de Computadores

Índice

| | |
|-----------------|----------------------|
| 1. Índice | 6.3 C++ |
| 2. Objetivos | 6.4 32 bits |
| 3. Introducción | 6.5 64 bits |
| 4. C | 6.6 ASM + C |
| 5. Ensamblador | 6.7 Optimización |
| 6. Ejemplos | 7. Compiler Explorer |
| 6.1 hola | 8. Enlaces |
| 6.2 make | |

Objetivos

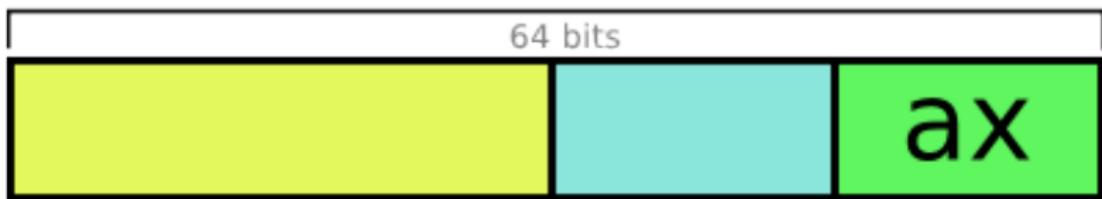
- Nociones de ensamblador 80x86 de 64 bits.
- Linux es tu amigo: si no sabes algo pregunta... **man**.
- Hoy aprenderemos varias cosas:
 - El esqueleto de un programa básico en ensamblador.
 - Como aprender de un maestro: el compilador **gcc**.
 - Herramientas clásicas del entorno de programación UNIX:
 - **make**: hará el trabajo sucio y rutinario por nosotros.
 - **as**: el ensamblador.
 - **ld**: el enlazador.
 - **gcc**: el compilador.
 - **nm**: lista los símbolos de un fichero.
 - **objdump**: el desensamblador.
 - **gdb** y **ddd** (gdb con cirugía estética): los depuradores.
 - Herramienta web: Compiler Explorer

Ensamblador 80x86

- ④ Los **80x86** son una familia de procesadores.
- ④ El más utilizado junto a los procesadores **ARM**.
- ④ En estas prácticas vamos a centrarnos en su **lenguaje ensamblador** (inglés).
- ④ El lenguaje ensamblador es el más básico, tras el binario, con el que podemos escribir programas utilizando las **instrucciones** que entiende el procesador.
- ④ Cualquier estructura de un lenguajes de alto nivel pueden crearse mediante instrucciones muy sencillas.
- ④ Normalmente es utilizado para poder acceder a partes que los lenguajes de alto nivel nos ocultan, complican o hacen de forma inconveniente.

Arquitectura 80x86: el registro A

rax



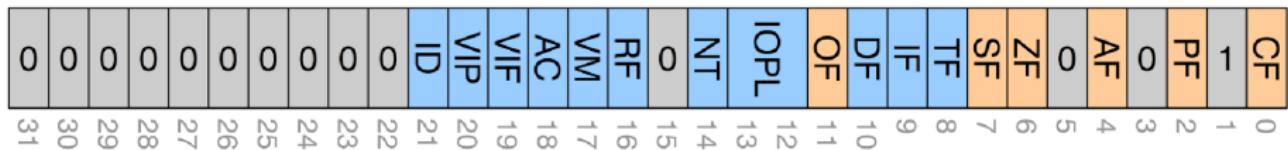
eax

Arquitectura 80x86: registros completos

| | | | | | | | | | | | | | | | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|--------|-------|-------|-------|-----|------|-------|--------|--------|------|------|-------|-----|-----|
| ZMM0 | YMM0 | XMM0 | ZMM1 | YMM1 | XMM1 | ST(0) | MM0 | ST(1) | MM1 | ALAH | AXEAX | RAX | RW | R8D | R8 | R12W | R12D | R12 | CR0 | CR4 | |
| ZMM2 | YMM2 | XMM2 | ZMM3 | YMM3 | XMM3 | ST(2) | MM2 | ST(3) | MM3 | BLBH | BXBX | RBX | RW | R9W | R9D | R9 | R13W | R13D | R13 | CR1 | CR5 |
| ZMM4 | YMM4 | XMM4 | ZMM5 | YMM5 | XMM5 | ST(4) | MM4 | ST(5) | MM5 | CLCH | CXECX | RCX | R10W | R10D | R10 | R14W | R14D | R14 | CR2 | CR6 | |
| ZMM6 | YMM6 | XMM6 | ZMM7 | YMM7 | XMM7 | ST(6) | MM6 | ST(7) | MM7 | DIBD | DXEDX | RDX | R11W | R11D | R11 | R15W | R15D | R15 | CR3 | CR7 | |
| ZMM8 | YMM8 | XMM8 | ZMM9 | YMM9 | XMM9 | | | | | BPL | BPEB | RBP | DIL | DI | EDI | RDI | | | IP | EIP | RIP |
| ZMM10 | YMM10 | XMM10 | ZMM11 | YMM11 | XMM11 | CW | FP_IP | FP_DP | FP_CS | SL | SI | ESI | RSI | SPN | SP | ESP | RSP | | MSW | CR9 | |
| ZMM12 | YMM12 | XMM12 | ZMM13 | YMM13 | XMM13 | SW | | | | | | | | | | | | | CR10 | | |
| ZMM14 | YMM14 | XMM14 | ZMM15 | YMM15 | XMM15 | TW | | | | | | | | | | | | | CR11 | | |
| ZMM16 | ZMM17 | ZMM18 | ZMM19 | ZMM20 | ZMM21 | ZMM22 | ZMM23 | FP_DS | | | | | | | | | | | CR12 | | |
| ZMM24 | ZMM25 | ZMM26 | ZMM27 | ZMM28 | ZMM29 | ZMM30 | ZMM31 | FP_OPC | FP_DP | FP_IP | CS | SS | DS | GDTR | IDTR | | DRO | DR6 | CR13 | | |
| | | | | | | | | | | | ES | FS | GS | TR | LDT | | DR1 | DR7 | CR14 | | |
| | | | | | | | | | | | | | | FLAGS | EFLAGS | RFLAGS | DR2 | DR8 | CR15 | | |
| | | | | | | | | | | | | | | | | | DR3 | DR9 | MXCSR | | |
| | | | | | | | | | | | | | | | | | DR4 | DR10 | DR12 | | |
| | | | | | | | | | | | | | | | | | DR5 | DR11 | DR13 | | |
| | | | | | | | | | | | | | | | | | | DR15 | DR15 | | |

Arquitectura 80x86: banderas

eflags register



■ Reserved flags

■ System flags

■ Arithmetic flags

TF: Trap
IF: Interrupt
DF: Direction

CF: Carry
PF: Parity
AF: Adjust
ZF: Zero
SF: Sign
OF: Overflow

Arquitectura 80x86: paso de parámetros a funciones

| | | | |
|------|---------------|------|--------------|
| %rax | Return value | %r8 | Argument #5 |
| %rbx | Callee saved | %r9 | Argument #6 |
| %rcx | Argument #4 | %r10 | Caller saved |
| %rdx | Argument #3 | %r11 | Caller Saved |
| %rsi | Argument #2 | %r12 | Callee saved |
| %rdi | Argument #1 | %r13 | Callee saved |
| %rsp | Stack pointer | %r14 | Callee saved |
| %rbp | Callee saved | %r15 | Callee saved |

Programa mínimo en C... todos ellos equivalentes

minimo1.c

```
int main() {}
```

minimo2.c

```
int main() { return 0; }
```

minimo3.c

```
#include <stdlib.h>
int main() { exit(0); }
```

Trasteando el programa mínimo en C


```
linux-vdso.so.1 (0x00007ffe2ddbc000)
libc.so.6 => /lib64/libc.so.6 (0x00007fbc5043a000)
/lib64/ld-linux-x86-64.so.2 (0x0000558dbe5aa000)
```

- Examinar biblioteca: `objdump -d /lib64/libc.so.6`

Ensamblador desde 0: secciones básicas de un programa

```
1 .data  
2  
3 .text
```

Ensamblador desde 0: punto de entrada

```
1 .text  
2     .global _start
```

Ensamblador desde 0: datos

```
1 .data
2 msg:      .string "¡hola, mundo!\n"
3 tam:      .quad . - msg
```

Ensamblador desde 0: código

```
1 write:    mov    $1,    %rax    # write
2           mov    $1,    %rdi    # stdout
3           mov    $msg,   %rsi    # texto
4           mov    tam,   %rdx    # tamaño
5           syscall          # llamada a write
6           ret
7
8 exit:     mov    $60,   %rax    # exit
9           xor    %rdi,   %rdi    # 0
10          syscall         # llamada a exit
```

Ensamblador desde 0: ejemplo básico hola.s

```
1 .data
2 msg:     .string "¡hola, mundo!\n"
3 tam:     .quad . - msg
4
5 .text
6     .global _start
7
8 write:   mov    $1,    %rax  # write
9         mov    $1,    %rdi  # stdout
10        mov    $msg,   %rsi  # texto
11        mov    tam,   %rdx  # tamaño
12        syscall          # llamada a write
13        ret
14
15 exit:    mov    $60,   %rax  # exit
16        xor    %rdi,   %rdi  # 0
17        syscall          # llamada a exit
18        ret
19
20 _start:  call   write      # llamada a función
21         call   exit       # llamada a función
22
```

¿Cómo hacer ejecutable mi programa?

¿Cómo hacer ejecutable el código anterior?

- ◎ opción a: ensamblar + enlazar
 - `as hola.s -o hola.o`
 - `ld hola.o -o hola`
- ◎ opción b: compilar = ensamblar + enlazar
 - `gcc -nostdlib hola.s -o hola`
- ◎ opción c: que lo haga alguien por mi → make
 - `makefile`: fichero con definiciones, objetivos y recetas.

Ejercicios:

1. Cree un ejecutable a partir de `hola.s`.
2. Use `file` para ver el tipo de cada fichero.
3. Descargue el fichero `makefile`, pruébelo e intente hacer alguna modificación.
4. Examine el código ensamblador con `objdump -d hola`.

makefile

<http://pccito.ugr.es/~gustavo/ec/practicas/1/makefile>

```
ASM = $(wildcard *.s)
SRC = $(wildcard *.c *.cc)
EXE = $(basename $(ASM) $(SRC))
ATT = $(EXE:=.att)
```

```
CFLAGS = -g -std=c11 -Wall
CXXFLAGS = $(CFLAGS:c11=c++11)
```

```
all: $(EXE) $(ATT)
```

```
clean:
    -rm -fv $(ATT) $(EXE) *~
```

Ejemplo en C++: hola-c++.cc

```
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "¡hola, mundo!"
6             << std::endl;
7 }
```

- ◎ ¿Qué hace gcc con mi programa?
- ◎ La única forma de saberlo es desensamblarlo:
 - Sintaxis AT&T: objdump -C -d hola-c++
 - Sintaxis Intel: objdump -C -d hola-c++ -M intel

Ejercicios:

5. ¿Qué hace ahora diferente la función main() respecto a C?

```
1 write:    movl    $4, %eax      # write
2             movl    $1, %ebx      # salida estándar
3             movl    $msg, %ecx      # cadena
4             movl    tam, %edx      # longitud
5             int     $0x80      # llamada a write
6             ret          # retorno
7
8 exit:     movl    $1, %eax      # exit
9             xorl    %ebx, %ebx      # 0
10            int    $0x80      # llamada a exit
```

Ejercicios:

6. Descargue hola32.s. Ejecute el programa instrucción por instrucción con el ddd hasta comprender como funciona.
7. Si quiere aprender un poco más estudie hola32p.s. Sobre el mismo podemos destacar: código de 32 bits, uso de “*little endian*”, llamada a subrutina, uso de la pila y codificación de caracteres.

Depuración: hola64.s

ejemplo de 64 bits

```
1 write:    mov      $1,      %rax    # write
2             mov      $1,      %rdi    # stdout
3             mov      $msg,    %rsi    # texto
4             mov      tam,    %rdx    # tamaño
5             syscall           # llamada a write
6             ret
7
8 exit:     mov      $60,    %rax    # exit
9             xor      %rdi,    %rdi    # 0
10            syscall          # llamada a exit
11            ret
```

Ejercicios:

8. Descargue hola64.s. Ejecute el programa instrucción por instrucción con el ddd hasta comprender como funciona.
9. Compare hola64.s con hola64p.s. Sobre este podemos destacar: código de 64 bits, llamada a subrutina, uso de la pila y codificación de caracteres.

¿Dónde están mis datos?

printf-c-1.c y printf-c-2.c

- ◎ ¿Sabes C? \Longleftrightarrow ¿Has usado la función printf()?

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int i = 12345;
6     printf("i=%d\n", i);
7     return 0;
8 }
```

```
1 #include <stdio.h>
2
3 int i = 12345;
4 char *formato = "i=%d\n";
5
6 int main()
7 {
8     printf(formato, i);
9     return 0;
10 }
```

Ejercicios:

10. ¿En qué se parecen y en qué se diferencian printf-c-1.c y printf-c-2.c? nm, objdump y kdiff3 serán muy útiles...

Mezclando lenguajes: ensamblador y C (32 bits) printf32.s

```
1 .data
2 i:      .int 12345          # variable entera
3 f:      .string "i = %d\n" # cadena de formato
4
5 .text
6     .extern printf        # printf en otro sitio
7     .globl _start         # función principal
8
9 _start: push (i)           # apila i
10    push $f               # apila f
11    mov $0, %eax          # n de registros vectoriales
12    call printf           # llamada a printf
13    add $8, %esp           # restaura pila
14
15    movl $1, %eax          # exit
16    xorl %ebx, %ebx       # 0
```

Ejercicios:

11. Descargue y compile printf32.s.
12. Modifique printf32.s para que finalice mediante la función exit() de C (`man 3 exit`). Solución: printf32e.s.

Mezclando lenguajes: ensamblador y C (64 bits) printf64.s

```
1 .data
2 i:      .int 12345          # variable entera
3 f:      .string "i = %d\n" # cadena de formato
4
5 .text
6     .globl _start
7
8 _start: mov $f, %rdi        # formato
9         mov (i), %rsi        # i
10        xor %rax, %rax      # n de registros vectoriales
11        call printf         # llamada a función
12
13        xor %rdi, %rdi      # valor de retorno
14        call exit           # llamada a función
```

Ejercicios:

13. Descargue y compile printf64.s.
14. Busque las diferencias entre printf32.s y printf64.s.

Optimización: sum.cc

```
1 int main()
2 {
3     int sum = 0;
4
5     for (int i = 0; i < 10; ++i)
6         sum += i;
7
8     return sum;
9 }
```

Ejercicios:

15. ¿Cómo implementa gcc los bucles **for**?
16. Observe el código de la función **main()** al compilarlo...
 - sin optimización: g++ -O0 sum.cc -o sum
 - con optimización: g++ -O3 sum.cc -o sum

Optimización: función main() de sum.cc

sin optimización (gcc -O0)

```
4005b6: 55                      push    %rbp
4005b7: 48 89 e5                mov     %rsp,%rbp
4005ba: c7 45 fc 00 00 00 00    movl    $0x0,-0x4(%rbp)
4005c1: c7 45 f8 00 00 00 00    movl    $0x0,-0x8(%rbp)
4005c8: eb 0a                   jmp    4005d4 <main+0x1e>
4005ca: 8b 45 f8                mov     -0x8(%rbp),%eax
4005cd: 01 45 fc                add    %eax,-0x4(%rbp)
4005d0: 83 45 f8 01             addl   $0x1,-0x8(%rbp)
4005d4: 83 7d f8 09             cmpl   $0x9,-0x8(%rbp)
4005d8: 7e f0                   jle    4005ca <main+0x14>
4005da: 8b 45 fc                mov     -0x4(%rbp),%eax
4005dd: 5d                      pop    %rbp
4005de: c3                      retq
```

con optimización (gcc -O3)

```
4004c0: b8 2d 00 00 00          mov     $0x2d,%eax
4004c5: c3                      retq
```

Compiler Explorer: <https://godbolt.org/z/7uIN9y>

The screenshot shows the Compiler Explorer interface with two compiler panes. The left pane displays C++ source code, and the right pane shows the assembly output generated by two different compilers.

C++ source #1:

```
template <class T>
concept bool Addable =
    requires (T t) { t + t; };

int main()
{
    int x = 1, y = 2;
    Addable a = x + y;
    return a;
}
```

x86-64 gcc 8.2 (Editor #1, Compiler #1) C++:

```
-fconcepts
```

```
main:
pushq %rbp
movq %rsp, %rbp
movl $1, -4(%rbp)
movl $2, -8(%rbp)
movl -4(%rbp), %edx
movl -8(%rbp), %eax
addl %edx, %eax
movl %eax, -12(%rbp)
movl -12(%rbp), %eax
popq %rbp
ret
```

x86-64 gcc 8.2 (Editor #1, Compiler #2) C++:

```
-fconcepts -O3
```

```
main:
movl $3, %eax
ret
```

Compiler Explorer: <https://godbolt.org/z/vUF7yA>

The screenshot illustrates the Compiler Explorer interface, comparing the assembly output of two different compilers for the same C++ code.

C++ Source Code:

```
template<typename T> T adder(T v)
{
    return v;
}

template<typename T, typename... Args>
{
    return first + adder(args...);
}

int main()
{
    return adder(1, 2, 3, 4, 5);
}
```

Assembly Output (Compiler #1, x86-64 gcc 8.2):

```
main:
    pushq %rbp
    movq %rsp, %rbp
    movl $5, %r8d
    movl $4, %ecx
    movl $3, %edx
    movl $2, %esi
    movl $1, %edi
    call int adder<int, int, int, int>(%r8d, %rcx, %rdx, %rsi)
    nop
    popq %rbp
    ret
int adder<int, int, int, int>(int, int, int, int):
    pushq %rbp
    movq %rsp, %rbp
    subq $32, %rsp
    movl %edi, -4(%rbp)
    movl %esi, -8(%rbp)
    movl %edx, -12(%rbp)
    movl %ecx, -16(%rbp)
    movl %r8d, -20(%rbp)
    movl -20(%rbp), %ecx
    movl -16(%rbp), %edx
    movl -12(%rbp), %esi
    movl -8(%rbp), %eax
    movl %eax, %edi
    call int adder<int, int, int>(%r8d, %rcx, %rsi)
    movl %eax, %edx
    movl -4(%rbp), %eax
    addl %edx, %eax
    leave
    ret
int adder<int, int, int>(int, int):
    pushq %rbp
    movq %rsp, %rbp
    subq $16, %rsp
```

Assembly Output (Compiler #2, x86-64 gcc 8.2):

```
main:
    movl $15, %eax
    ret
```

Enlaces de interés

Manuales:

◎ Hardware:

- AMD
- Intel

◎ Software:

- AS
- NASM

Programación:

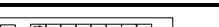
- ◎ Programming from the ground up
- ◎ Linux Assembly

Chuletas:

- ◎ Chuleta del 8086
- ◎ Chuleta del GDB

i para más información ver especificaciones de la instrucción.

Flags: +≡Afectado por esta instrucción ?=Indefinido luego de esta instrucción

| ARITMÉTICOS | | Código | Operación | Flags | | | | | | | | | |
|-------------|------------------------------|-----------------|---|-----------------------|-----|---|---|-----|----|------|---|-----|-----|
| Nombre | Comentario | | | O | D | I | T | S | Z | A | P | C | |
| ADD | Suma | ADD Dest,Fuente | Dest:=Dest+ Fuente | ± | | | | ± | ± | ± | ± | ± | |
| ADC | Suma con acarreo | ADC Dest,Fuente | Dest:=Dest+ Fuente +CF | ± | | | | ± | ± | ± | ± | ± | |
| SUB | Resta | SUB Dest,Fuente | Dest:=Dest- Fuente | ± | | | | ± | ± | ± | ± | ± | |
| SBB | Resta con acarreo | SBB Dest,Fuente | Dest:=Dest-(Fuente +CF) | ± | | | | ± | ± | ± | ± | ± | |
| DIV | División (sin signo) | DIV Op | Op=byte: AL:=AX / Op | AH:=Resto | ? | | | ? | ? | ? | ? | ? | |
| DIV | División (sin signo) | DIV Op | Op=word: AX:=DX:AX / Op | DX:=Resto | ? | | | ? | ? | ? | ? | ? | |
| DIV 386 | División (sin signo) | DIV Op | Op=doublew.: EAX:=EDX:EAX / Op | EDX:=Resto | ? | | | ? | ? | ? | ? | ? | |
| IDIV | División entera con signo | IDIV Op | Op=byte: AL:=AX / Op | AH:=Resto | ? | | | ? | ? | ? | ? | ? | |
| IDIV | División entera con signo | IDIV Op | Op=word: AX:=DX:AX / Op | DX:=Resto | ? | | | ? | ? | ? | ? | ? | |
| IDIV 386 | División entera con signo | IDIV Op | Op=doublew.: EAX:=EDX:EAX / Op | EDX:=Resto | ? | | | ? | ? | ? | ? | ? | |
| MUL | Multiplicación (sin signo) | MUL Op | Op=byte: AX:=AL*Op | si AH=0 ◆ | ± | | | ? | ? | ? | ? | ± | |
| MUL | Multiplicación (sin signo) | MUL Op | Op=word: DX:AX:=AX*Op | si DX=0 ◆ | ± | | | ? | ? | ? | ? | ± | |
| MUL 386 | Multiplicación (sin signo) | MUL Op | Op=double: EDX:EAX:=EAX*Op | si EDX=0 ◆ | ± | | | ? | ? | ? | ? | ± | |
| IMUL i | Multiplic. entera con signo | IMUL Op | Op=byte: AX:=AL*Op | si AL es suficiente ◆ | ± | | | ? | ? | ? | ? | ± | |
| IMUL | Multiplic. entera con signo | IMUL Op | Op=word: DX:AX:=AX*Op | si AX es suficiente ◆ | ± | | | ? | ? | ? | ? | ± | |
| IMUL 386 | Multiplic. entera con signo | IMUL Op | Op=double: EDX:EAX:=EAX*Op | si EAX es sufi. ◆ | ± | | | ? | ? | ? | ? | ± | |
| INC | Incrementar | INC Op | Op:=Op+1 (El Carry no resulta afectado !) | ± | | | | ± | ± | ± | ± | ± | |
| DEC | Decrementar | DEC Op | Op:=Op-1 (El Carry no resulta afectado !) | ± | | | | ± | ± | ± | ± | ± | |
| CMP | Comparar | CMP Op1,Op2 | Op1-Op2 | ± | | | | ± | ± | ± | ± | ± | |
| SAL | Desplazam. aritm. a la izq. | SAL Op,Cantidad |  | CF | MSB | | | LSB | ←0 | →MSB | | LSB | →CF |
| SAR | Desplazam. aritm. a la der. | SAR Op,Cantidad | | i | | | | ± | ± | ? | ± | ± | |
| RCL | Rotar a la izq. c/acarreo | RCL Op,Cantidad |  | CF | MSB | | | LSB | ← | →MSB | | LSB | →CF |
| RCR | Rotar a la derecha c/acarreo | RCR Op,Cantidad | | i | | | | ± | ± | ? | ± | ± | |
| ROL | Rotar a la izquierda | ROL Op,Cantidad |  | CF | MSB | | | LSB | ← | →MSB | | LSB | →CF |
| ROR | Rotar a la derecha | ROR Op,Cantidad | | i | | | | ± | ± | ? | ± | ± | |

i para más información ver especificaciones de la instrucción.

♦ entonces $CF=0$, $QE=0$ sino $CF=1$, $QE=1$

| LÓGICOS | | Código | Operación | Flags | | | | | | | | |
|---------|-----------------------------|-----------------|--|-------|---|---|---|---|---|---|---|---|
| Nombre | Comentario | | | O | D | I | T | S | Z | A | P | C |
| NEG | Negación (complemento a 2) | NEG Op | Op:=0-Op si Op=0 entonces CF:=0 sino CF:=1 | ± | | | | ± | ± | ± | ± | ± |
| NOT | Invertir cada bit | NOT Op | Op:=~Op (invierte cada bit) | | | | | | | | | |
| AND | 'Y' (And) lógico | AND Dest,Fuente | Dest:=Dest&Fuente | 0 | | | | ± | ± | ? | ± | 0 |
| OR | 'O' (Or) lógico | OR Dest,Fuente | Dest:=Dest Fuente | 0 | | | | ± | ± | ? | ± | 0 |
| XOR | 'O' (Or) exclusivo | XOR Dest,Fuente | Dest:=Dest (xor) Fuente | 0 | | | | ± | ± | ? | ± | 0 |
| SHL | Desplazam. lógico a la izq. | SHL Op,Cantidad |  | i | | | | ± | ± | ? | ± | ± |
| SHR | Desplazam. lógico a la der. | SHR Op,Cantidad | | i | | | | ± | ± | ? | ± | ± |

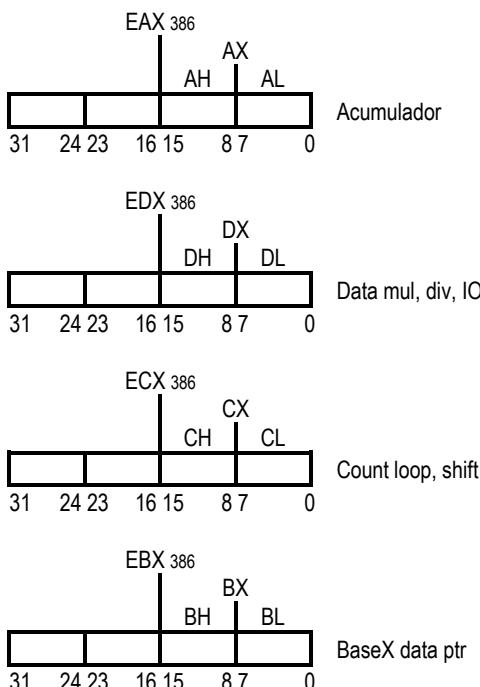
Tabla de Códigos 2/2

| MISCELÁNEOS | | Código | Operación | Flags | | | | | | | |
|-------------|---------------------------|-----------------|---|-------|---|---|---|---|---|---|---|
| Nombre | Comentario | | | O | D | I | T | S | Z | A | P |
| NOP | Hacer nada | NOP | No hace operación alguna | | | | | | | | |
| LEA | Cargar dirección Efectiva | LEA Dest,Fuente | Dest := dirección fuente | | | | | | | | |
| INT | Interrupción | INT Num | Interrumpe el progr. actual, corre la subrutina de int. | | 0 | 0 | | | | | |

| SALTOS (generales) | | Código | Operación | Name | Comentario | Código | Operación |
|--------------------|---------------------------|-----------|-----------|-------|-----------------------------|------------|-----------|
| Nombre | Comentario | | | | | | |
| CALL | Llamado a subrutina | CALL Proc | | RET | Retorno de subrutina | RET | |
| JMP | Saltar | JMP Dest | | | | | |
| JE | Saltar si es igual | JE Dest | (= JZ) | JNE | Saltar si no es igual | JNE Dest | (= JNZ) |
| JZ | Saltar si es cero | JZ Dest | (= JE) | JNZ | Saltar si no es cero | JNZ Dest | (= JNE) |
| JCXZ | Saltar si CX es cero | JCXZ Dest | | JECXZ | Saltar si ECX es cero | JECXZ Dest | 386 |
| JP | Saltar si hay paridad | JP Dest | (= JPE) | JNP | Saltar si no hay paridad | JNP Dest | (= JPO) |
| JPE | Saltar si hay paridad par | JPE Dest | (= JP) | JPO | Saltar si hay paridad impar | JPO Dest | (= JNP) |

| SALTOS Sin Signo (Cardinal) | | | | SALTOS Con Signo (Integer) | | | |
|-----------------------------|--------------------------------|-----------|---------------|----------------------------|----------------------------------|-----------|----------|
| JA | Saltar si es superior | JA Dest | (= JNBE) | JG | Saltar si es mayor | JG Dest | (= JNLE) |
| JAE | Saltar si es superior o igual | JAE Dest | (= JNB = JNC) | JGE | Saltar si es mayor o igual | JGE Dest | (= JNL) |
| JB | Saltar si es inferior | JB Dest | (= JNAE = JC) | JL | Saltar si es menor | JL Dest | (= JNGE) |
| JBE | Saltar si es inferior o igual | JBE Dest | (= JNA) | JLE | Saltar si es menor o igual | JLE Dest | (= JNG) |
| JNA | Saltar si no es superior | JNA Dest | (= JBE) | JNG | Saltar si no es mayor | JNG Dest | (= JLE) |
| JNAE | Saltar si no es super. o igual | JNAE Dest | (= JB = JC) | JNGE | Saltar si no es mayor o igual | JNGE Dest | (= JL) |
| JNB | Saltar si no es inferior | JNB Dest | (= JAE = JNC) | JNL | Saltar si no es inferior | JNL Dest | (= JGE) |
| JNBE | Saltar si no es infer. o igual | JNBE Dest | (= JA) | JNLE | Saltar si no es menor o igual | JNLE Dest | (= JG) |
| JC | Saltar si hay carry | JC Dest | | JO | Saltar si hay Overflow | JO Dest | |
| JNC | Saltar si no hay carry | JNC Dest | | JNO | Saltar si no hay Overflow | JNO Dest | |
| | | | | JS | Saltar si hay signo (=negativo) | JS Dest | |
| | | | | JNS | Saltar si no hay signo (=posit.) | JNS Dest | |

Registros Generales:



Ejemplo:

```
.DOSSEG
.MODEL SMALL
.STACK 1024
Two EQU 2 ; Constante
.VarB DB ?
.VarW DW 1010b ; define un Byte, cualquier valor
.VarW2 DW 257 ; define un Word, en binario
.VarD DD 0AFFFFh ; define un DoubleWord, en hexa
.S DB "Hello !",0 ; define un String
.CODE
main: MOV AX,DGROUP ; resuelto por el linker
      MOV DS,AX ; inicializa el reg. de segmento de datos
      MOV [VarB],42 ; inicializa VarB
      MOV [VarD],-7 ; setea VarD
      MOV BX,Offset[S] ; dirección de "H" de "Hello !"
      MOV AX,[VarW] ; poner el valor en el acumulador
      ADD AX,[VarW2] ; suma VarW2 a AX
      MOV [VarW2],AX ; almacena AX en VarW2
      MOV AX,4C00h ; regresa al sistema
      INT 21h
END main
```



Flags: - - - - O D I T S Z - A - P - C

Flags de Control (cómo se manejan las instrucciones):

- D: Dirección 1=Los op's String se procesan de arriba hacia abajo
I: Interrupción Indica si pueden ocurrir interrupciones o no.
T: Trampa Paso por paso para debugging

Flags de Estado (resultado de las operaciones):

- C: Carry resultado de operac. sin signo es muy grande o inferior a cero
O: Overflow resultado de operac. sin signo es muy grande o pequeño.
S: Signo Signo del resultado. Razonable sólo para enteros. 1=neg. 0=posit.
Z: Cero Resultado de la operación es cero. 1=Cero
A: Carru Aux. Similar al Carry, pero restringido para el nibble bajo únicamente
P: Paridad 1=el resultado tiene cantidad par de bits en uno



RullenCastro

www.wuolah.com/student/RullenCastro



750-preguntas-test-SWAD-EC-Resueltas.pdf

750 preguntas test SWAD EC Resueltas



2º Estructura de Computadores



Grado en Ingeniería Informática



Escuela Técnica Superior de Ingenierías Informática y de
Telecomunicación
Universidad de Granada



Descarga la APP de Wuolah.
Ya disponible para el móvil y la tablet.





**KEEP
CALM
AND
ESTUDIA
UN POQUITO**

1-La(s) instrucción(es) necesaria(s) para cargar el dividendo 0xa30bf18a en la pareja edx:eax como

paso previo a una división sin signo son:

- b) movl \$0xa30bf18a,%eax
 xorl %edx,%edx

2-¿Qué hace gcc -O?

- a) Compilar con optimización suave

3-La diferencia entre el flag de acarreo y de overflow es que...

- c) Ambos se recalculan tras cada operación aritmético-lógica con ints, correspondiendo al programador consultar uno u otro según piense que sus datos son con o sin signo

4-De las siguientes instrucciones, ¿cuál no utiliza un modo de direccionamiento implícito?

Push

5-El direccionamiento relativo a registro base utiliza...

- c) un registro y un desplazamiento.

6-En un sistema de gestión de interrupciones mediante "polling" o sondeo, el dispositivo que

solicita la interrupción envía, junto con la señal de petición de interrupción, su correspondiente vector de interrupción.

Falso

7-Indique cuál de las siguientes afirmaciones sobre el ENIAC no es correcta:

Como los computadores actuales, era una máquina binaria, es decir, los números estaban representados en forma binaria y los cálculos aritméticos se realizaban también en el sistema binario.

8-Se puede programar un controlador de interrupciones 8259 de manera que atienda equitativamente a 8 dispositivos de igual prioridad (cada vez que se atiende a un dispositivo, éste

pasa automáticamente a tener la prioridad más baja).

Verdadero

9-¿Qué modificador (switch) de ld hace falta para enlazar una aplicación de 32bits en un sistema de

64bits en el que se ha instalado también el compilador de 32bits?

-m elf_i386

10-En una bomba como las estudiadas en prácticas, del tipo...

0x080486e8 <main+120>: call 0x8048524 <strcmp>

```
0x080486ed <main+125>: test %eax,%eax  
0x080486ef <main+127>: je 0x80486f6 <main+134>  
0x080486f1 <main+129>: call 0x8048604 <boom>  
0x080486f6 <main+134>: ...
```

- a) el valor que tenga %eax
- b) el string almacenado a partir de donde apunta %eax
- c) el entero almacenado a partir de donde apunta %eax
- ✓ d) ninguna de las anteriores

11-¿Qué modo de funcionamiento permite a la interfaz de periféricos programable 8255 utilizar un bus bidireccional?

2

12-En IA32, el registro contador de programa se denomina:

EIP

13-La desventaja de las transferencias por bloques en un bus es que hay que transmitir todas las direcciones consecutivas de los datos del bloque.

Falso

14-La consulta de estado que se puede llevar a cabo en la E/S programada sirve para...

Consultar si el dispositivo está dispuesto para recibir datos o tiene datos disponibles

15-El controlador de interrupciones programable 8259 no permite enmascarar selectivamente líneas de interrupción.

Falso

16-El concepto de "ventanas de registros solapadas" es un mecanismo eficiente de llamada/retorno de procedimientos, utilizado en el procesador RISC-I.

Verdadero

17-Una posible codificación en microinstrucciones de la instrucción CALL X es:
 $SP=SP-1 ; m[SP]=PC ; PC=X$

18-La técnica de sondeo, escrutinio o "polling"...

Se utiliza para identificar la fuente de una interrupción

19-El lenguaje máquina es...

Difícil de codificar manualmente.

20-Un programa de ordenador que convierte un programa fuente de alto nivel completo en lenguaje máquina se llama un:
Compilador

21-Un puerto de entrada de un bit puede estar constituido únicamente por un simple biestable tipo
D.
Falso

22-Si el registro EAX contiene X, la sentencia en C
x &= 0x1;
se traducirá a ensamblador como:
andl \$1, %eax

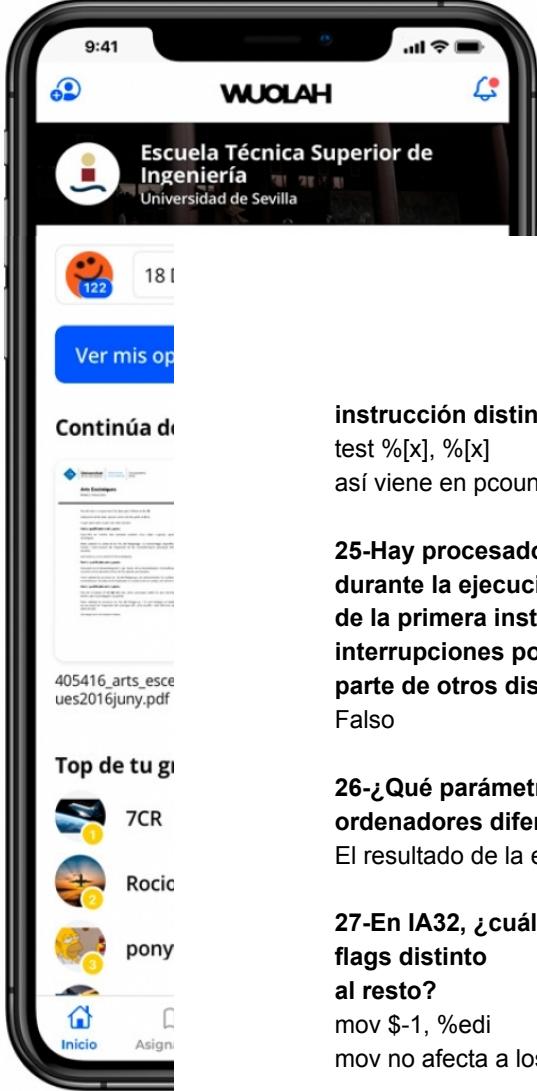
23-En la práctica "media" se suma una lista de 32 enteros de 4 B con signo para producir una media y un resto usando la instrucción IDIV. ¿Cuál de las siguientes afirmaciones es falsa?

La media se redondea al entero más próximo
división truncada IDIV - no se redondea, se trunca

24-La práctica "popcount" debía calcular la suma de bits (peso Hamming) de los elementos de un array. Un estudiante entrega la siguiente versión de popcount3:

```
int popcount3(unsigned* array,
int len){
    int i, res = 0;
    unsigned x;
    for( i=0; i<len; i++ ) {
        x = array[i];
        asm("ini3: \n"
            "shr %[x] \n"
            "adc $0, %[r] \n"
            "add $0, %[x] \n"
            "jne ini3 \n"
            : [r] "+r" (res)
            : [x] "r" (x );
    }
    return res;
}
```

Esta función produce siempre el resultado correcto, a pesar de que una instrucción máquina en la sección `asm()` es distinta a la que se esperaba después de haber estudiado `pcount_r` en teoría. La



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

instrucción distinta también se podría haber cambiado por...

test %[x], %[x]
así viene en pcount_r (Tema 2.3 tr.38)

25-Hay procesadores que desactivan automáticamente las interrupciones sólo durante la ejecución de la primera instrucción de la ISR, con lo cual es inevitable que se produzcan interrupciones por parte de otros dispositivos de forma anidada.
Falso

26-¿Qué parámetro es más importante para comparar la velocidad de dos ordenadores diferentes?

El resultado de la ejecución de un conjunto de programas de prueba.

27-En IA32, ¿cuál de los siguientes fragmentos de programa tiene un efecto sobre los flags distinto al resto?

mov \$-1, %edi
mov no afecta a los flags

28-El especificador de operando de una instrucción, cuando existe, es siempre una dirección de memoria o de entrada/salida
Falso

29-Una de las diferencias entre el 8086 y el 8088 es que este último no dispone de BIU ("Bus Interface Unit").
Falso

30-Respecto a registros base e índice en IA32, la excepción es que ESP no puede ser registro índice

31-La práctica "popcount" debía calcular la suma de bits de los elementos de un array. Un estudiante entrega la siguiente versión de popcount4:

```
int popcount4(unsigned* array, int len)
{
    int i, j, res = 0;
    for(i = 0; i < len; ++i) {
        unsigned x = array[i];
        int n = 0;
        do {
            n += x & 0x01010101L;
            x >>= 1;
        } while(x);
```

```
for(j = 16; j == 1; j /= 2){  
    n ^= (n >= j);  
}  
res += n & 0xff;  
}  
return res;  
}
```

Esta función `popcount4`:

produce el resultado correcto

Caso real, entregado en prácticas. La máscara está pensada para `for(j=0;j<8;j++)`. En lugar de eso,

se hace `do...while(x)`, con lo cual no se ahorran iteraciones y todo el resultado queda acumulado

en el LSB de `n`. El `for(j)` es absurdo, no itera ninguna vez. El resultado se extrae y acumula con

`n&0xFF`. Es correcto, pero no mejora la eficiencia. `popcount2` es igual de eficiente y más elegante, porque no tiene código superfluo.

32-Una forma usual de realizar el arbitraje distribuido consiste en una competición por la concesión del bus realizada por medio del envío por cada maestro peticionario de un número de arbitraje que lo identifica, de manera que un solo número "gane", y se le conceda el bus al maestro con ese número ganador.

Verdadero

33-¿Qué modificador (switch) de gcc hace falta para compilar una aplicación de 32 bits en un sistema de 64 bits?

`-m32`

34-¿Cuál de las siguientes afirmaciones es incorrecta?

a) En el direccionamiento inmediato el dato se encuentra en la propia instrucción

b) El direccionamiento indexado es útil para manejo de vectores.

c) En el direccionamiento implícito no se indica la ubicación del operando

***d) El direccionamiento indirecto indica la dirección del operando.

INDICA UN PUNTERO AL OPERANDO

35-¿Cuál de las siguientes afirmaciones sobre los lenguajes ensambladores es falsa?

A cada sentencia le corresponde un conjunto preestablecido de instrucciones máquina.

36-La conexión de un 8086 a un sistema de memoria y E/S requiere algún circuito externo más en modo máximo que en modo mínimo.

Verdadero

37-¿Cuál de las siguientes afirmaciones es incorrecta?

- a) Las arquitecturas RISC simplifican la decodificación
- b) Las arquitecturas RISC son del tipo registro-registro.
- c) En las arquitecturas CISC hay más instrucciones que en las RISC.
- >d) El tamaño de una instrucción en lenguaje máquina siempre ocupa dos bytes en los procesadores RISC

38-En la nomenclatura del ensamblador de IA32, una cantidad de 16 bits es designada como:

Word

39-Suponga un programa residente que utiliza DMA por robo de ciclo para reproducir a través de

una tarjeta de sonido una melodía de fondo, que se encontraba en memoria, de manera simultánea

a la ejecución de un programa de usuario. La velocidad de ejecución de dicho programa de usuario

no se verá afectada por el programa residente, ya que éste utiliza DMA.

Falso

40-Una instrucción típica de entrada / salida tiene

tiene dos argumentos: un registro del procesador y una dirección de puerto de E/S claramente inspirado en el repertorio IA32 - no encuentro dónde aparece en las transparencias -

"I da A Data" en transparencia 40 es mapeado a memoria - evitar volver a poner esta pregunta en exámenes

41-Si D es un desplazamiento, RI un registro índice e I una constante apropiada, el modo de

direcciónamiento indexado con postautodecremento realiza EA = RI+D y a continuación RI = RI-I.

(EA = Effective Address)

Verdadero

42-En un computador cuya pila "crece" hacia direcciones menores se puede simular la instrucción

PUSH con una instrucción MOV con direcciónamiento indexado con preautodecremento a través del registro SP.

Verdadero

43-La instrucción ADD Rn,#3 suma el contenido del registro Rn con el de la posición de memoria 3.

Falso

44-El fragmento de código ensamblador de un microprocesador de 8 bits

**Ids IOBuf ; Apuntar puntero pila a
; ...área mem.intermedia
Idx Count ; Inicializar X-contador
poll Ida a Status; Leer estado en A
bpl poll ; Signo(A)!=1 => repetir
Ida a Data ; Leer dato en A
psh a ; Transferir dato a pila
dex ; Decrementar contador X
bne poll ; Seguir leyendo si X!=0
corresponde a:**

Entrada programada con consulta de estado

Ida a Data -> lectura del puerto de Datos

Ida a Status -> poll (consulta) de estado

45-Siendo EDX=0xf000 y ECX=0x0100, ¿cuál de las siguientes instrucciones tiene como dirección efectiva 0xf400?

xorl (%edx, %ecx, 4), %eax

46-Suponiendo que todos los registros inicialmente contienen el valor 1 y que el destino es el primer

registro, ¿cuál es el valor de r1 tras la ejecución de la siguiente secuencia de instrucciones?

```
mov r1, #4
mov r2, #3
add r3, r1, r1
sub r1, r3, r2
mul r3, r1, r1
5
```

47-En la captación de la instrucción:

en MAR indicamos la dirección donde está la instrucción y en MBR recogemos la instrucción.

48-En el modo mínimo el 8086 genera menos señales que en el máximo, y por tanto depende del

controlador de bus 8288 para generar el conjunto completo de señales de control del bus.

Falso

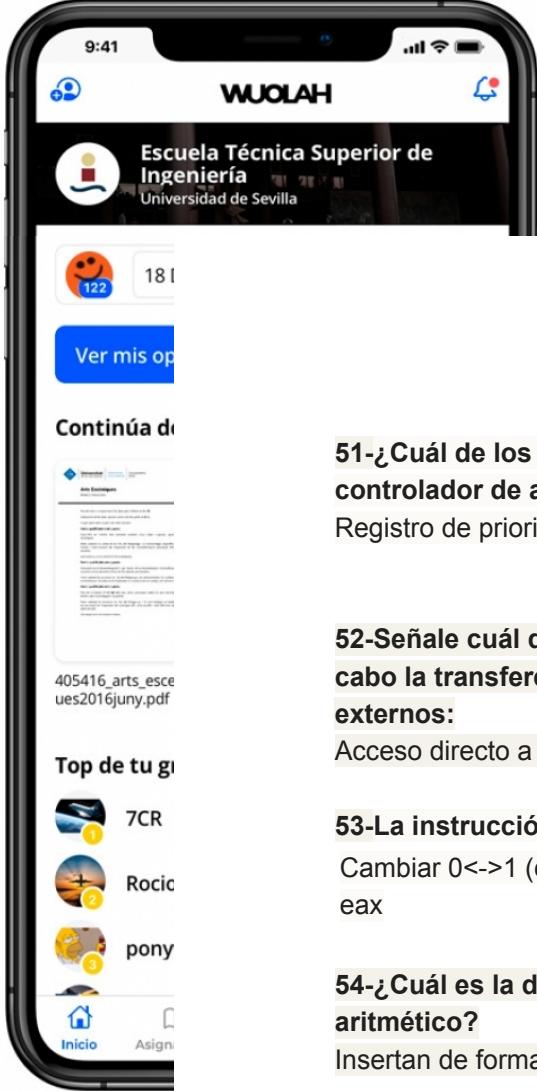
49-Si N es el número de instrucciones máquina de un programa, F es la frecuencia de reloj, y C el

número promedio de ciclos por instrucción, el tiempo de ejecución del programa será:

N·C/F

50-En general, cualquier ordenador debe incluir instrucciones específicas para E/S, por ejemplo IN y OUT.

Falso



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

51-¿Cuál de los siguientes elementos no forma parte de un canal de un controlador de acceso directo a memoria?

Registro de prioridades

52-Señale cuál de las siguientes opciones es una técnica habitual para llevar a cabo la transferencia de datos entre el computador y los dispositivos de E/S externos:

Acceso directo a memoria (DMA)

53-La instrucción xor \$3, %eax tiene como resultado:

Cambiar 0<->1 (complemento a 1 de) los 2 bits menos significativos del registro eax

54-¿Cuál es la diferencia entre los desplazamientos a la derecha lógico y aritmético?

Insertan de forma distinta el bit más a la izquierda

55-Cuando se ejecuta la instrucción ret al final de una subrutina:

la dirección almacenada en la cima de la pila se transfiere al contador de programa

56-Después de ejecutar el siguiente código, ¿qué variables serán igual a 0?

(Suponer ints de 32bits y longs de 64bits)

c y d

En el problema 3.4 sólo se explica que una extensión de tamaño se hace según el fuente sea signed (extensión sgn) o unsigned (ext. con ceros), pero no se explica la sección 2.2.6 (y 2.2.5) en donde se aclara que las operaciones que impliquen a algún unsigned se hacen en unsigned. Evitar esta pregunta en el futuro.

57-En el modo mínimo el 8086 genera menos señales que en el máximo, y por tanto depende del controlador de bus 8288 para generar el conjunto completo de señales de control del bus.

Falso

58-¿Es posible utilizar 4 GB de memoria en un sistema cuya CPU emplea E/S mapeada en memoria, cuyo bus de direcciones es de 32 bits y que tiene al menos un puerto de E/S? Supondremos que no se puede emplear ninguna técnica de extensión del bus de direcciones.

No

59-En un sistema de gestión de interrupciones mediante "polling" o sondeo, el dispositivo que solicita la interrupción envía, junto con la señal de petición de interrupción, su correspondiente vector de interrupción.

Falso

60-Alguna de las siguientes líneas de código sirve para definir una variable entera llamada tam en GNU/as Linux x86. ¿Cuál?

tam: .int .-msg

61-En la práctica "media" se pide calcular la media y resto de una lista de 32 enteros CON signo de 32bits en una plataforma de 32bits sin perder precisión, esto es, evitando desbordamiento. ¿Qué (media : resto) se debe obtener para una lista rellena a -1 salvo el primer elemento, que valiera -31?

(-1 :-30)

62-Si declaramos int val[5]={1,5,2,1,3}; entonces

&val[2] es de tipo int* y vale lo mismo que (void*)val+8

En Sep15 faltaba (void*) y entonces sería falsa por aritmética de punteros

63-Un procesador de 8 bits, ¿a cuántos puertos de E/S podrá acceder?

Depende del método de selección de periféricos que emplee

64-Las interrupciones generadas por el teclado interrumpirán al procesador:
sólo si el procesador tiene activado el indicador de habilitación de interrupciones

65-La codificación Huffman no es la más utilizada debido a que el promedio del código de operación no es mínimo.

Falso

66-¿Qué hace gcc -O1?

Compilar con optimización

67-En un máquina con arquitectura de pila, todas las instrucciones aritméticas tienen dos operandos implícitos: la cima de la pila y el elemento siguiente de la cima de la pila.

Falso

68-Se suelen utilizar PLA en las unidades de control cableadas.

Verdadero

69-Suponga que la micropalabra de una máquina microprogramada tiene 8 bits de ancho y se usan 16 micropalabras diferentes en un microprograma de 256 micropalabras. Si se usa nanoprogramación...
se ahorran bits pero el funcionamiento es más lento.

70-De los siguientes grupos de técnicas de E/S, ¿cuáles están controlados por programa?

E/S programada y E/S mediante interrupciones.

71-¿Cuál de las siguientes instrucciones x86 se puede usar para sumar dos registros y guardar el resultado sin sobrescribir ninguno de los registros originales?

lea

72-La instrucción leave equivale a:

movl %ebp, %esp; popl %ebp

73-En el 8086, si tras una instrucción de comparación CMP A,B aparece una instrucción JC, ésta realiza un salto si A > B, siendo A y B números sin signo.

Falso

74-La instrucción necesaria para cargar 0x07 en %eax es:

movl
\$0x07,%eax

75-Compilar .c→exe (de fuente C a ejecutable) usando sólo as y ld, sin gcc...

No se puede

76-Si varios dispositivos comparten una única línea de solicitud de interrupción:

todas son ciertas

77-¿Cuál es el contenido de una pila al terminar de ejecutarse la siguiente secuencia de operaciones push y pop:

push #1
push #2
push #3
pop a
push #4
pop a
pop a

78-¿Cuál de las siguientes secuencias de instrucciones calcula a=b-a, suponiendo que %eax es a y %ebx es b?

Varias o ninguna de las respuestas anteriores son correctas, no se puede marcar una y sólo una

79-La práctica "parity" debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la siguiente versión de parity6:

```
int parity6(unsigned * array, int len)
{
    int i, result = 0;
    unsigned x;
    for (i=0; i<len; i++){
        x = array[i];
        asm( "mov %[x], %%edx \n\t"
            "shr $16, %%edx \n\t"
            "shr $8, %%edx \n\t"
            "xor %%edx,%%edx \n\t"
            "setp %%dl      \n\t"
            "movzx %%dl, %[x] \n\t"
            : [x] "+r" (x)
            :
            : "edx"
        );
        result += x;
    }
    return result;
}
```

Esta función parity6:

no es correcta; fallaría por ejemplo con array={0,1,2,3}

Caso real, entregado en prácticas. Las tres primeras instrucciones asm se pierden al poner edx a 0 usando xor. Consecuentemente, se activa PF para ajustar a impar, y termina siendo x=1. Es decir, todos los elementos del array contabilizan paridad=1. El array {1,2,4,8} pasa desapercibido, pero {0,1,2,3} debería producir resultado=2<>4.

80-En el contexto del lenguaje máquina, el acrónimo ISA suele referirse a:

Instruction Set Architecture



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

Continúa dí



405416_arts_esc_ues2016juny.pdf

Top de tu grupo

| | |
|--|--------|
| | 7CR |
| | Rocío |
| | pony |
| | Inicio |
| | Asign |

81-El polling consiste en chequear cada fuente de eventos (por ejemplo interrupciones) en algún orden predeterminado.

Verdadero

82-En un microprocesador de 4 bits, una operación en la que el bit 0 de un registro se copia en el acarreo, después el bit 1 se copia en el bit 0, después el bit 2 se copia en el bit 1, y por último el bit 3 se copia en el bit 2, es:

Un desplazamiento aritmético a la derecha.

83-Alguna de las siguientes señales NO es salida de la unidad de control.

¿Cuál? Dirección de la siguiente microinstrucción (bits del campo DIR o Memoria B de Wilkes)

84-La segmentación de cauce...

- a) permite ejecutar varias instrucciones concurrentemente
- b) acelerar la ejecución de un programa
- c) provoca riesgos debido a datos
- ✓ d) todas las respuestas son ciertas

85-El bus AGP permite que los aceleradores gráficos accedan directamente a texturas almacenadas en la memoria principal.

Verdadero

86-¿Qué método de control de acceso directo a memoria es preferible por velocidad (más rápida), economía (coste no prohibitivo) y conveniencia de diseño (compatible con memorias y sistemas actuales)?

Transferencia de bloques o parada de CPU

87-En la práctica "media" se pide sumar una lista de 32 enteros SIN signo de 32 bits en una plataforma de 32 bits sin perder precisión, esto es, evitando perder acarreos. De entre los siguientes, ¿cuál es el mínimo valor entero que repetido en toda la lista causaría acarreo con 32 bits (sin signo)? Se usa notación decimal y espacios como separadores de millares/millones/etc.

1 000 000 000

Se pasa, 1000 millones >> 128M

88-¿Qué método de identificación de la fuente de una interrupción suele ser más económico desde el punto de vista hardware?

La identificación mediante la técnica de sondeo

89-Un computador tiene una memoria de control de 16 Kpalabras de 250 bits, de las que 447 son diferentes. ¿Cuántos bits ahorramos usando nanoprogramación en lugar de microprogramación?

a)

3928652

b)

259206

c)

287935

✓ •d)

ninguno de los resultados anteriores es exacto

90-Las instrucciones máquina que aparecen en segundo lugar en el análisis dinámico de uso de instrucciones son las de bifurcación.

Verdadero

91-Considerar las siguientes declaraciones de estructuras en una máquina Linux de 64-bit.

```
struct RECORD {  
    long value2;  
    int ref_count;  
    char tag[4];  
};
```

```
struct NODE {  
    double value;  
    struct RECORD record;  
    char string[8];  
};
```

También se declara una variable global "my_node" como sigue:

```
struct NODE my_node;
```

¿Cuál es el tamaño de my_node en bytes?

32 bytes

92-La función gettimeofday() en la práctica de la "bomba digital" se utiliza para
Para cronometrar lo que tarda el usuario en introducir la contraseña

93-En un sistema Linux x86-64, ¿cuál de las siguientes expresiones es equivalente a la expresión C $(x[2] + 4)[3]$? Suponer que previamente se ha declarado int **x.

$*((*(x + 2) + 4) + 3)$
sería $x[2][4+3]$, $+4+3 == +7$ sí sería aritmética punteros

94-En las arquitecturas RISC hay...

muchos registros y pocos modos de direccionamiento.

95-En la práctica "media" se pide sumar una lista de 32 enteros CON signo de 32bits en una plataforma de 32bits sin perder precisión, esto es, evitando desbordamiento. ¿Cuál es el valor negativo más pequeño (en valor absoluto) que repetido en toda la lista causaría desbordamiento con 32bits (en complemento a 2)?

0xfbff ffff

96-En el direccionamiento indirecto a través de registro, la dirección efectiva... se encuentra en un registro general del procesador.

97-En 80x86, los parámetros a las subrutinas se pueden pasar:

a)
a través de variables globales

b)
a través de los registros

c)
a través de la pila
✓ d)
todas las anteriores son ciertas

98-La etiqueta del punto de entrada a un programa ensamblador en el entorno de las prácticas 1 a 4 (GNU/as Linux x86) es:

_start

P3 y P4 se redactan en C.

P1 y P2 sí son en ensamblador GNU/as Linux x86.

Incluso en P2 se llega a usar main para ensamblar con gcc ya que usamos printf.

En cualquier caso, las otras opciones son descabelladas

99-Sean un int*a y un int n. Si el valor de %ecx es a y el valor de %edx es n, ¿cuál de los siguientes fragmentos de ensamblador se corresponde mejor con la sentencia C return a[n]?

mov (%ecx,%edx,4),%eax

ret

100-Durante un robo de ciclo DMA el procesador mantiene en alta impedancia el bus de direcciones.

Verdadero.

101-Una cola de precaptación sirve para:

Reducir el efecto de los fallos de cache

102-¿Cuál es el contenido de la pila al terminar de ejecutarse la siguiente secuencia de instrucciones de una arquitectura de pila: push #4; push #7; push #8; add; push #10; sub; mul?

20

103-En el RISC-I, una ventana de registros contiene:

registros de propósito general.

104-En un camino de datos con un solo bus, para realizar la operación de copia de un registro r1 en un registro r2, es decir $r2 \leftarrow r1$, es necesario:

Habilitar la salida triestado del registro r1 y activar la carga del registro r2

105-Se llama bit slice a la operación de desplazamiento de bits que realiza una instrucción SHL o SHR.

Falso

106-De los siguientes grupos de técnicas de E/S, ¿cuáles están controlados por programa?

E/S programada y E/S mediante interrupciones.



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

107-Respecto a MBR y MAR

AR requiere menos señales de control que MBR
en Tema 3 tr.10, sólo una (LoadMAR), frente a 4 de MBR (Load/Enable Mem/Bus)

108-Las instrucciones máquina más usadas según el análisis dinámico son las de bifurcación

Falso

109-Si %edx contiene 0xf000 y %ecx contiene 0x0100, el direccionamiento 0x80(%ecx,%edx,2) se refiere a la posición

0x1e180

110-Señale cuál de las siguientes opciones no es un modo para llevar a cabo la transferencia de datos entre el computador y los dispositivos de E/S externos:

E/S por flanco

111-El hecho de utilizar "polling" no implica que la prioridad de los dispositivos interruptores quede fijada mediante encadenamiento ("daisy-chain").

Verdadero

112-¿En qué pareja de registros están el dato/instrucción que se leerá o escribirá en memoria, y la dirección de memoria?

MBR y MAR

113-Debido al pequeño número de operaciones simples, la sección de control de un procesador RISC puede ser cableada en lugar de microprogramada.

Verdadero

114-Se podría diseñar una CPU microprogramada de manera que la captación y la ejecución de microinstrucciones se solapasen en el tiempo.

Verdadero

115-De entre las siguientes construcciones de flujo de control en lenguaje C, la que se traduce más directamente a lenguaje ensamblador es...

El bucle do-while

116-¿Qué tipo de direccionamiento se usa para el registro destino en la instrucción IA32 add array(%ebx,4), %edx?

Direccionamiento a registro

117-La instrucción test es...

Lo mismo que and, pero no guarda el resultado, sólo ajusta los flags

118-El primer computador electrónico basaba su funcionamiento en:

tubos de vacío

1^a generación

119-¿Cuál de las siguientes afirmaciones es cierta respecto al lenguaje C?

En lenguaje C, al llamar a una subrutina o función se introducen los parámetros en la pila y después se realiza una llamada a la subrutina

suponiendo convención cdecl x86, porque x86-64 usa regs.

120-¿Cuál es la característica tecnológica principal de la segunda generación de computadores?

Los transistores

121-Un procesador cuya instrucción CALL guarda la dirección de retorno en un registro:

No permite llamadas anidadas ni recursivas.

122-Un procesador de 8 bits, ¿a cuántos puertos de E/S podrá acceder?

Depende del método de selección de periféricos que emplee

123-Las instrucciones de salto...

complican el diseño eficiente de los procesadores segmentados.

124-En la técnica de salto retardado:

El compilador puede reorganizar el código para llenar los huecos de retardo con instrucciones útiles

125-En la actualidad todos los microprocesadores utilizan pipeline, tanto RISC como CISC.

Verdadero

126-En una máquina con 32 registros direccionables e instrucciones de 16 bits, es posible codificar 63 instrucciones de dos registros, 31 instrucciones de 1 registro y 32 instrucciones de 0 direcciones.

Verdadero

127-¿Qué nº de modo de funcionamiento permite a la interfaz de periféricos programable 8255 utilizar un bus bidireccional?

2, el modo 2.

128-En una arquitectura RISC típica:

suele usarse segmentación

129-El controlador de DMA programable 8237 puede realizar una operación de acceso directo a memoria en la que se transfiera un bloque de 512 KBytes sin intervención de la CPU una vez comenzada la transferencia.

Verdadero

130-¿Cuál de los siguientes elementos no forma parte del canal de un controlador de acceso directo a memoria?

Registro de prioridades

131-¿Cuál es el contenido de la pila al terminar de ejecutarse la siguiente secuencia de instrucciones de una arquitectura de pila?:

push #4

push #7

add

push #10

sub

a) 1

132-Un procesador está segmentado en las etapas F, D, E, M y W. Cada una de ellas consume un tiempo t. La aceleración ideal (si no hay riesgos) al ejecutar n instrucciones respecto a un procesador no segmentado será:

a) $5n / (4+n)$

$kn/(k+n-1)$, con

k=5

133-La idea de desarrollar máquinas CISC surgió para:

tener instrucciones cercanas al lenguaje de alto nivel.

134-¿En qué método para determinar la dirección de comienzo de una rutina de servicio de interrupción se envia parte de dicha dirección?

Direccionamiento relativo

135-Un procesador con unidad de control microprogramada tiene un generador de direcciones de memoria de microprograma en lugar de otros mecanismos de resolución de dirección efectiva.

Falso

136- Si un computador X ejecuta un programa de 450 millones de instrucciones en 26 segundos y un computador Y tarda 14 segundos en ejecutar ese mismo programa, ¿cuántas veces es más rápido el computador Y que el X?

1.857

137-Suponga la siguiente sentencia asm en un programa:

```
asm(" add (%[a],%[i],4),%[r]"  
    :[r] "+r" (result)  
    :[i] "r" (i),  
    [a] "r" (array)  
);
```

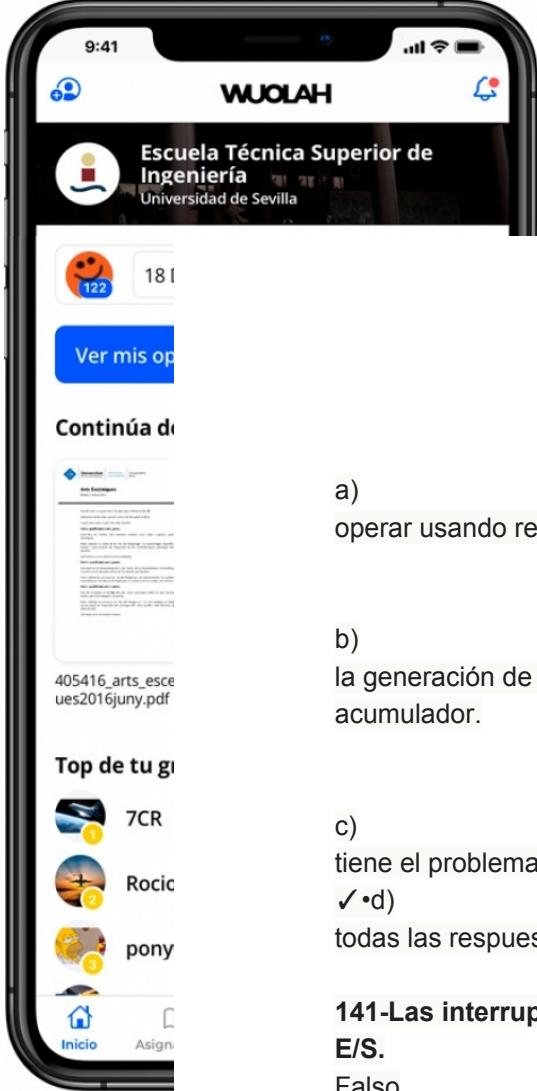
¿Cuál de las siguientes afirmaciones es correcta?

- a) r es una posición de memoria de entrada/salida
- b) el código de retorno de la función asm se fuerza a que esté en la variable result
- c) i es un registro de entrada
- d) a es una posición de memoria de entrada

138-Un computador que utilice únicamente controladores programables 8237 para realizar el DMA por robo de ciclo puede realizar:
más de cuatro transferencias por DMA concurrentes

139-La instrucción máquina DI (Disable Interrupts), conocida como CLI (Clear Interrupt Flag) en x86, se utiliza para desactivar:
Todas las interrupciones enmascarables

140-En una arquitectura de registros (a nivel de lenguaje máquina):



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

a)

operar usando registros es más rápido.

b)

la generación de código resulta más simple que en arquitecturas de pila o acumulador.

c)

tiene el problema de que las instrucciones pueden ser largas.

✓ d)

todas las respuestas anteriores son ciertas.

141-Las interrupciones se utilizan únicamente para realizar operaciones de E/S.

Falso

142-¿Cuál de las siguientes afirmaciones es falsa respecto al lenguaje C (en convención cdecl x86)?

a)

Al llamar a una subrutina o función se introducen los parámetros en la pila y después se realiza una llamada a la subrutina

b)

Los parámetros se introducen en la pila en el orden inverso a como aparecen en la llamada de C, es decir, empezando por el último y acabando por el primero

•c)

Antes de volver de la rutina llamada, el programa en C se encarga de quitar de la pila los parámetros de llamada realizando varios pop

X

d)

Pasar a una función un puntero a una variable se traduce en introducir en la pila el valor de la dirección de memoria donde está almacenada la variable

143-El objetivo del control residual es optimizar el tamaño del microprograma.

Verdadero

144-Uno de los objetivos del uso de direccionamiento relativo al contador de programa en los saltos es posibilitar la escritura de programas relocalizables.

Verdadero

145-En general, el acceso a una palabra no alineada es más lento que a una alineada.

Verdadero

146-El pipeline o segmentación consiste es una técnica para solapar la ejecución de varias instrucciones máquina.

Verdadero

147-El 8086 tiene:

14 registros de 16 bits y un bus de direcciones de 20 bits

148-En una memoria de bytes que contuviera a partir de la posición 0 los valores 1,0,0,0,0xFE,0xFF,0xFF,0xFF, se puede decir que...

a)

Hay una palabra de 16bit big-endian con valor 1 en la posición 0

b)

Hay una palabra de 16bit little-endian con valor 254 en la posición 3

c)

Hay una palabra de 32bit little-endian con valor -1 en la posición 4

d)

Todas las respuestas anteriores son incorrectas

149-Los procesadores comerciales con unidad de control microprogramada suelen almacenar los microprogramas...

en una ROM.

150-El arbitraje estático de un bus es muy simple, pero es poco flexible, y además hay que calcular previamente qué ancho de banda va a requerir cada dispositivo para no bloquear al más rápido.

Verdadero

151-Si el registro %eax contiene el siguiente valor binario:

11111111 10101010 01010101 11110000

¿Cuál será el valor de %eax tras ejecutar la instrucción xorb %al, %al?

11111111 10101010 01010101 00000000

152-¿Cuál de las siguientes tareas no es responsabilidad de un circuito de interfaz o controlador de periféricos sencillo?

Ejecutar el programa de transferencia de información entre el procesador y los dispositivos de E/S

153-¿Cómo se almacenaría como palabra de 32 bits el número -128 en un sistema que utilice el criterio del extremo menor ("little endian")?

a)

posición 0: FF pos.1:FF pos.2: FF pos.3: 00

b)

0:00 1:FF 2:FF 3:FF

c)

0:00 1:01 2: 00 3:80

✓ d)

Ninguna de las anteriores

154-La E/S por DMA requiere la presencia de un circuito controlador de DMA.

Verdadero

155-¿Cuál de los siguientes registros tiene que ser salvaguardado (si va a modificarse) dentro de una subrutina según la convención cdecl para IA32?

ebx

156-Una señal Bus Grant sirve para indicar que se cede el control del bus a un módulo que lo había necesitado.

Verdadero

157-La instrucción INTO del 8086 realiza la entrada de un dato desde un puerto de entrada al registro AL o AX.

Falso

158-Estudiando el listado de una función C presuntamente compilada con gcc en modo 64bit (x86-64), nos dicen que la primera instrucción, movl %eax, (%rdi), carga en EAX el valor adonde apunta el primer argumento.

Está mal, porque EAX no se carga con ningún valor

159-Alguna de las siguientes no es una operación básica de la unidad de control

a)

(leer o escribir) un registro (de / a) memoria

✓ •b)

(guardar o recuperar) un registro (en / de) la pila

c)

transferir un registro a otro

d)

realizar operación ALU y guardar resultado en registro

160-En una máquina little-endian con memoria de bytes y representación en complemento a dos que permite accesos a memoria de tamaño byte (1 B), media palabra (2 B) y palabra (4 B), se almacenan a partir de la posición 0xCAFEBAB0 cuatro palabras con valores -1, -2, -3, -4. ¿Qué se obtendría al consultar la media palabra de la posición 0xCAFEBABE?

-1

los contenidos son

CAFEBAB0: FF FF FF FF

CAFEBAB4: FE FF FF FF

CAFEBAB8: FD FF FF FF

CAFEBABC: FC FF FF FF

las últimas dos posiciones, a partir de CAFEBABC, contienen FF FF, que es -1



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

161-Si usamos una estructura de bus con DMA:

la CPU puede dejar las transferencias entre MP y periféricos en manos de este controlador (DMA) y seguir ejecutando otras instrucciones.

162-La diferencia entre temporización de bus asíncrona y semisíncrona es que en la asíncrona las transferencias ocurren en algún múltiplo de ciclo de reloj, y en la semisíncrona no existe reloj del bus.

Falso

163-Una sentencia en C del tipo "while (test) body;" puede transformarse en código "goto" como:

```
if (!test) goto done;  
loop:  
    body;  
    if (test) goto loop;  
done
```

164-Alguna/s de las ventajas de la E/S mapeada en memoria frente a la E/S aislada o independiente es/son:

El diseño de la CPU es más sencillo.

165-Respecto a las unidades de control nanoprogramadas:

La anchura de la memoria de nanoprograma es la misma que la de memoria de micropograma en un diseño de la misma unidad de control que no usara nanoprogramación.

166-¿Cuál de los siguientes fragmentos es correcto para comenzar un programa en ensamblador que conste de un solo archivo .s?

```
.section .text  
.global _start  
_start:
```

167-Las arquitecturas memoria-memoria tienen la ventaja de emplear instrucciones máquina muy cortas y así los accesos a memoria son mínimos.

Falso

168-¿Cuál de las siguientes parejas de buses contiene sólo buses de tipo serie?

169-De las siguientes instrucciones, ¿cuál utiliza un modo de direccionamiento no implícito?

push

170-En una unidad de control microprogramada con formato de microinstrucciones vertical, un subcampo que deba especificar 16 señales de control codificadas de tal forma que pueda activarse sólo una o ninguna habrá de tener una anchura mínima de

5 bits

171-En un computador cuya pila "crece" hacia direcciones menores se puede simular la instrucción PUSH con una instrucción MOV con direccionamiento indexado con preautodecremento a través del registro SP.

Verdadero

172-El registro RDM (MAR en inglés) contiene la última instrucción o dato leído de memoria o el dato que se va a escribir en memoria.

Falso

173-¿Cuál de los siguientes es el orden correcto en el ciclo de compilación de un programa en lenguaje C? (el fichero sin extensión es un ejecutable):

fich.c → fich.s → fich.o → fich

174-Técnicas que se pueden usar para determinar la causa de una interrupción (señalar la opción incorrecta)

a)

interrupciones vectorizadas

b)

múltiples líneas de interrupción INT1#, INT2#...

c)

consulta de estado, o polling

✓ d)

Línea de reconocimiento INTA#

175-En las instrucciones de salto condicional se suele usar direccionamiento relativo.

Verdadero

176-Para compilar un programa escrito en C en el entorno GNU/Linux se usa el programa:

gcc

177-La tendencia actual y futura en buses de periféricos es pasar de diseños serie a paralelo.

Falso

178-En el fragmento de código

```
804854e:e8 3d 06 00 00 call 8048b90 <main>
8048553:50          pushl %eax
```

la instrucción call suma al contador de programa la cantidad:

0x0000063d

179-En el pseudocódigo usado para representar las microinstrucciones, la expresión “goto f(IR)”:

Salta a una dirección de memoria de control que depende de la instrucción máquina actual.

180-¿Cuál fue el primer procesador de Intel de 64-bits en la familia x86(-64)?

Pentium 4F

181-Un procesador con una unidad de control microprogramada tiene una memoria de control de 256 palabras de 16 bits, de las que 128 son diferentes. ¿Qué ahorro en número de bits obtendríamos si usáramos nanoprogramación?

256 bits

182-En el secuenciamiento de microinstrucciones explícito cada microinstrucción incluye la dirección de la microinstrucción siguiente.

Verdadero

183-La instrucción JGE / JNL provoca un salto si...

OF = SF

basta recordar que "Less" no era un flag solo (es OF^SF)

recordar también que "Below" comprueba CF

184-El direccionamiento directo a memoria utiliza...

un desplazamiento.

185-En un sistema con interrupciones vectorizadas, el dispositivo o interfaz siempre suministra la dirección de la rutina de servicio de interrupción, aunque a esa dirección le falten bits.

Falso

186-El direccionamiento relativo necesita hacer uso de una tabla de direcciones, cada una de las cuales apunta a la base del siguiente dato.

Falso

187-¿Cuál de los siguientes no es un tipo de bus?

Secuencial

opuestos a buses paralelos son los buses serie

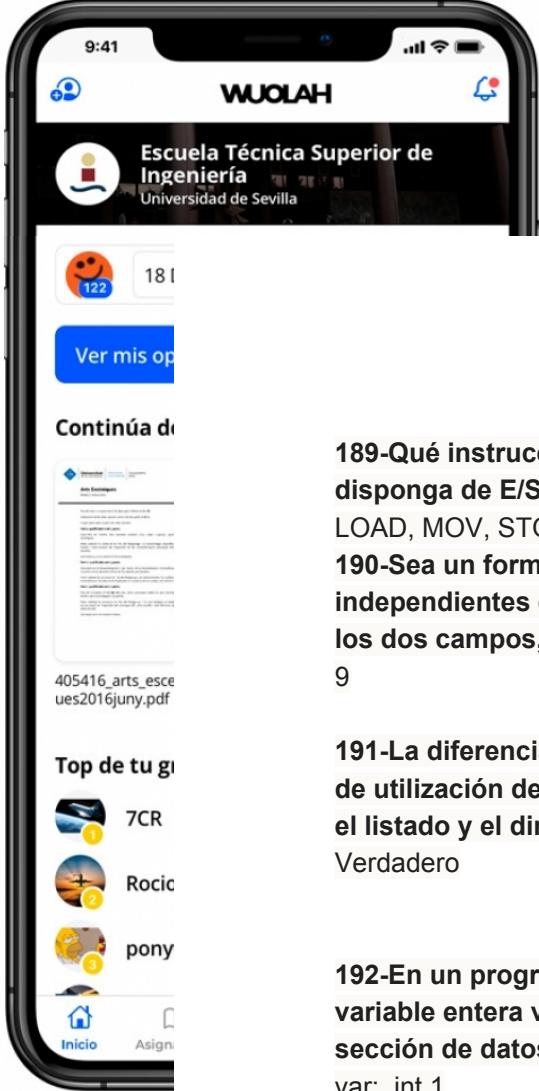
opuestos a programas secuenciales son los programas paralelos

188-La operación aritmética calculada por el programa

```
mov $5, %eax  
mov $3, %ebx  
mov $7, %ecx  
mov $8, %edx  
mul %ebx, %ecx  
add %ecx, %eax  
sub %edx, %eax
```

es:

$$5 + (3 \times 7) - 8$$



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

189-Qué instrucciones son típicas del repertorio de un procesador que sólo disponga de E/S mapeada en memoria?

LOAD, MOV, STORE

190-Sea un formato de microinstrucción que incluye dos campos independientes de 10 bits cada uno. Si se rediseña de modo que se solapen los dos campos, ¿cuántos bits se ahorran en cada microinstrucción?

9

191-La diferencia entre un análisis estático y uno dinámico sobre la frecuencia de utilización de las instrucciones máquina es que el estático recuenta sobre el listado y el dinámico contabiliza las repeticiones debidas a bucles, etc.

Verdadero

192-En un programa en ensamblador queremos crear espacio para una variable entera var inicializada a 1. La línea que hemos de escribir en la sección de datos es:

var: .int 1

193-El espacio direccionable de memoria de un computador depende del diseño del:

Bus de direcciones

194-Si almacenamos según el criterio little-endian la palabra de 64 bits 0xFACEB00C a partir de la dirección 0xCAFEBAE, el byte 0xCE quedará almacenado en la dirección:

0xCAFEBAC0

195-Un fragmento de una “bomba” desensamblada es:

```
0x0804873f: call 0x8048504 <scanf>
0x08048744: mov 0x24(%esp),%edx
0x08048748: mov 0x804a044,%eax
0x0804874d: cmp %eax,%edx
0x0804874f: je 0x8048756 <main+230>
0x08048751: call 0x8048604 <boom>
0x08048756: ...
```

La contraseña/clave en este caso es...

el entero almacenado a partir de la posición de memoria 0x804a044 se hace mov 0x0804a044, %eax justo antes de cmp %eax, %edx, en donde %edx está relacionado con scanf

196-En el 8086, la secuencia de instrucciones ADD SP,10 seguida de RET es equivalente a RET 10.

Falso

197-Si la variable val está almacenada en ebx y la variable x está almacenada en eax, la sentencia val ^= x; se puede traducir a ensamblador como:
gunas de las ventajas de la E/S mapeada en memoria frente a la E/S aislada o independiente son:

xorl %eax,%ebx

198-La primera generación de computadores se caracteriza por el uso de:
Tubos de vacío.

199-La práctica “parity” debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la siguiente versión de parity6:

```
int parity6(unsigned* array, int len){  
    int i,j,res=0;  
    unsigned x;  
    for (i=0; i<len; i++){  
        x=array[i];  
        asm("\n"  
            "mov %[x],%%edx \n\t"  
            "shr $16, %%edx \n\t"  
            "xor %%edx,%[x] \n\t"  
            "mov %[x],%%edx \n\t"  
            "mov %%dh, %%dl \n\t"  
            "xor %%edx, %[x]\n\t"  
            "setpo %%cl \n\t"  
            "movzx %%cl, %[x]"  
            :[x] "+r" (x)
```

```
:  
    :"edx","ecx"  
);  
    res+=x;  
}  
return res;  
}
```

La sentencia asm() del listado anterior tiene las siguientes restricciones

a)

ninguna

b)

arquitectura de 32 bits

x

c)

dos entradas y una salida

d)

un registro y dos sobrescritos (clobber)

el registro es [x] "+r" y los sobrescritos son "edx","ecx"

200-En la E/S controlada por interrupciones la CPU no tiene que ejecutar un programa para realizar la transferencia de datos.

Falso

201-Hay procesadores que desactivan automáticamente las interrupciones sólo durante la ejecución de la primera instrucción de la ISR, con lo cual es inevitable que se produzcan interrupciones por parte de otros dispositivos de forma anidada

Falso

202-En los procesadores CISC gran parte del área del chip se consume en la unidad de control.

Verdadero

203-¿Cuál es la diferencia entre las instrucciones mov y lea?

mov referencia (accede) la posición indicada, mientras que lea no lo hace

204-Un sistema no segmentado tarda 10 ns en procesar una tarea. La misma tarea puede ser procesada en un cauce (pipeline) de 5 segmentos con un ciclo de reloj de 4 ns. Cuando se procesan muchas tareas, la ganancia máxima de velocidad que se obtiene se acerca a:

2,5

205-Considerar las siguientes declaraciones de estructuras en una máquina Linux de 64-bit.

```
struct RECORD {  
    int value2;  
    short ref_count;  
    char tag[10];  
};  
  
struct NODE {  
    long value;  
    struct RECORD record;  
    char string[8];  
};
```

También se declara una variable global "my_node" como sigue:

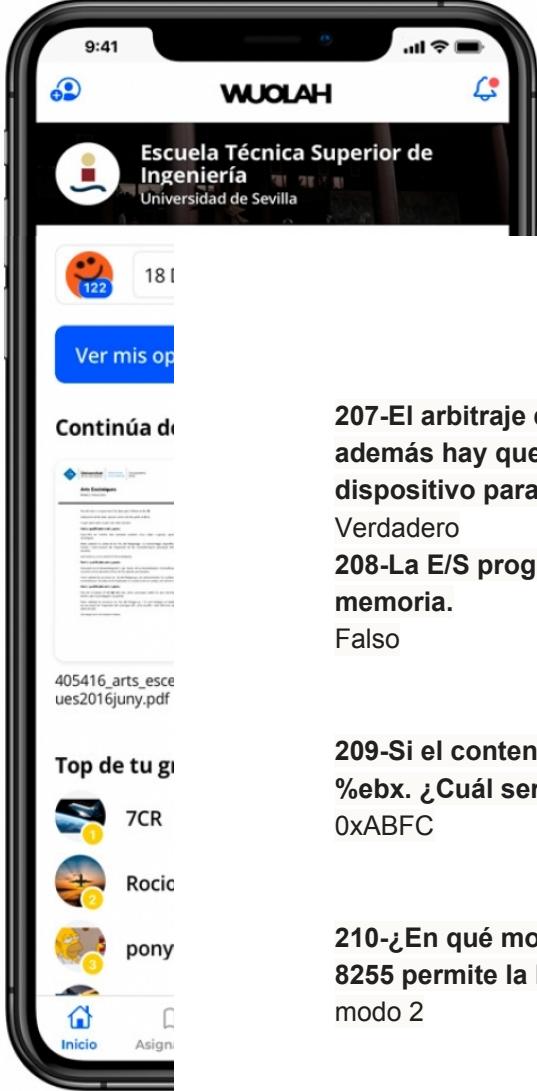
```
struct NODE my_node;
```

Si la dirección de my node es 0x600940, ¿cuál es el valor de &my node.record.tag[1]?

0x60094f

206-Un computador tiene una memoria de control de 640 palabras de 70 bits, de las que 280 son diferentes. ¿Qué ahorro en número de bits obtendríamos si usáramos nanoprogramación en lugar de microprogramación?

19440



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

207-El arbitraje dinámico de un bus es muy simple, pero es poco flexible, y además hay que calcular previamente qué ancho de banda va a requerir cada dispositivo para no bloquear al más rápido

Verdadero

208-La E/S programada con consulta de estado ha de ser mapeada en memoria.

Falso

209-Si el contenido de ESP es 0xAC00 antes de ejecutar la instrucción push %ebx. ¿Cuál será su contenido tras ejecutarla?

0xABFC

210-¿En qué modo de funcionamiento la interfaz de periféricos programable 8255 permite la E/S con bus bidireccional?

modo 2

211-Utilizando E/S programada y como modo de direccionamiento selección lineal, ¿cuántos periféricos podrían conectarse a un 8086? Recordar que el 8086 disponía de E/S separada, con bus de direcciones de 20 bits para memoria y de 16 bits para E/S.

16 periféricos

212-Un sistema no segmentado tarda 50 ns en procesar una tarea. La misma tarea puede ser procesada en un cauce ("pipeline") de 5 segmentos con un ciclo de reloj de 50 ns. Cuando se procesan muchas tareas, la ganancia de velocidad que se obtiene se acerca a 5.

Falso

213- ¿Es cierto que la consulta de estado permite averiguar el estado de un periférico y alterar la prioridad de los periféricos?

Sí, permite averiguar el estado de un periférico y también alterar la prioridad de los periféricos

214-Las siguientes afirmaciones sugieren que el tamaño de varios tipos de datos en C (usando el compilador gcc) son iguales tanto en IA32 como en x86-64. Sólo una de ellas es FALSA. ¿Cuál?

a)

El tamaño de un puntero es 4 bytes

b)

El tamaño de un int es 4 bytes

c)

El tamaño de un double es 8 bytes

d)

El tamaño de un short es 2 bytes

215-Un computador que utilice únicamente controladores programables 8237 para realizar el DMA puede realizar más de cuatro transferencias por DMA concurrentes.

Verdadero

216-La conexión de las salidas de tres registros hacia un bus común en el camino de datos puede realizarse usando...

dos multiplexores de 2 a 1

217-¿Cuál de los siguientes grupos de instrucciones IA32 sólo modifican los indicadores de estado sin almacenar el resultado de la operación?

CMP, TEST

218-¿Cuál de los siguientes grupos de instrucciones podrá pertenecer a un procesador con E/S mapeada en memoria?

LOAD, MOV, STORE

219-Al traducir de lenguaje C a ensamblador, gcc en máquinas Linux/IA-32 almacena (reserva espacio para) una variable “var” local a una función “fun” en una dirección de memoria referenciable (en lenguaje ensamblador) como: -k(%ebp), siendo k un número constante positivo relativamente pequeño

220-¿En qué técnica para determinar la dirección de comienzo de la rutina de servicio de interrupción se fija dicha dirección en los circuitos de la CPU?
Direcciones fijas.

221-Si R0=2, R1=5 y M[3]=3 ¿Qué valor toman R0, R1 y M[3] tras ejecutarse la instrucción XOR 1h[R0],R1?
R0=2 , R1=5 , M[3]=6

222-El sufijo l de la instrucción movl significa:

Que la instrucción trabaja con operandos de 32 bits (long word).

223-Las ventanas de registros solapadas se utilizan para ahorrar registros en el diseño del procesador.

Falso

224-¿Cuál de los siguientes lenguajes no permite el paso de parámetros por referencia?

C

225-Teniendo en cuenta que el tiempo de ejecución de un programa es proporcional al producto número de instrucciones del programa * número medio de ciclos de reloj por instrucción, la filosofía CISC pretende reducir el tiempo de ejecución reduciendo el número de instrucciones del programa y la RISC reduciendo el número medio de ciclos de reloj por instrucción.

Verdadero

226-En el 80x86, la instrucción "JNE etiqueta" es equivalente a "JNZ etiqueta".

Verdadero

227-Tras comparar números con signo, para saltar si menor se utilizan:

El overflow y el signo

228-Dada la siguiente declaración en lenguaje C, una estructura de este tipo podría ocupar, bien sea en un sistema Linux IA32 o bien en uno x86-64, un total de...

```
struct a{  
    int i;  
    double d;  
    char c;  
    short s; };
```

24 B

4 B relleno entre i y d, 1B entre c y s, 4B relleno tras s.

229-Cuál de las siguientes características es posterior a la segunda generación de computadores?

RISC.

230-Si usamos una estructura de bus con DMA:

la CPU puede dejar las transferencias entre MP y periféricos en manos de este controlador (DMA) y seguir ejecutando otras instrucciones.

231-La especificación de un bus a nivel eléctrico debe incluir las siguientes partes:

Alimentación, impedancia, nivel de señal,...

232-¿Cuál de las siguientes afirmaciones es cierta?

En E/S independiente, las instrucciones de acceso a memoria suelen ser más largas que las de E/S

233-Un conjunto de microoperaciones compatibles constituye una microinstrucción, y una secuencia de microinstrucciones es un micropograma.

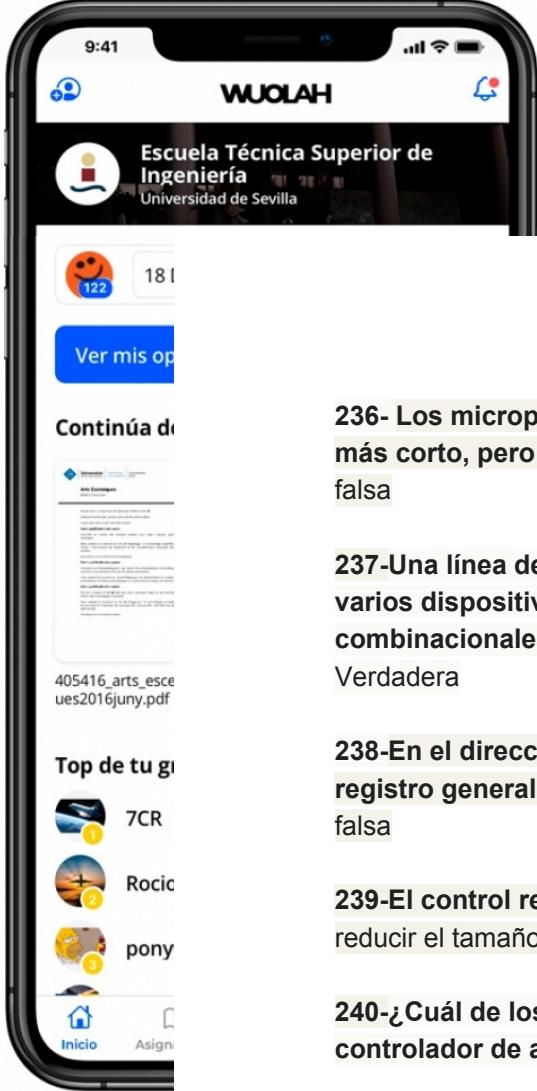
Verdadero

234-¿Cuáles de las siguientes instrucciones utilizan sólo direccionamiento implícito?

lahf, movs

235-La instrucción cmovb %edx, %eax

copia en %eax el contenido de %edx si el indicador de acarreo es 1
"below" equivale a CF



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

236- Los microprocesadores RISC puede ser implementados en un tiempo más corto, pero requieren mayor área de silicio que los CISC.
falsa

237-Una línea de interrupción organizada en colector abierto permite conectar varios dispositivos de manera sencilla sin necesidad de circuitos combinacionales (puertas OR).

Verdadera

238-En el direccionamiento directo, la dirección efectiva se encuentra en un registro general del procesador.

falsa

239-El control residual se utiliza para:

reducir el tamaño de la memoria de control

240-¿Cuál de los siguientes elementos no forma parte de un canal de un controlador de acceso directo a memoria?

a) Registro contador

b) Registro de órdenes

c) Registro de dirección

**d) Registro de prioridades

241-En la E/S mapeada en memoria, el procesador (CPU) no distingue entre accesos a memoria y accesos a los dispositivos de E/S.
veradero

242-Si varios dispositivos comparten una única línea de solicitud de interrupción se puede usar "polling" para identificar el origen de una interrupción.

veradero

243-Una operación de transferencia READ-MODIFY-WRITE es ideal para implementar la instrucción TEST AND SET.

veradero

244-La convención de llamada Linux/GCC x86-32 considera, respecto a convenios de uso de registros:

3 registros salva-invocante, 3 registros salva-invocado, y 2 especiales

245-¿Qué valor contendrá el registro edx tras ejecutar las dos instrucciones siguientes?

`movl $-1, %edx
movb $1, %dl`

11111111 11111111 11111111 00000001

246-En la realización de la práctica de la bomba digital, una parte del código máquina es el siguiente:

```
0x080486e8 <main+120>:  call  0x8048524 <strcmp>  
0x080486ed <main+125>:  test   %eax,%eax  
0x080486ef <main+127>:  je    0x80486f6 <main+134>  
0x080486f1 <main+129>:  call   0x8048604 <boom>
```

¿Cuál de los siguientes comandos cambiaría el salto condicional por un salto incondicional?

`set *(char*)0x080486ef=0xeb`

247-El direccionamiento inmediato no es más lento que el direccionamiento directo

verdadero

248-Supongamos dos CPU con bus de direcciones de ancho idéntico. Si una de ellas emplea E/S independiente y la otra mapeada en memoria, ¿cuál podrá acceder a una mayor cantidad de memoria?

La CPU con E/S independiente.

249-Sea un formato de microinstrucción que incluye dos campos independientes de 9 bits cada uno. Si se rediseña de modo que se solapen los dos campos, ¿cuántos bits se ahorran en cada microinstrucción?

8

250- ¿Cuál de las siguientes afirmaciones es incorrecta?

a) El repertorio de instrucciones es el conjunto de operaciones que es capaz de interpretar la unidad de control.

b) El repertorio de instrucciones debe ser capaz de realizar una tarea en un tiempo finito.

c) El modo de direccionamiento permite determinar un operando o la ubicación del operando.

***d) Los operandos siempre están almacenados en memoria.

251- ¿De qué depende el tamaño del contador de programa?

a) de la longitud del código de operación

b) del ancho del bus de datos

c) el tamaño no importa

***d) ninguna de las anteriores es cierta

252- El número -12 se almacenará en complemento a 2 en el registro %eax como:

0xFFFFFFFF4

253- En el controlador de DMA 8237 los registros de dirección de memoria y contador de bytes están duplicados para cada canal para que en el modo de autoinicio el circuito sea capaz de recordar los valores originales de esos registros.

Verdadero

254- Un sistema no segmentado tarda 200 ns en procesar una instrucción. Las instrucciones pueden ser procesadas en un cauce segmentado de 20 etapas con un ciclo de reloj de 12 ns. Cuando se procesan muchas instrucciones, la máxima ganancia de velocidad que podría obtenerse se acerca a:

16,67

255- El punto de entrada de un programa ensamblador en GNU/as Linux x86 se llama

_start

256- Como parte del proceso de compilación de una aplicación en lenguaje C, enlazar .o → .exe (de objeto proveniente de fuente C a ejecutable) usando sólo as y ld, sin gcc...

Basta usar ld, con los modificadores de gcc que corresponda, y añadiéndole el runtime de C

257- En los modos de direccionamiento del tipo Desplazamiento (Base, Índice, Factor Escala), puede usarse como desplazamiento, cualquier constante de 1, 2 o 4 bytes (incluso el nombre de una

variable, por su dirección)

258- Un sistema no segmentado tarda 10 ns en procesar una instrucción. Las instrucciones pueden ser procesadas en un cauce (pipeline) de 5 segmentos con un ciclo de reloj de 4 ns. Cuando se procesan muchas instrucciones, la ganancia máxima de velocidad que se obtiene se acerca a:

2,5

259-Indique cuál de las siguientes características no es cierta en el direccionamiento indirecto a memoria a través de memoria:

La instrucción contiene la dirección de memoria exacta en que se encuentra el objeto.

260- En la práctica “media” se pide sumar una lista de enteros *con* signo de 32 bits en una plataforma de 32 bits sin perder precisión, esto es, evitando overflow. ¿Cuál es el menor valor positivo que repetido en los *dos* primeros elementos de la lista causaría overflow con 32 bits al realizar la suma de *esos dos* primeros elementos de la lista?

0x4000 0000

261- Las interrupciones no se utilizan únicamente para realizar operaciones de E/S.

Verdadero

262- Si una pila crece hacia direcciones de memoria decrecientes, una instrucción PUSH utiliza direccionamiento indexado con postautodecremento, siendo SP el registro índice.

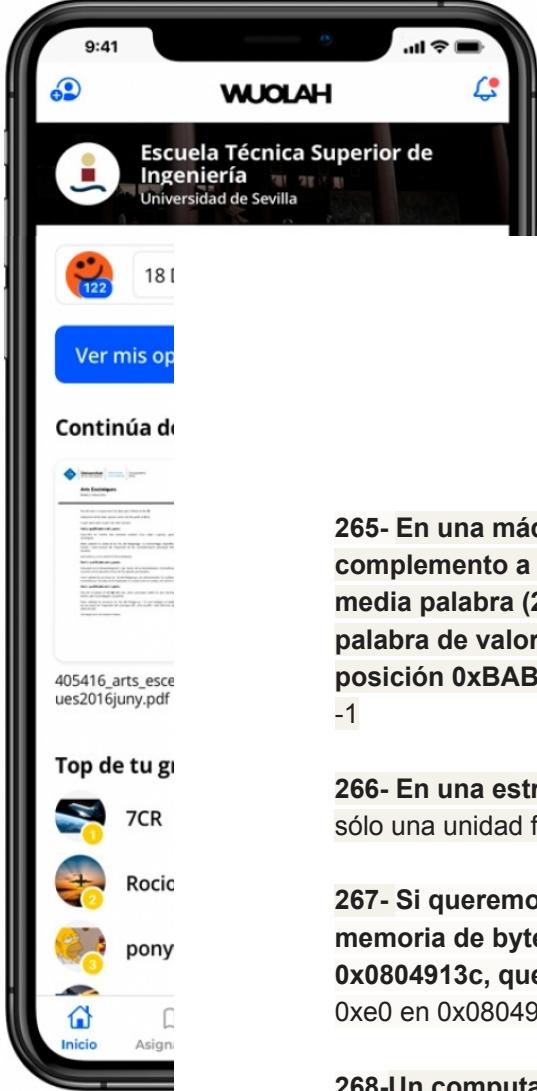
falsa

263- La técnica de sondeo, escrutinio o "polling"...

- a)Se utiliza para identificar el destino de una interrupción
- ***b)Permite establecer un mecanismo de asignación de prioridades a los distintos dispositivos
- c)En caso de utilizarse, es necesario emplear varias líneas para que los dispositivos soliciten una interrupción
- d)Todas las respuestas anteriores son falsas

264- Cuál de las siguientes características es típica de la microprogramación horizontal?

Ninguna o escasa codificación.



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

265- En una máquina little-endian con memoria de bytes y representación en complemento a dos que permite accesos a memoria de tamaño byte (1B), media palabra (2B) y palabra (4B), si se almacena en la posición 0xBABC una palabra de valor -2, ¿qué se obtendría al consultar la media palabra en la posición 0xBABE?

-1

266- En una estructura de computador de bus único (bus del sistema): sólo una unidad funcional puede tener el control del bus en cada momento

267- Si queremos almacenar la palabra de 64bits 0x0000001f ffffffe0 en una memoria de bytes según la convención little-endian a partir de la posición 0x0804913c, quedará

0xe0 en 0x0804913c y 0x1f en 0x08049140

268- Un computador con 13 líneas de direcciones utiliza E/S mapeada a memoria. Si se supone que cada uno de los periféricos ocupa 4 direcciones y que el número de periféricos que se planea conectar es de 2^{10} , ¿qué tamaño de memoria admite el computador?

2^{12} palabras

269- La segunda generación de computadores se caracteriza tecnológicamente por la utilización del transistor.

V

270- Son funciones de la unidad de control:

el secuenciamiento de las instrucciones máquina

271- La ecuación básica de rendimiento calcula

cuánto tiempo tarda en ejecutarse un programa concreto conociendo su número de instrucciones y el número de etapas (promedio) y la frecuencia del procesador

272- ¿Cuál de las siguientes afirmaciones es cierta?

- a) la unidad de control necesita como entrada el registro contador de programa para saber cuál es la instrucción que debe ejecutar a continuación realmente la UC copia PC en MAR, y lo que lee en MDR (captación) lo lleva a IR que sí es entrada a la UC (decodificación)
- b) la arquitectura von Neumann de los computadores tradicionales consiste en tener almacenados los datos separados de las instrucciones en memorias distintas vimos en [T4.2EjmSeg] tr.12 que tener memoria de instrucciones separada de la de datos es una variante llamada "arquitectura Harvard", llamándose "arquitectura Princeton" la original
https://en.wikipedia.org/wiki/Von_Neumann_architecture
https://en.wikipedia.org/wiki/Modified_Harvard_architecture
- ✓ • c) el registro de estado (flags) es un registro de propósito específico cuyo contenido puede ser visto directa o indirectamente por el usuario mediante el uso de ciertas instrucciones específicas aunque no hemos visto pushf y popf, por eliminación ésta es la respuesta
- d) el registro de direcciones de memoria es un registro de propósito general que puede contener tanto direcciones como datos
MAR no está disponible al programador, lo usa la UC para indicar la dirección de memoria a leer/escribir

273- La 5^a generación de computadores comenzó aproximadamente en 1971 con el primer microprocesador.

F

274-¿Cuántos bits hacen falta como mínimo para implementar tres niveles de inhibición de interrupciones (general, nivel y máscara) en un sistema con cuatro niveles de interrupción?

7

275-Un circuito controlador de interrupciones permite resolver prioridades entre las fuentes de interrupción conectadas a él.

Verdadero

276-En un sistema de 32bits, ¿cuál de las siguientes expresiones C es equivalente a la expresión $(x[2] + 4)[3]$?

(Asumir que x se ha declarado como int **x. Recordar que C usa aritmética de punteros. Notar que muchos de los paréntesis no son necesarios, sólo se han añadido para evitar confusiones por precedencia de operadores)

$\ast(\ast(x + 2)) + 7$

277-Alguna de las siguientes técnicas NO es de utilidad para determinar la causa de una interrupción

a)

Línea de reconocimiento INTA#

b)

Interrupciones vectorizadas

x

c)

Múltiples líneas de interrupción INT1#, INT2#...

f

d)

Consulta de estado, o polling

278-¿A qué tipo de interrupciones corresponde la forma de determinar la dirección de comienzo de una rutina de interrupción en la que se envía una instrucción de bifurcación completa?

Interrupciones vectorizadas

279-El programa ensamblador siempre resuelve todas las referencias usadas en el fichero fuente.

Falso

280-La instrucción TEST_AND_SET comprueba si la posición de memoria especificada como operando vale 0 y a continuación la pone a 1, todo ello sin posibilidad de interrupción.

Verdadero

281-En una arquitectura de registros de propósito general (a nivel de lenguaje máquina):

a)

operar usando registros es más rápido.

b)

la generación de código resulta más simple que en arquitecturas de pila o acumulador.

c)

se evita el cuello de botella (por ejemplo, pila, o acumulador) que otras arquitecturas presentan al evaluar expresiones aritméticas complejas

✓ d)

todas las respuestas anteriores son ciertas.

**282-¿Cuál de las siguientes direcciones NO está alineada a double (8-byte)?
(Al no poder escribir el 2 como subíndice, aclaramos que ")2" indica binario)**

1110110101110100)2



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

283-Considerar la declaración C

`int array[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};`

Suponer que el compilador tiene la dirección de array en el registro %ecx.

¿Cómo se movería el valor array[3] al registro %eax? Asumir que %ebx es 3.

`movl (%ecx,%ebx,4),%eax`

284-Para traducir una construcción if-then-else de lenguaje C a lenguaje ensamblador, gcc utiliza generalmente

un salto condicional, según la condición opuesta a la del código C, y otro salto incondicional

285-Según el convenio little-endian, encontraremos el byte más significativo en una dirección menor.

Falso

286-Para conectar las salidas de dos registros hacia un bus común en el datapath...

se pueden usar dos buffers triestado.

287-Las operaciones elementales de transferencia de datos en E/S por interrupciones deben ser realizadas por programa.

Verdadero

288-Cuanto más horizontal es la microprogramación más largas son las microinstrucciones.

Verdadero

289-¿Qué técnica de E/S requiere menos atención por parte del procesador?

E/S mediante acceso directo a memoria

290-¿Qué técnica de E/S consume menos tiempo del procesador?

E/S mediante DMA

291-Una instrucción máquina puede desglosarse en las siguientes operaciones elementales:

sp := sp - 1; m[sp] := pc; pc := x

llamada a subrutina

292-EI 8255 es un interfase paralelo de periféricos que puede utilizarse para E/S programada, pero también puede generar interrupciones.

Verdadero

293-En 8086, los parámetros a las subrutinas se pueden pasar:

a)

a través de variables globales

b)

a través de los registros

c)

a través de la pila

✓ d)

todas las anteriores son ciertas

294-En arbitraje por "daisy-chain" la señal BUS_REQUEST es compartida por los maestros potenciales.

Verdadero

295-Las instrucciones JB y JNAE del Pentium provocan un salto si...

CF == 1

296-Una instrucción de llamada a subrutina es más compleja en general que una de salto.

Verdadero

297-¿Cuál de las siguientes afirmaciones acerca del daisy-chain es cierta?

a) Es
incompatibl
e con la
técnica de
sondeo o
polling

- b) Todos los componentes se conectan directamente y con igual prioridad al procesador o gestor de interrupciones
- c) Los componentes se comportan de forma cooperativa: sólo al de mayor prioridad se le concede la interrupción o se apodera del bus de comunicaciones
lo del "bus de comunicaciones" no se explica en las transparencias actuales - el daisy-chain se puede usar también como método de arbitraje para conceder el bus - las señales suelen llamarse BR/BG (Bus request/grant) en lugar de INTR/INTA (IRQ request/acknowledge)
- d) Los componentes están conectados todos con todos y un gestor centralizado decide la prioridad

298-¿En qué modo del transferencia el controlador de DMA programable 8237 no tiene en cuenta el registro contador?

Transferencia por demanda

299-¿Cuál afirmación es FALSA en arquitecturas x86-64?

- a) El tamaño de las posiciones de memoria es 64 bits ←Esta es la falsa
- b) El tamaño de un double es 64 bits
- c) El tamaño de los registros es 64 bits
- d) El tamaño de un puntero es 64 bits

300-En la práctica de la bomba, el tercer ejercicio consistía en usar un editor hexadecimal para crear un ejecutable sin “explosiones”. Para saber qué contenidos del fichero hay que modificar, se puede utilizar... (marcar la opción *falsa*)

hexedit

el enunciado implícitamente indica que no basta con hexedit. Aún así hubo quien la dejó en blanco (pocos) y quien falló (50%)

301-En los computadores con arquitectura Von Neumann los programas están en memoria principal (o caché) cuando se ejecutan.

Verdadero

302-La instrucción seta %al (seta significa "set if above"):

Pone AL a 1 si CF=0 y ZF=0

303-¿Cuál de las siguientes afirmaciones acerca del concepto de interrupción no es cierta?

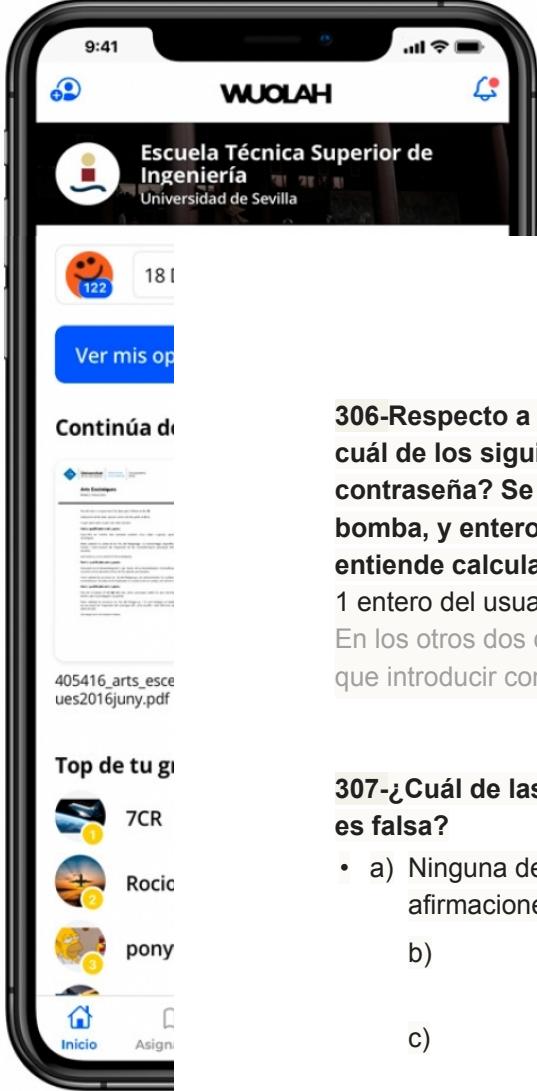
- a) Es una bifurcación normalmente externa al programa en ejecución
- b) Su objetivo es reclamar la atención del procesador
- c) Solicita que se ejecute un programa específico para tratarla
-
- d) Ninguna de las anteriores <-Esta

304-En las instrucciones aritméticas con dos operandos de un procesador con arquitectura de pila, los dos operandos, implícitos, son la cima de la pila y el elemento siguiente de la cima de la pila.

Verdadero

305-La principal desventaja de la nanoprogramación es que requiere un programa ensamblador muy complejo.

Falso



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

306-Respecto a las bombas estudiadas en la práctica "bomba digital", ¿en cuál de los siguientes tipos de bomba sería más difícil descubrir la contraseña? Se distingue entre enteros definidos en el código fuente de la bomba, y enteros solicitados al usuario mediante scanf(). Por "procesar" se entiende calcular el n-ésimo elemento de la serie de Fibonacci.

1 entero del usuario se procesa, y se compara con el entero del fuente

En los otros dos casos usando el debugger se puede ver el resultado literal que hay que introducir como contraseña, parando justo antes de comparar

307-¿Cuál de las siguientes afirmaciones acerca del concepto de interrupción es falsa?

- a) Ninguna de las afirmaciones es falsa <-ESTA
- b) Solicita que se ejecute un programa específico para tratarla
- c) Su objetivo es reclamar la atención del procesador
- d) Es una bifurcación normalmente externa al programa en ejecución

308-Respecto a la convención de llamada usada en Linux/gcc:

Hay registros que pueden ser modificados libremente por las subrutinas, y otros que, si se modifican, se deben restaurar posteriormente. Y también hay registros especiales

309-¿Cuál de las siguientes afirmaciones es cierta?

- a) Ninguna de las otras respuestas es cierta
- b) Los estándares ATA/IDE, SCSI y Firewire (IEEE 1394) definen buses de funcionamiento paralelo a diferencia del estándar USB que define un bus de funcionamiento serie
- c) El bus AGP se utiliza para conectar tarjetas gráficas y controladoras de disco
- d) Los buses PC XT, AT/ISA, MCA, EISA, VLB, PCI y AGP son buses de placa madre de PC <-ESTA

310-Un 8086 necesita al menos 1 KB de memoria RAM para funcionar correctamente.

Falso

311-¿Cuál de las siguientes afirmaciones acerca de las interrupciones en el PC es cierta?

- a) Ninguna de las otras respuestas es cierta **<-ESTA**
- b) No todas las interrupciones se pueden generar por software
- c) Cada vector de interrupción es una palabra de 16 bits
- d) Existen 1024 vectores de interrupción

312-Respecto a la diferencia entre dispositivos activos y pasivos en un bus podemos decir:

- a) Los dispositivos pasivos siempre son esclavos
- ✓ •
- b) Los dispositivos activos pueden actuar como maestros
 - c) Las respuestas a y b son ciertas
 - d) Las respuestas a y b son falsas

**313-Respecto a registros de propósito general (RPG), el 80386 tiene:
8 registros de 32 bits**

314-Suponga una memoria de microprograma de n microinstrucciones de w bits cada una. Suponiendo que de esas n microinstrucciones, hay 2^m distintas, el ahorro en bits si se utiliza nanoprogramación es $n \cdot w - (n \cdot m + 2^m \cdot w)$.

✓

315- Sobre las técnicas de transferencia en operaciones de E/S:

- a) Pueden ser controladas por programa o por hardware
- b) Si se emplea E/S programada puede hacerse con consulta de estado o sin consulta de estado
- c) En el caso de utilizar E/S mediante DMA hace falta emplear un controlador de DMA
- d) Todas las respuestas anteriores son ciertas

316-Un 8086 puede funcionar correctamente con 1 MB de memoria ROM.

V

317-En IA32, tras dividir 0x640000 (64 bits) entre 0x8000 (32 bits), el resultado será:

0xC8 en EAX y 0x0 en EDX

318-La práctica “parity” debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la siguiente versión de parity5:

```
int parity5(unsigned* array, int len){
    int i,j,res=0;
    unsigned x;
    for (i=0; i<len; i++){
        x=array[i];
        for (j=sizeof(unsigned)*4;
             j>0; j=j/2){
            x^=x>>j;
        }
        x = x & 0x1;
        res+=x;
    }
    return res;
}
```

Esta función parity5:

produce siempre el resultado correcto

solemos escribir res+=x&0x1, en lugar de ponerlo en 2 sentencias C

319-El registro MBR...

contiene el valor que va a ser almacenado en la memoria, o bien se usa para recibir un valor procedente de la memoria

320-Un sistema no segmentado tarda 20 ns en procesar una instrucción. Las instrucciones pueden ser procesadas en un cauce (pipeline) de 4 segmentos con un ciclo de reloj de 5 ns. Cuando se procesan muchas instrucciones, la ganancia máxima de velocidad que se obtiene se acerca a:

4

321-Si queremos almacenar la palabra de 16 bits 8965h en memoria según "little-endian", quedará almacenada a partir de la posición 8600h como:
en el byte 8600h se guarda 65h y en el byte 8601h se guarda 89h

322-En las últimas generaciones de computadores la mejora de prestaciones viene dada por:

avances en tecnología y avances en la estructura y arquitectura del computador.

323-Los arrays bidimensionales en lenguaje C se almacenan en orden... mayor-de-fila" (row-major)

324-La secuencialidad es la forma de direccionamiento implícita en el flujo de ejecución de un programa.

v

325-Un controlador de DMA suele ser programado con la siguiente información relativa a una operación de E/S:
tipo de operación, tamaño de bloque a transferir, dirección inicial de memoria

326-Una implementación diferencial del bus SCSI permite mayor distancia o velocidad que una "single-ended".

v

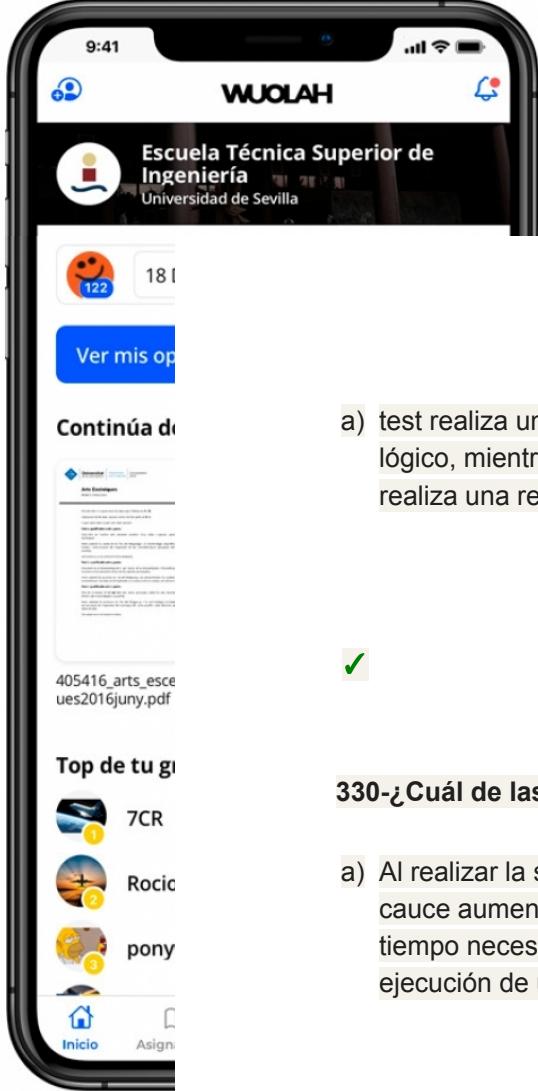
327-Más de la cuarta parte de las instrucciones máquina que se ejecutan en un programa típico son de movimiento o transferencia de datos.

v

328-A qué tipo de interrupciones pertenecen las condiciones de tiempo real y los fallos hardware?

No enmascarables

329-La diferencia entre las instrucciones test y cmp consiste en que



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

Continúa dí

- a) test realiza una operación and lógico, mientras que cmp realiza una resta
- b) test modifica sólo los flags lógicos (ZF,SF) mientras que cmp modifica los aritmético-lógicos (ZF,SF,CF,OF)
- c) ambas respuestas son correctas
- d) ambas respuestas son incorrectas

330-¿Cuál de las siguientes afirmaciones es cierta?

- a) Al realizar la segmentación de cauce aumenta en general el tiempo necesario para la ejecución de un programa
- b) Debido a que pueden existir dependencia de datos, los resultados de un programa pueden ser diferentes a si el programa se ejecutara sin segmentación
- c) La segmentación de cauce disminuye el número de instrucciones necesarias para la ejecución de un programa
- d) Ninguna de las afirmaciones anteriores

331-Una instrucción de salto si menor, para números positivos sin signo, tiene que comprobar el valor de: el bit de acarreo

332-¿Cuál es el contenido de la pila al terminar de ejecutarse la siguiente secuencia de instrucciones de una arquitectura de pila?:

push #4
push #7
add
push #10
sub
1

333-Para traducir una construcción if-then-else de lenguaje C a lenguaje ensamblador, gcc utiliza generalmente

un salto condicional, según la condición opuesta a la del código C, y otro salto incondicional

334-No en todas las instrucciones máquina
hay una fase de captura de operandos

335-El programa RISC

ld r4,(r2)
ld r5,(r3)
add r6,r4,r5
st (r1),r6

Almacena la suma de los contenidos de las posiciones de memoria direccionadas por los registros r2 y r3, y almacena el resultado en la posición de memoria direccionada por el registro r1.

336-¿Cuál de las siguientes características es posterior a la segunda generación de computadores?

RISC

337-¿En qué pareja de registros están el dato/instrucción que se leerá o escribirá en memoria, y la dirección de memoria?

MBR y MAR

338-En un procesador con segmentación de cauce, aumentar el número de etapas (p.ej. de 2 a 4, o de 4 a 8), tiene en general como consecuencia:

Un incremento de las prestaciones

339-¿Cuál de las siguientes afirmaciones acerca de las interrupciones en el PC es cierta?

Todas las interrupciones se pueden generar por software

340-¿Cuál de las siguientes listas menciona registros x86-64 del mismo tipo respecto a convenio de uso? (salva-invocante, invocado, etc)

a)
RAX, RBX, RCX, RDX

•b)
CL, DX, R8d, R9

c)
RBX, RSI, RDI

d)
RSP, RBP

341-Con tres controladores de interrupciones 8259 se pueden manejar exactamente:

a)
8 niveles de prioridad

b)
16 niveles de prioridad

c)
24 niveles de prioridad

d)
Ninguna de las anteriores es cierta

342-La instrucción SKIP pone a 0 un registro o posición de memoria si se cumple una condición.

Falso

343-Si AX = FA50h y ejecutamos AND AX, 00FFh

El registro AH se pone a 0

344-El cuerpo del siguiente código C: unsigned copy(unsigned u) {return u;}; puede traducirse a ensamblador como:

movl 8(%ebp), %eax

345-¿Qué modificador (switch) de as hace falta para ensamblar una aplicación de 32bits en un sistema de 64bits en el que se ha instalado también el compilador de 32bits?

-32

346-En el direccionamiento inmediato, tras captarse completamente la instrucción:

se accede al operando, que es una constante contenida en la propia instrucción.

347-¿Cuál de los siguientes contenidos no está incluido en un fichero ELF ejecutable?

a)
código máquina
sección .text

b)
tabla de símbolos
incluso símbolos de depuración con -g

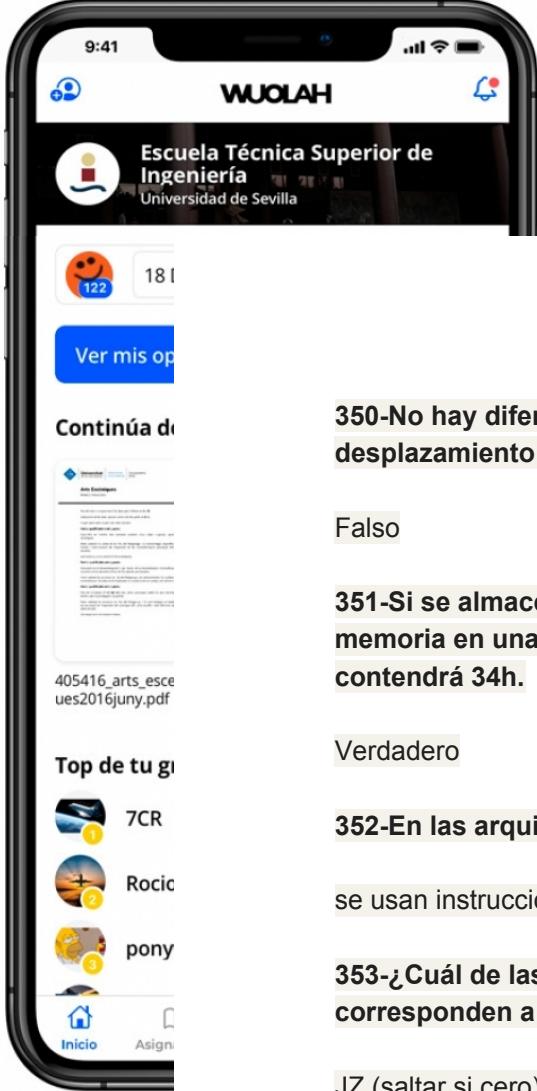
c)
variables globales
secciones .data/.bss
 d)
pila del usuario
por exclusión... y por lógica, para qué almacenarla si está vacía

348-La precaptación (cola de instrucciones) está relacionada con...

Los riesgos estructurales (intenta evitar el efecto de un fallo de cache)

349-¿Cuántos niveles de interrupción podremos gestionar si disponemos de 7 controladores de interrupciones programables 8259?

50



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

350-No hay diferencia entre un desplazamiento lógico a la derecha y un desplazamiento aritmético a la derecha.

Falso

351-Si se almacena el número de 16 bits 1234h en la palabra (16 bits) 0 de memoria en una arquitectura "Big Endian", el byte 0 contendrá 12h y el byte 1 contendrá 34h.

Verdadero

352-En las arquitecturas RISC:

se usan instrucciones muy simples que se pueden segmentar.

353-¿Cuál de las siguientes parejas de mnemotécnicos de ensamblador IA32 corresponden a la misma instrucción máquina?

JZ (saltar si cero), JE (saltar si igual)

354-El primer parámetro de printf:

es un puntero

355-En un 8086, la ejecución de una instrucción LODSB seguida inmediatamente de una STOSB nunca modifica el contenido de ninguna posición de memoria

Falso

356-En la arquitectura Von Neumann...

el programa se encuentra residente en memoria.

357-En una microinstrucción que hace uso del registro de control residual, el contenido de éste normalmente se interpreta como señales de control.

V

358-Para direccionar una memoria de bytes en la que quepan 1G palabras de 32 bits se necesitarán:

32 bits

359-Cuando dos o más instrucciones necesitan un recurso hardware en el mismo ciclo, se trata de un riesgo:
estructural

360-La instrucción INC [3] no tiene sentido.

F

361-Cuando una CPU dispone de muchas líneas de interrupción con un dispositivo en cada una, será necesario utilizar "polling" para detectar la fuente de la interrupción.

F

362-¿Cuál de las siguientes afirmaciones sobre el direccionamiento absoluto es falsa?

El tamaño del operando direccionado queda limitado por el nº de bits del campo dirección de memoria.

363-Un programa con muchas bifurcaciones hace que no se aprovechen al máximo las prestaciones del pipeline.

V

364-Si el sistema operativo lo permite, la tabla de vectores de interrupción puede ser modificada por el usuario.

V

**365-Si el registro EAX contiene X, la sentencia en C
x &= 0x1;
se traducirá a ensamblador como:
andl \$1, %eax**

366-Al diseñar el formato de instrucción:

se suele omitir el campo que indica la siguiente instrucción (la siguiente a ejecutar es la siguiente en memoria, salvo en caso de salto).

367-Se llama "Broadcall" a la escritura simultánea en varios esclavos.

F

368-La transmisión isócrona garantiza un ancho de banda fijo sin que se produzcan interrupciones en el flujo de datos.

V

369-Sobre la E/S mapeada en memoria podemos decir que:

Usa direcciones de memoria para acceder a puertos de E/S

370-Respecto a tamaños de tipos integrales en x86 y x86-64, la excepción es que

long int pasa de 4 B a 8 B

371-En los modos de direccionamiento del tipo

Desplazamiento(Base,Indice,Factor Escala), puede usarse como

desplazamiento, cualquier constante de 1, 2 o 4 bytes (incluso el nombre de una variable, por su dirección)

372-Una computadora puede funcionar prescindiendo de:

un acumulador

373-Los compiladores pueden generar código más fácilmente, y también código más eficiente, para conjuntos de instrucciones tipo RISC.

✓

374-Alguna de las siguientes afirmaciones sobre sistemas Linux x86-64 no es cierta

Todos los argumentos de función se pasan a través de la pila

375-Los microprocesadores RISC puede ser implementados en un tiempo más corto, y requieren menos área de silicio que los CISC.

✓

376-Dada la siguiente definición de datos:

lista: .int 0x10000000, 0x50000000,
0x10000000, 0x20000000

longlista: .int (.lista)/4

resultado: .quad 0x123456789ABCDEF

formato: .ascii "suma=%lu=%lx hex\n\0"

movl \$lista, %ebx

377-La instrucción xor \$3, %eax tiene como resultado:

Cambiar 0<->1 (complemento a 1 de) los últimos 2 bits del registro EAX

378- El direccionamiento indirecto a través de registro es más rápido que el indirecto a través de memoria.

✓

379- Una de las diferencias entre el 8086 y el 8088 es que este último no dispone de BIU ("Bus Interface Unit").

F

380- Las arquitecturas memoria-memoria tienen la ventaja de emplear instrucciones máquina muy cortas y así los accesos a memoria son mínimos.

F

381-Todas las instrucciones de salto/bifurcación realizan una operación del tipo PC = X, si bien X puede ser un dato inmediato, el contenido de un registro, o el contenido de una posición de memoria.

F

382-Un procesador emplea codificación en bloque del código de operación. Existen 123 instrucciones que tienen una longitud de 16 bits. ¿A cuántas direcciones de memoria pueden acceder como máximo si todas emplean una estructura "código de operación + dirección de memoria"?

512

383-Si una pila crece hacia direcciones de memoria decrecientes, una instrucción PUSH utiliza direccionamiento indexado con postautodecremento, siendo SP el registro índice.

F

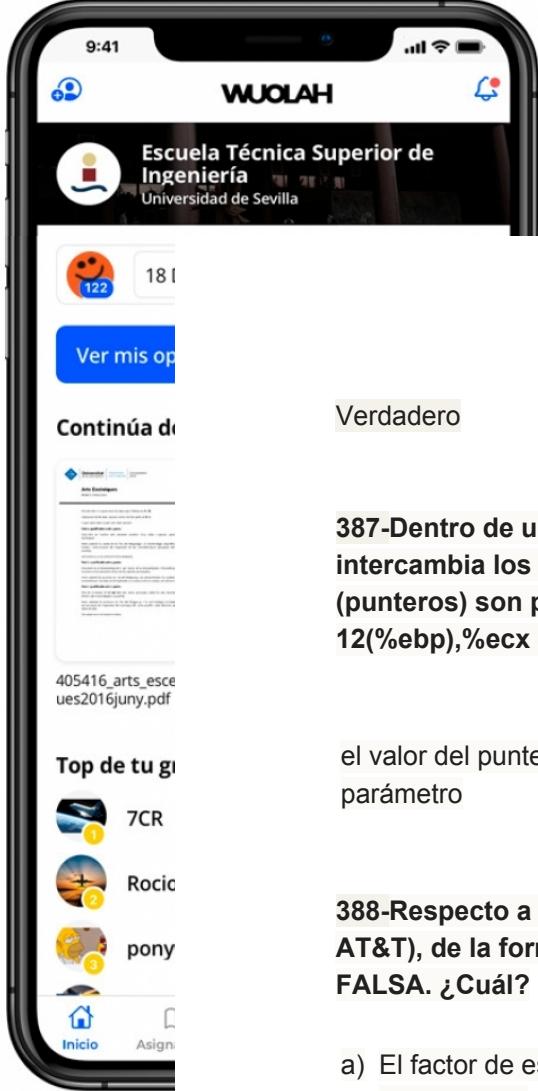
384-¿Cuántos controladores de interrupciones 8259 hacen falta como mínimo para manejar 25 líneas de interrupción?

d) 4

385-Una instrucción de "salto si igual" tiene que comprobar el valor de:

- c) el bit de cero

386-Un diseño CISC pretende disminuir el número de instrucciones a ejecutar por un programa, mientras que uno RISC pretende disminuir el número medio de ciclos por instrucción.



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the App Store

GET IT ON Google Play

Continúa di...

Verdadero

387-Dentro de una función declarada como void swap(int *xp, int *yp), que intercambia los valores de los dos enteros cuyas direcciones de memoria (punteros) son pasadas como parámetros a la función, la instrucción movl 12(%ebp),%ecx copia en %ecx...

el valor del puntero pasado como segundo parámetro

388-Respecto a direccionamiento a memoria en ensamblador IA32 (sintaxis AT&T), de la forma D(Rb, Ri, S), sólo una de las siguientes afirmaciones es FALSA. ¿Cuál?

- a) El factor de escala S puede ser 1, 2, 4, 8
- b) ESP no se puede usar como registro índice
- c) EBP no se puede usar como registro base
- d) El desplazamiento D puede ser una constante literal (1, 2 ó 4 bytes)

389-Para obtener una única velocidad comparativa final, el benchmark SPEC CPU combina las velocidades de ejecución de una serie de tests, respecto a un ordenador de referencia, usando la media...

- c) geométric
- a

390-Por x86-64 se entiende la misma arquitectura de repertorio (ISA) que AMD64

391-Las rutinas de servicio de interrupción son microprogramas que pueden ser escritos por el usuario.

Falso

392-En un PC, un dispositivo que pida interrupciones a través del controlador de interrupciones 8259 no podrá interrumpir por segunda vez a la CPU hasta que no mandemos el comando EOI (20h) al 8259 (puerto 20h).

VERDADERO

393-En las instrucciones aritméticas con dos operandos de un procesador con arquitectura de pila, los dos operandos...

son la cima de la pila y el elemento siguiente de la cima de la pila.

394-Si queremos almacenar la palabra de 16 bits 8965h en memoria según "big-endian", quedará almacenada a partir de la posición 1000h como en el byte 1000h se guarda 89h y en el 1001h 65h

395-¿Qué novedad se desarrolló en la tercera generación de computadores?
Los circuitos integrados

396-En el direccionamiento inmediato, la dirección efectiva se encuentra en un registro general del procesador

Falso

397-¿Cuál de las siguientes afirmaciones sobre la segmentación de cauce es cierta?

En general, un operación segmentada ("pipelined") requiere el mismo tiempo o más, desde el principio hasta el fin, que la misma operación en una implementación no segmentada

398-¿Cuál de las siguientes afirmaciones es correcta?

- a) Las instrucciones en lenguaje máquina se almacenan y tratan como cadenas de unos y ceros.
- b) Las instrucciones en lenguaje ensamblador se almacenan y tratan como cadenas de unos y ceros.
- c) El lenguaje máquina es igual para todos los computadores.
- d) El lenguaje ensamblador es igual para todos los computadores.

399-Cuando se usan interrupciones vectorizadas, el periférico puede suministrar al procesador un índice referente a una tabla de vectores de interrupción.

Verdadero

400-Los modos de direccionamiento de una instrucción se especifican en ensamblador mediante las directivas

Falso

401-El programador de lenguaje ensamblador necesita conocer:

la arquitectura del ordenador.

402-¿Qué forma de realizar acceso directo a memoria es más rápida?

Transferencia de bloques o parada de CPU

403-En un sistema IA32 Linux, ¿cuál es el tamaño de un long?

4 bytes

404-¿Qué hace gcc -S?

Compilar .c → .s (fuente C a fuente ASM)

405-Un sistema no segmentado tarda 20 ns en procesar una instrucción. Las instrucciones pueden ser procesadas en un cauce (pipeline) de 4 segmentos con un ciclo de reloj de 6 ns. Cuando se procesan muchas instrucciones, la ganancia máxima de velocidad que se obtiene se aproxima a:

3,33

406-Las unidades de control microprogramadas utilizan habitualmente memorias RAM para almacenar los microprogramas.

Falso

407-Justo antes de que una instrucción máquina escriba un resultado en memoria:

en MBR está el resultado y en MAR la dirección donde se almacenará

408-En la práctica "media" se suma una lista de 32 enteros de 4 B con signo para producir una media y un resto usando la instrucción IDIV. ¿Cuál de las siguientes afirmaciones es falsa?

- a) El resto siempre tiene el mismo signo que la suma

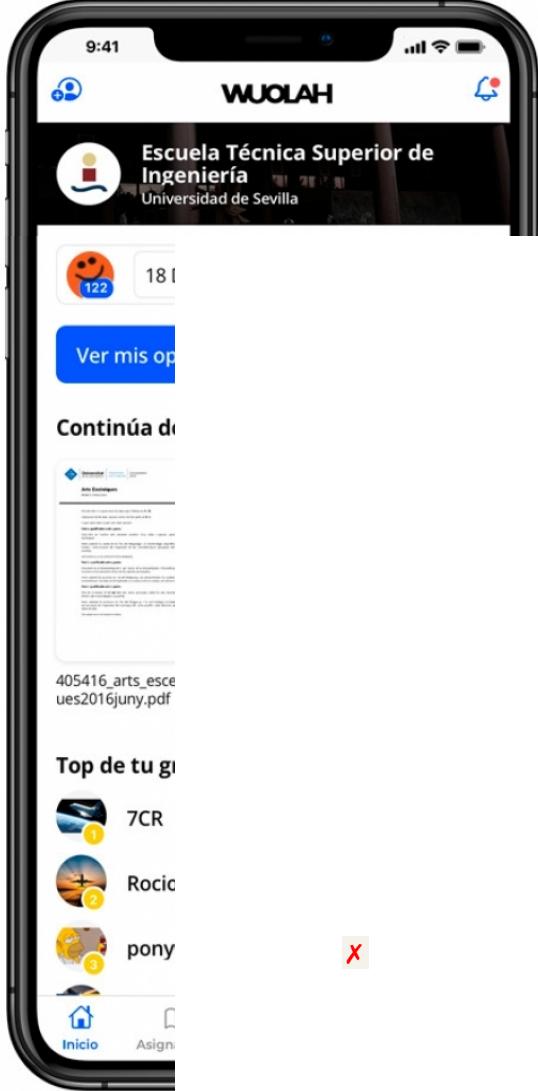
división truncada IDIV - resto del mismo signo que dividendo

En matemáticas se usa división euclídea - resto modular $0 \leq r < divisor$

https://es.wikipedia.org/wiki/Divisi%C3%B3n_eucl%C3%ADdea

•

- b) La media se redondea al entero más próximo división truncada IDIV - no se redondea, se trunca



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the App Store GET IT ON Google Play

c) IDIV
produce el mismo resto que el operador % en lenguaje C

división truncada ambos - resto del mismo signo que dividendo

d) IDIV
produce el mismo cociente que el operador / en lenguaje C

división truncada ambos - resto del mismo signo que dividendo

409-¿Qué modificador (switch) de ld hace falta para enlazar una aplicación de 32bits en un sistema de 64bits en el que se ha instalado también el compilador de 32bits?

-m elf_i386

410-¿Cuál de las siguientes afirmaciones es correcta?

Las instrucciones en lenguaje máquina se almacenan y tratan como cadenas de unos y ceros.

411-Un sistema no segmentado tarda 50 ns en procesar una tarea. La misma tarea puede ser procesada en un cauce ("pipeline") de 5 segmentos con un ciclo de reloj de 10 ns. Cuando se procesan muchas tareas, la ganancia de velocidad que se obtiene se acerca a 5.

Verdadero

412-Si queremos almacenar la palabra de 16 bits 0x8965 en una memoria de bytes según "little-endian", quedará almacenada a partir de la posición 0x8600 como

M[0x8600]=0x65 y M[0x8601]=0x89

413-Considerar las siguientes declaraciones de estructuras en una máquina Linux de 64-bit.

```
struct RECORD {  
    long value2;  
    int ref_count;  
    char tag[4];  
};  
  
struct NODE {  
    double value;  
    struct RECORD record;  
    char string[8];  
};
```

También se declara una variable global "my_node" como sigue:

```
struct NODE my_node;
```

Si la dirección de my_node es 0x601040, ¿cuál es el valor de &my_node.record.tag[1]?

0x601055

414-¿Qué hace gcc -O?

Compilar con optimización, igual que -O1

415-En una bomba como las estudiadas en prácticas, del tipo...

```
0x080486e8 <main+120>: call 0x8048524 <strncmp>
0x080486ed <main+125>: test %eax,%eax
0x080486ef <main+127>: je 0x80486f6 <main+134>
0x080486f1 <main+129>: call 0x8048604 <boom>
0x080486f6 <main+134>: ...
```

la contraseña es...

- a) el string almacenado a partir de
0x8048524
 - b) el string almacenado a partir de
0x80486f6
 - c) el string almacenado a partir de
0x8048604
 -
 - d) ninguna de las anteriores

416-Las técnicas principales de E/S son (señalar la respuesta falsa)

- a) IRQ (por
interrupciones)
 - ✓
 -
 - b) E/S cableada
(hardwired)
 - c) E/S programada
 - d) DMA (por acceso
directo)

417-Una forma usual de realizar el arbitraje distribuido consiste en una competición por la concesión del bus realizada por medio del envío por cada maestro peticionario de un número de arbitraje que lo identifica, de manera que un solo número "gane", y se le conceda el bus al maestro con ese número ganador.

VERDADERO

418-Si AX = FA50h y ejecutamos XOR AX, 00FFh

Se realiza el complemento a 1 de AL

419-La instrucción "desplazar a la derecha aritméticamente" suele ser idéntica a la "desplazar a la derecha lógicamente".

FALSO

420-Suele existir una instrucción en el repertorio de la mayoría de los microprocesadores para inhibir la petición HOLD del controlador de DMA.

FALSO

421-Respecto a los conceptos de procesamiento segmentado y superescalar, una de las siguientes afirmaciones es falsa

En cualquier procesador resulta ventajoso usar una cola de instrucciones, pero es más importante para uno segmentado (fundamental) que para uno superescalar (conveniente)

423-La CPU no guarda automáticamente el contexto del programa que ejecuta cuando le llega y atiende una petición de DMA.

Verdadero

424-La diferencia entre temporización de bus semisíncrona y asíncrona es que en la semisíncrona las transferencias ocurren en algún múltiplo de ciclo de reloj, y en la asíncrona no existe reloj del bus

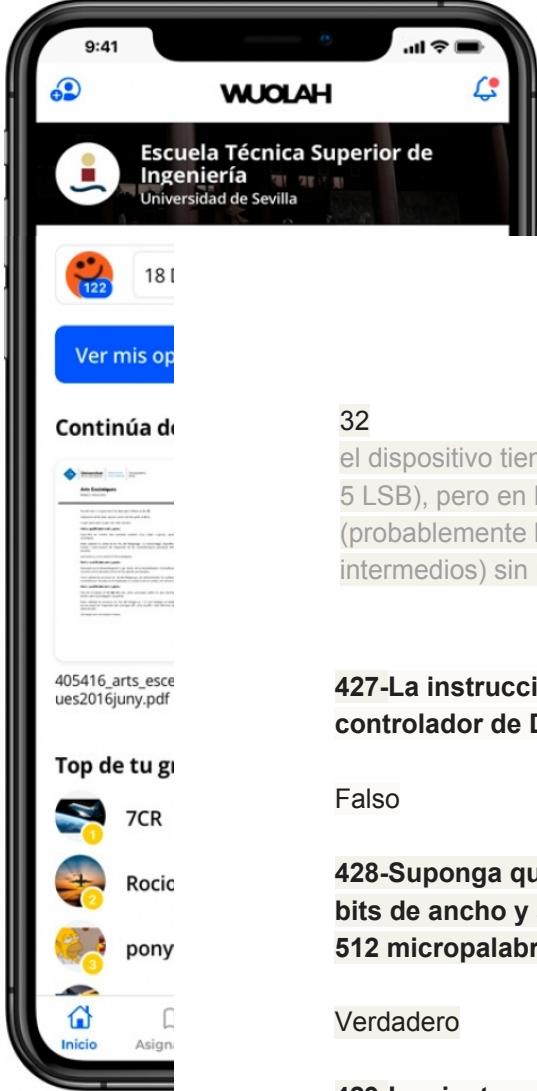
VERDADERO

425-Una unidad de control microprogramada con secuenciamiento explícito con dos direcciones por microinstrucción, tiene una memoria de control con 35 bits de longitud de palabra. Si las microinstrucciones emplean 15 bits en total para los campos de control y de tipo y condición de salto, el número máximo de palabras de la memoria de control de esta unidad de control microprogramada es de:

2^{10}

426-Si en un bus de direcciones de 32 bits se decodifica parcialmente la dirección de un dispositivo de 32 posiciones usando 22 bits, ¿cuántas veces aparecerá repetido en el mapa de memoria?

32



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

32

el dispositivo tiene 32 puertos, usa 5 bits de direccionamiento (probablemente los 5 LSB), pero en lugar de decodificarse con 27, se usan sólo 22 bits (probablemente los 22 MSB) de manera que quedan 5 bits (probablemente intermedios) sin usar, que pueden tomar 32 combinaciones posibles

427-La instrucción DISABLE (CLI en el 8086) inhibe la petición HOLD del controlador de DMA.

Falso

428-Suponga que la micropalabra de una máquina microprogramada tiene 8 bits de ancho y se usan 256 micropalabras diferentes en un microprograma de 512 micropalabras. No se ahorran bits usando nanoprogramación.

Verdadero

429-Las instrucciones máquina más usadas según el análisis dinámico son las de bifurcación.

Falso

430-El ajuste de marco de pila que gcc (Linux/IA-32) prepara para todas las funciones consiste en las instrucciones

```
pushl %ebp  
movl %esp,  
%ebp
```

431-Algunas de las ventajas de la E/S mapeada en memoria frente a la E/S aislada o independiente son:

El diseño de la CPU es más sencillo.

432-En la práctica “media” se pide sumar una lista de 32 enteros *con* signo de 32bits en una plataforma de 32bits sin perder precisión, esto es, evitando overflow. ¿Cuál es el mayor valor negativo (menor en valor absoluto) que repetido en toda la lista causaría overflow con 32bits?

0xfbff
ffff

433-La transferencia de datos en un computador y los dispositivos de E/S puede manejarse de diversos modos. Uno de los siguientes es falso; indíquelo:

Manejo de todas las líneas del bus de control, paralizando la CPU

434-Las señales de carga/incremento/desplazamiento de registros, las de selección de entradas de multiplexores, y las de selección de función de la ALU son entradas a la unidad de control

FALSO

435-¿Cuál es el contenido de la pila al terminar de ejecutarse la siguiente secuencia de instrucciones de una arquitectura de pila: push #4; push #7; add; push #10; sub?

1

436-Un controlador de DMA de un sistema de que emplee buses separados avanzados suele ser programado con la siguiente información relativa a una operación de E/S

tipo de operación, tamaño de bloque a transferir, dirección inicial de memoria

437-¿Por qué se impusieron las arquitecturas de registros de propósito general a las arquitecturas basadas en pila?

Porque las basadas en registros son capaces de lograr un mejor rendimiento cuando se asignan variables a registros

438-¿Cuál de los siguientes microprocesadores no es de 64 bits?

Pentium III

439-¿Cuál de las siguientes afirmaciones acerca de las interrupciones en el PC (modo real) es cierta?

a) Existen

1024
vectores de
interrupción



- b) Cada vector de interrupción es una doble palabra de 32 bits formada en primer lugar (dirección menor) por el segmento y seguida por el desplazamiento (dirección mayor) de cada rutina de servicio de interrupción
- c) No todas las interrupciones se pueden generar por software
- d) Ninguna de las anteriores afirmaciones es cierta

440-Un procesador emplea codificación en bloque del código de operación. Existen 120 instrucciones que tienen una longitud de 16 bits. ¿A cuántas direcciones de memoria pueden acceder como máximo si todas emplean una estructura "código de operación + dirección de memoria"?

512

441-El direccionamiento a registro es similar al directo. La diferencia es que el campo de direcciones referencia un registro en lugar de una dirección de memoria

VERDADERO

442-La práctica "popcount" debía calcular la suma de bits (peso Hamming) de los elementos de un array. Un estudiante entrega la siguiente versión de popcount3:

```
int popcount3(unsigned* array,
              int len){
    int i, res = 0;
    unsigned x;
    for( i=0; i<len; i++ ) {
        x = array[i];
        asm("ini3: \n"
            "shr %[x] \n"
            "adc $0, %[r] \n"
            "add $0, %[x] \n"

```

```
"jne ini3 \n"
: [r] "+r" (res)
: [x] "r" (x );
}
return res;
}
```

Esta función produce siempre el resultado correcto, a pesar de que una instrucción máquina en la sección asm() es distinta a la que se esperaba después de haber estudiado pcount_r en teoría. La instrucción distinta también se podría haber cambiado por test %[x], %[x]

443-¿Cuál es el contenido de una pila al terminar de ejecutarse la siguiente secuencia de operaciones push y pop:

```
push #1
push #2
push #3
pop a
push #4
pop a
pop a
```

1

444-En la práctica "media" se programa la suma de una lista de 32 enteros de 4 B para producir un resultado de 8 B, primero sin signo y luego con signo. Si la lista se rellena con el valor 0x0400 0000, ¿en qué se diferencian los resultados de ambos programas?

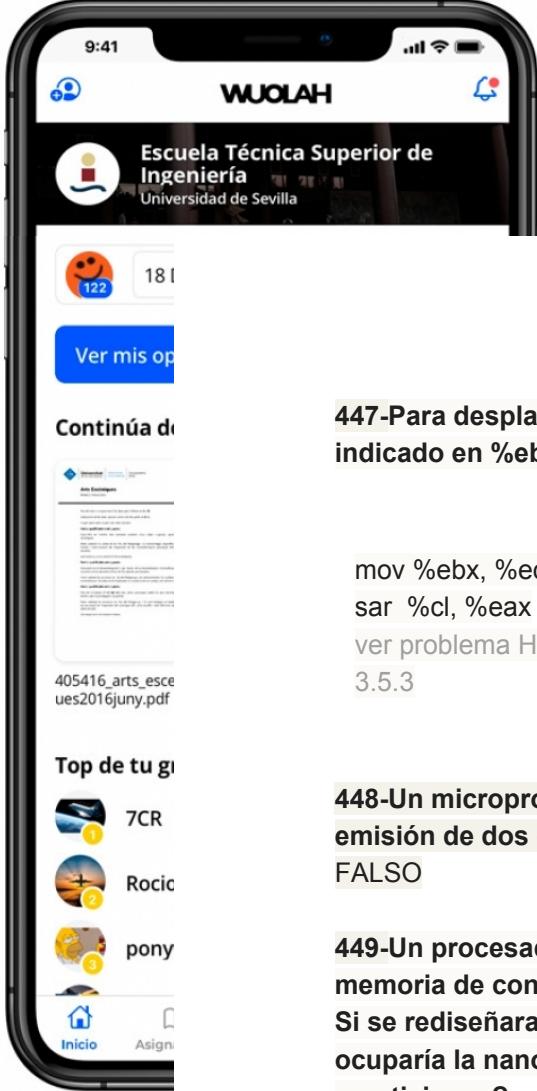
no se diferencian

445-El campo tipo de secuenciamiento indica al generador de direcciones de una unidad de control microprogramada el mecanismo de cómputo de la EA del operando de la instrucción.

FALSO

446-La conexión de un 8086 a un sistema de memoria y E/S requiere algún circuito externo más en modo máximo que en modo mínimo.

Verdadero



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

447-Para desplazar %eax a la derecha un número variable de posiciones <= 32, indicado en %ebx, se puede hacer

mov %ebx, %ecx
sar %cl, %eax
ver problema Hallaron 3.8 y sección
3.5.3

448-Un microprocesador es superencauzado ("superpipelined") si permite la emisión de dos o más instrucciones en un mismo ciclo de reloj
FALSO

449-Un procesador con una unidad de control microprogramada tiene una memoria de control de 300 palabras de 100 bits, de las que 200 son diferentes. Si se rediseñara como unidad de control nanoprogramada, ¿qué tamaño ocuparía la nanomemoria que contiene las microinstrucciones completas sin repeticiones?

20000 bits
200 uinstr. x 100 bits

450-Una máquina que almacene el número 4321h a partir de la dirección 0 como 21h, 43h utiliza el sistema little-endian.
VERDADERO

451-En las arquitecturas de registros de propósito general suele ser menor el tráfico entre procesador y memoria que en las de acumulador.

Verdadero

452-En la codificación por extensión de campo, el campo de código de operación aumenta su tamaño a medida que disminuye el número o la longitud de campos de dirección de operandos.

VERDADERO

453-

Si un procesador no segmentado necesita 5 ns para leer una instrucción de memoria, 2 ns para decodificar la instrucción, 3 ns para leer del banco de registros, 3 ns para realizar el cálculo requerido por la instrucción, y 2 ns para

escribir el resultado en el banco de registros, ¿cuál es la frecuencia de reloj máxima del procesador?

66,67 MHz

454-El especificador de operando de una instrucción, cuando existe, es siempre una dirección de memoria o de entrada/salida.

F

455-¿Cuál de las siguientes instrucciones de IA32 (en sintaxis Intel) no es una instrucción de transferencia?

cmp eax,15h

456-El análisis dinámico de la frecuencia de utilización de instrucciones se realiza mediante el estudio de un gran número de listados de programas.

F

457-¿Cuál de las siguientes instrucciones convierte %eax = 5 * %eax?

- 1) mov (%eax, %eax, 4), %eax
- 2) lea (%eax, %eax, 4), %eax

Sólo 2

458-La codificación Huffman es la más utilizada debido a que consigue resultados óptimos y su proceso de decodificación es sencillo.

F

459-Un computador que almacene el número 2143h como 43h en el byte de la dirección 0 y 21h en el byte de la dirección 1, utiliza el sistema big-endian.

F

460-¿Cuál de las siguientes afirmaciones es incorrecta?

- a) En lenguajes de alto nivel no se tiene acceso a detalles del procesador como las tablas de páginas, el paso de modo usuario a modo supervisor, el bit de paridad, etc.
- b) El lenguaje de alto nivel es más portable que el lenguaje máquina.

- c) En lenguaje ensamblador cada instrucción se corresponde con una instrucción máquina.
- d) En lenguaje ensamblador las instrucciones se escriben en binario.

461-El direccionamiento relativo a contador de programa favorece la implementación de código reubicable.

V

462-Un computador usa el formato vertical de codificación de instrucciones para parte de las señales de control y el formato horizontal para k señales de control. El formato vertical posee n campos codificados de m bits cada uno. ¿Cuál es el máximo número de señales de control que pueden usarse en este computador?

$$k + n \cdot (2^m - 1)$$

463-¿De qué depende el tamaño del contador de programa?

Del número de direcciones de memoria.

464-En X86-64, el registro contador de programa se denomina:

RIP

465-En una CPU de 32 bits con memoria de bytes, el problema es que...

Hay que respetar el ordenamiento de bytes y reglas de alineamiento con que se diseñó la CPU

466-En el direccionamiento inmediato:

el tamaño (rango de valores) de la constante está limitado.

467-Un procesador emplea codificación en bloque del código de operación. Existen 130 instrucciones que tienen una longitud de 16 bits. ¿A cuántas direcciones de memoria pueden acceder como máximo si todas emplean una estructura "código de operación + dirección de memoria"?

256

468-Se puede programar un controlador de interrupciones 8259 de manera que atienda equitativamente a 8 dispositivos de igual prioridad (cada vez que se atiende a un dispositivo, éste pasa automáticamente a tener la prioridad más baja)

VERDADERO

469-¿Cuál es el resultado de evaluar la expresión 0b1110 ^ 0b1010 en lenguaje C?

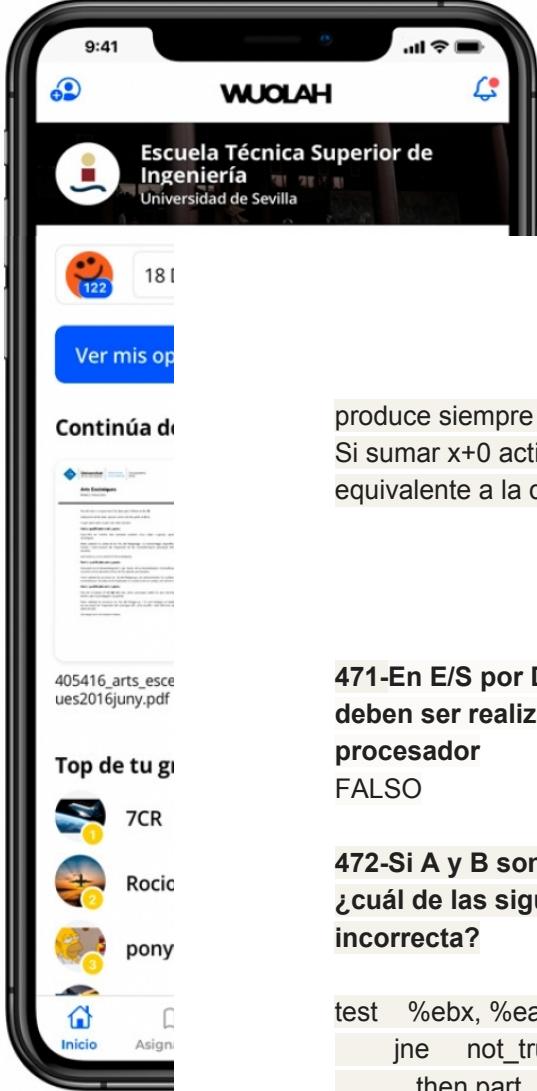
0b0100 (hace el AND y pone 1 cuando no es AND, 1 y 1 = 0, 0 y 1 = 1)

470-La práctica "popcount" debía calcular la suma de bits (peso Hamming) de los elementos de un array. Un estudiante entrega la siguiente versión de `popcount3`:

```
int popcount3(unsigned* array,
              int len){
    int i, res = 0;
    unsigned x;
    for( i = 0; i < len; i++ ) {
        x = array[i];
        asm("ini3: \n"
            "shr %[x] \n"
            "adc $0, %[r] \n"
            "add $0, %[x] \n"
            "jne ini3 \n"
            : [r] "+r" (res)
            : [x] "r" (x) );
    }
    return res;
}
```

Esta función sólo tiene una diferencia con la versión "oficial" recomendada en clase. En concreto, una instrucción máquina en la sección `asm()` es distinta.

Esta función `popcount3`:



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

produce siempre el resultado correcto

Si sumar $x+0$ activa ZF sólo puede ser porque ya era $x==0$, así que la lógica es equivalente a la deseada.

471-En E/S por DMA, las operaciones elementales de transferencia de datos deben ser realizadas por medio de la ejecución de instrucciones del procesador

FALSO

472-Si A y B son dos enteros almacenados respectivamente en %eax y %ebx, ¿cuál de las siguientes implementaciones de if (!A && !B) {...then part...} es incorrecta?

```
test %ebx, %eax
    jne not_true
    ...then part...
not_true:
    ...
test hace AND, no OR
equivale a if (A&&B) {... then_part ...}
```

473-Si el contenido de r4 es 0x13000, ¿a qué dirección se hace referencia con la instrucción LD -0x80(r4)?

(Nota: 0x indica notación hexadecimal en C y C++)

0x12F80

474-¿Cuál de las siguientes instrucciones es errónea? (sale mensaje de error al intentar ensamblar)

a)
movw %dx, (%eax)

b)
movzbl %dl, %eax

c)

movswl (%eax), %edx

•d)

pushb \$0xFF

475-El "handshaking" se utiliza en transferencias de datos "síncronas" entre CPU y periféricos.

Falso

476-El primer microprocesador de 32 bits de la familia x86 fue 80386

477-Si el acceso directo a memoria se realiza mediante robo de ciclo:
es posible que la ejecución de una instrucción máquina sea temporalmente detenida

478-Un cauce ("pipeline") de instrucciones inicialmente vacío y con 3 etapas tardará siempre 5 ciclos de reloj en ejecutar 3 instrucciones si cada una de ellas utiliza las 3 etapas.

Falso

479-Para direccionar una memoria de 2 G palabras de 32 bits cada una, que se direcciona byte a byte, se necesitarán:

33 bits como mínimo

480-En los RISC la sección de control del procesador debe ser microprogramada en lugar de cableada debido al uso de un pequeño número de operaciones simples.

481-¿En qué generación, dentro de la historia de los computadores digitales, aparecen los sistemas operativos multiusuario?
tercera

482-¿Qué dice la ley de Moore?

Que el número de transistores de un chip se duplica cada 18 meses.

483-En el acceso directo a memoria la CPU pone en el bus de direcciones del sistema las direcciones de memoria correspondientes a cada dato que se transfiere por DMA.

FALSO

484-¿Qué circuito suele utilizarse para traducir el código de operación de una instrucción máquina a dirección de comienzo en la memoria de control del microprograma correspondiente?

Una memoria.

485-¿Con cuál de los siguientes dispositivos tendría sentido utilizar E/S programada sin consulta de estado?

Salida a un display de 7 segmentos

486-Después de ejecutar el siguiente código, ¿qué variables serán igual a 0? (Suponer ints de 32bits y longs de 64bits)

```
unsigned int a = 0xffffffff;
unsigned int b = 1;
unsigned int c = a + b;
unsigned long d = a + b;
unsigned long e = (unsigned long)a + b;
```

c y d

En el problema 3.4 sólo se explica que una extensión de tamaño se hace según el fuente sea signed (extensión sgn) o unsigned (ext. con ceros), pero no se explica la sección 2.2.6 (y 2.2.5) en donde se aclara que las operaciones que impliquen a algún unsigned se hacen en unsigned. Evitar esta pregunta en el futuro.

487-Un microprocesador es superencauzado ("superpipelined") si permite la emisión de dos o más instrucciones en un mismo ciclo de reloj.

Falso

488-Una arquitectura de registros de propósito general puede tener instrucciones máquina de la ALU con un único operando explícito.

Verdadero

489-¿Qué tipo de instrucciones se emplean más en una arquitectura de acumulador?

de transferencia de datos con memoria

490-¿Qué modificador (switch) de gcc hace falta para compilar .s → .o sin llamar al enlazador?

gcc -c

491-Si queremos almacenar la palabra de 16 bits 9660h en memoria según "little-endian", quedará almacenada a partir de la posición 1000h como:

en el byte 1000h se guarda 60h y en el 1001h se guarda 96h

492-En una resta de dos números en complemento a dos, se produce desbordamiento cuando...

a)

los dos operandos son negativos y el resultado es positivo.

b)

los dos operandos son positivos y el resultado es negativo.

x

c)

Las dos respuestas a y b son correctas.

d)

Ninguna de las anteriores es correcta.

493-Si el registro %edx contiene la variable definida con la sentencia en C “int n”, y queremos dividirla por 2, usaremos la instrucción:

sarl %edx,1

494-Una de las funciones de una interfaz de E/S es realizar la conversión de cualquier formato que pueda ser necesaria para transferir datos entre el bus de E/S y el dispositivo de E/S (por ejemplo, de paralelo a serie).



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

Verdadero

495-Indicar cuál es la dirección de salto (en qué dirección empieza la subrutina <main>) para esta instrucción call

0804854e: e8 3d 06 00 00 call <main>
08048553: 50 pushl %eax

08048b90

496-Si tenemos un número n, de 64 bits, almacenado en la pareja de registros EDX:EAX (EDX contiene los 32 bits más significativos y EAX los 32 bits menos significativos) y queremos realizar la división n / (2^32) entonces:

Podemos quedarnos con EDX tanto si n es un número con signo como sin signo. Cocientes positivos o sin signo redondeados hacia cero, cocientes negativos redondeados hacia más negativo: es decir, redondeado siempre hacia inferior. No es la división entera estándar de C (que redondea hacia cero) pero el enunciado tampoco lo exige.

497-¿Cuántos puertos de E/S permite manejar la interfaz de periféricos programable 8255?

- c) 4 puertos de 32 bits

498-Con respecto a los buses de la placa base:

- El bus ISA es un bus
a) local.

X

.

- b) El bus EISA es un bus local.
- c) El bus MCA es un bus local.
- d) Ninguna de las anteriores es cierta.

499-Una función C llamada get_el() genera el siguiente código ensamblador. Se puede adivinar que:

```
movl 8(%ebp), %eax
leal (%eax,%eax,4), %eax
addl 12(%ebp), %eax
movl var(%eax,4), %eax
```

var es un array bidimensional de enteros, con cinco columnas

se puede adivinar incluso el código:

```
int get_el(int fila, int columna){
    return var[fila][columna];
}
```

porque se calcula %eax=fila*5+columna para indexar en var(%eax,4)

500-La práctica “parity” debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la siguiente versión de parity4:

```
int parity4(unsigned* array, int len){
    int val,i,res=0;
    unsigned x;
    for (i=0; i<len; i++){
        x=array[i];
        val=0;
        asm("\n"
            "ini3:      \n\t"
            "xor %[x],%[v] \n\t"
            "shr %[x]     \n\t"
            "test %[x], %[x]\n\t"
            "jne ini3   \n\t"
            "[v]" +r" (val)
            "[x]" "r" (x)
            );
        val = val & 0x1;
        res+=val;
    }
    return res;
}
```

La sentencia asm() del listado anterior tiene las siguientes restricciones

dos entradas y una salida

- c) [v] cuenta como salida y entrada, [x] como
entrada

**501-¿Cuántas patillas de dirección tiene una memoria DRAM de 1G palabra,
siendo la longitud de palabra de 16 bits?**

15

**502-Una memoria que está estructurada en palabras de 8 bits tiene una
capacidad de 64 Kbits. ¿Cuántas líneas de dirección tiene dicha memoria?**

13

503-Las celdas de memoria estática...

mantienen la información almacenada por tiempo indefinido mientras se
mantenga la alimentación

504-Toda celda de memoria DRAM tiene al menos cuatro transistores.

FALSO

**505-Una organización de interrupciones de niveles múltiples con prioridad
significa que, durante la ejecución de una rutina de servicio de interrupción, se
aceptarán solicitudes de interrupción de algunos dispositivos, pero no de
otros, según sea su prioridad**

**506-¿Cuál de los siguientes grupos de señales son entradas a la unidad de
control?**

- a) Las señales de
carga/incremento/desplazamiento
de registros
- b) Las señales de selección de entradas
de multiplexores del datapath
-
- c) Los bits del registro de indicadores
(flags)
- d) Los bits de las opciones b y c

X

507-Para direccionar una memoria de bytes en la que quepan 2G palabras de 32 bits se necesitarán:

33 bits como mínimo

508-Los riesgos de datos consisten en que...

una instrucción necesita un dato calculado por otra anterior

509-Las unidades de control cableadas memorizan los pasos de ejecución de una instrucción máquina en una memoria de control, y las microprogramadas en una PLA.

Falso

510-La instrucción IA32 test sirve para...

Realizar la operación and lógico bit-a-bit (a&b) pero no guardar el resultado, sino simplemente ajustar los flags

511-Respecto a las técnicas de direccionamiento por selección lineal, decodificación centralizada y distribuida

la selección lineal permitiría escribir un mismo dato a varios puertos E/S

512-¿Cuál sería el "equivalente x86-64" del "pseudo-código C" `rcx = ((int*)rax)[rcx]`?

`mov (%rax,%rcx,4),%rcx`



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

513-En la configuración de E/S mapeada en memoria, la CPU tiene instrucciones de E/S, y cuando se ejecuta una de ellas, la CPU habilita alguna línea especial que sirve para que la circuitería externa decodifique por separado las direcciones correspondientes a memoria y las correspondientes a puertos de E/S.

Falso

514-Si ECX vale 0, la instrucción adc \$0,%ecx

Pone CF=0

515-En el modo de direccionamiento relativo a registro base se especifica un registro base y una dirección de memoria cuyo contenido se suma al contenido del registro base para obtener la dirección efectiva.

Falso

516-¿Cuál de las siguientes afirmaciones sobre el benchmark SPEC CPU es falsa?

El resultado final es la media aritmética de las (12 ó 17) velocidades, bien sea de enteros ó de punto flotante (SPECint2006 ó SPECfp2006)

517-¿Qué valor contendrá %edx tras ejecutar las siguientes instrucciones?

```
xor %eax, %eax
sub $1, %eax
cltd
idiv %eax
```

0

518-¿Cuál de las siguientes no es una sección de un fichero ELF?

- a)
.text
para el código
- b)
.static

inventada

c)

.data

para datos inicializados como en la Práctica "media"

x d)

.bss

para datos sin inicialización, mencionada en P2 Apéndice 2 Tabla 9

519-La evolución de la arquitectura de computadores a lo largo de la historia de éstos se ha producido de forma desligada a la evolución de la tecnología de fabricación de dispositivos electrónicos.

Falso

520-El incremento de velocidad de los RISC se consigue a costa de un aumento del área de chip dedicado a la unidad de control.

Falso

521-El objetivo del control residual es aumentar la velocidad de ejecución de microinstrucciones, aunque esto tiene el inconveniente de aumentar el tamaño del microprograma.

FALSO

522-¿De qué depende el tamaño del contador de programa?

Del número de direcciones de memoria.

523-En un sistema con un único bus...

a) sólo un dispositivo puede escribir en un instante dado en el bus

- b) se utilizan las mismas líneas de control para conectar todos los dispositivos
- c) el procesador y los periféricos pueden funcionar a diferentes velocidades si el funcionamiento del bus es asíncrono



•

- d) Todas las respuestas anteriores son ciertas

524-Escribiendo en el registro de máscara de interrupciones es posible inhabilitar todas las interrupciones enmascarables.

Verdadero

525-¿Cuál de las siguientes no es una característica de los computadores RISC?

La decodificación de las instrucciones

- a) debe ser simple: un computador RISC debería emplear un único formato de instrucción
-
- b) Para acelerar el computador RISC se emplean técnicas de pipelining.
- c) Las funciones que realizan los computadores RISC deben ser lo más complejas y potentes que sea posible.
- d) Un computador RISC no debe emplear microprogramación.

526-Los procesadores RISC no tiene instrucciones para sumar un registro con una posición de memoria.

Verdadero

527-En la práctica "media" se programa la suma de una lista de 32 enteros de 4 B para producir un resultado de 8 B, primero sin signo y luego con signo. Si la lista se rellena con el valor que se indica a continuación, ¿en qué caso ambos programas producen el mismo resultado?

0x1111 1111

resultado 0x0000 0002 2222 2220

porque es positivo incluso en complemento a 2

todos los demás valores se interpretan como negativos, lo primero que hace la suma con signo es extenderlos a 64bit de manera que se activan los 32 bits superiores... resultado radicalmente distinto

528-El entrelazado de orden superior se utiliza más que el de orden inferior en procesadores vectoriales debido a que su expansión es más fácil.

FALSO

529-La expresión $d(t+k) = d(t)$, donde $d(x)$ es la dirección de memoria referenciada en el instante de tiempo x y k es un pequeño incremento de tiempo, corresponde al principio de localidad espacial.

FALSO

530-La penalización por una falta de página suele ser de unas pocas decenas de ciclos de reloj de la CPU.

Compilar sin optimización

531-De las siguientes parejas de instrucciones, ¿cuál utiliza únicamente direccionamiento implícito?

- a) mul,
div
- b) push,
call
- c) jnz, cmp
- d) sti, popf

532-Suponga un ordenador que usa un tamaño de palabra de 32 bits y un espacio de direccionamiento de 20 bits. El valor hexadecimal de la dirección más alta es FFFFFFFFh.

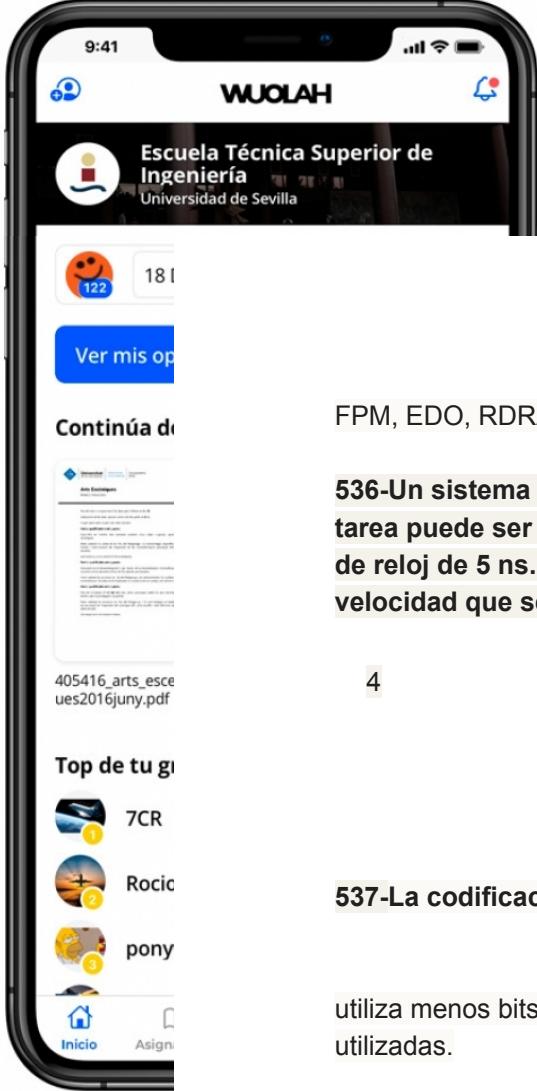
FALSO

533-¿Qué hace gcc -O0?

534-El 8255 es un interfase paralelo de periféricos que suele utilizarse para E/S programada, pero también puede generar interrupciones.

VERDADERO

535-¿Cuál de las siguientes secuencias de tipos de memoria está ordenada de menores a mayores prestaciones?



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

FPM, EDO, RDRAM

536-Un sistema no segmentado tarda 20 ns en procesar una tarea. La misma tarea puede ser procesada en un cauce (pipeline) de 4 segmentos con un ciclo de reloj de 5 ns. Cuando se procesan muchas tareas, la ganancia máxima de velocidad que se obtiene se acerca a:

4

537-La codificación Huffman...

utiliza menos bits para las instrucciones más frecuentes y más bits para las menos utilizadas.

538-La penalización por una falta de página suele ser de unas pocas decenas de ciclos de reloj de la CPU.

FALSO

539-En un procesador capaz de direccionar la memoria a nivel de bytes, con una longitud de palabra y tamaño del bus de datos de 32 bits, ¿cuántos accesos a memoria hacen falta para llevar de la memoria al procesador un objeto de tamaño doble palabra situado a partir del byte de dirección 4 (inclusive)?

2

540-El primer microprocesador de 32 bits de la familia x86 fue el:

80386

541-En los procesadores RISC todas las instrucciones son del tipo registro-registro, exceptuando las de movimiento de datos entre registros.

Falso

542-El acceso a memoria en modo página no se utiliza en segmentación no paginada.

Falso

543-¿Qué técnica de E/S se dice controlada por hardware?

Acceso directo a memoria

544-GCC/Linux IA32 resuelve el ajuste de marco de pila mediante las instrucciones:

`pushl %ebp; movl %esp, %ebp`

545-En una memoria virtual con segmentos paginados, cada entrada de la tabla de segmentos contiene un puntero a la primera dirección física del segmento correspondiente.

Falso

546-Respecto a los términos microinstrucción y microcódigo:

Microcódigo es el contenido de la memoria de control, y una microinstrucción es una palabra de dicha memoria

tr.42 define literalmente ucód. como conjunto de microprogs. que a su vez son el conjunto de uinstr. de una instr.

547-Sólo los microprocesadores RISC utilizan segmentación en la ejecución de sus instrucciones.

FALSO

548-SP y PC no son registros de uso general (GPR).

VERDADERO

549-La propiedad de localidad espacial se puede enunciar como: "la información que se usará en un futuro próximo es la misma que se está usando actualmente".

Falso

550-Un procesador de 1GHz tarda 4ns en realizar 4 instrucciones sin realizar segmentación de cauce. ¿Cuanto tardaría en realizar 9 instrucciones un procesador con segmentación de cauce de 4 etapas si no existiera ningún retraso en ninguna de las instrucciones?

3 ns

551-Cuando se usan interrupciones vectorizadas, el periférico siempre suministra a la CPU la dirección de memoria (completa o incompleta) donde comienza la rutina de servicio de interrupción.

FALSO

552-RAS significa impulso de acceso a filas o selección de acceso a filas.

VERDADERO

553-¿Cuál de las siguientes afirmaciones es falsa?

a)

el bus de control transporta señales de estado

x

b)

el bus de direcciones es unidireccional

c)

el bus de datos es bidireccional

d)

la anchura del bus de datos es de 16 bits

554-No existen instrucciones máquina que únicamente contengan campo de operación.

FALSO

555-La eficiencia de un sistema que emplea memoria caché (definida como la razón entre el tiempo de acceso a caché y el tiempo medio de acceso al sistema caché-M.P.) vale 0 cuando la razón de aciertos de la caché vale también 0.

VERDADERO

556-En una arquitectura RISC típica:

se usan muchas instrucciones de las disponibles en el conjunto de instrucciones.

557-Si un byte puede agrupar píxeles consecutivos, un modo de vídeo de 512 x 256 píxeles y 16 colores por píxel ocupa una memoria de:

64 KB

558-La instrucción movl %esp,%ebp

Copia el contenido del registro ESP en el registro EBP.

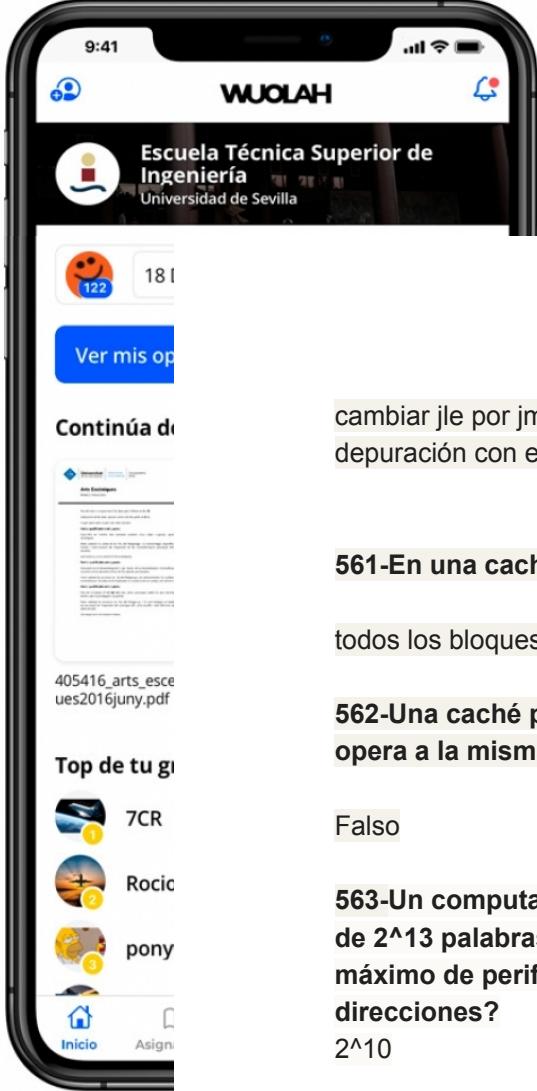
559-Es más caro emplear buses dedicados que no dedicados, pero ello permite establecer más comunicaciones simultáneamente, aumentando las prestaciones de un computador.

VERDADERO

560-En una bomba como las estudiadas en prácticas, del tipo...

```
0x08048705 <main+149>: call 0x80484c4    <gettimeofday>
...
0x08048718 <main+168>: cmp $0x5,%eax
0x0804871b <main+171>: jle 0x8048722 <main+178>
0x0804871d <main+173>: call 0x8048604 <boom>
0x08048722 <main+178>: ...
```

ejecutada paso a paso con el depurador ddd, interesaría...



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

cambiar jle por jmp usando ddd o un editor hex, salvar el programa y reiniciar la depuración con el nuevo ejecutable

561-En una caché asociativa por conjuntos, la vía i está constituida por:

todos los bloques i-ésimos de cada conjunto

562-Una caché puede incrementar las prestaciones de la memoria incluso si opera a la misma velocidad que la memoria principal.

Falso

563-Un computador con 15 líneas de direcciones tiene 3 módulos de memoria de 2^{13} palabras y utiliza E/S mapeada en memoria. ¿Cuál es el número máximo de periféricos que pueden conectarse, si cada uno de ellos utiliza 8 direcciones?

2^{10}

564-Para un tamaño de línea fijo, llega un momento en que la tasa de faltas aumenta al aumentar el tamaño de la memoria caché.

Falso

565-En las transferencias asíncronas, es común acompañar a cada dato transferido de una señal de control que indica la presencia del dato en el bus.
VERDADERO

566-¿En qué tipo de traducción de memoria virtual se utilizan los campos base y límite?

Segmentación

567-En el contexto de una llamada a función cdecl: si los contenidos de ESP y EBP son, respectivamente, 0x0008d040 y 0x00000000 antes de ejecutar una instrucción call, tras ejecutar el ret correspondiente, los contenidos serán: 0x0008d040 y 0x00000000

568-La directiva de ensamblador "DW 100 DUP(0)" reserva 100 palabras inicializadas a cero.

Verdadero

569-En un sistema con dos buses separados, uno para el subsistema de memoria y otro para la E/S...

el bus que une la memoria y el procesador suele funcionar a la velocidad de la memoria

570-El marco de pila en x86-64 Linux...

se crea para funciones en las que GCC no puede evitar que RSP baje más, como por ejemplo: que haya que calcular la dirección de una variable local, o pasar más de 6 argumentos a otra función

571-El objetivo de un diseño CISC es...

disminuir el número de instrucciones a ejecutar por un programa.

572-¿Cuál de las siguientes afirmaciones es falsa?

Las memorias DRAM

- a) presentan generalmente una capacidad de almacenamiento mucho mayor que las SRAM.
- b) La lectura de un bit de la matriz de almacenamiento de una memoria DRAM proporciona una señal mucho más débil que la suministrada por los inversores de una celda de memoria SRAM.
- c) Las memorias DRAM son en general mucho más rápidas que las SRAM
- d) Una celda DRAM sólo necesita un transistor y un condensador.

573-¿Cómo actúa el indicador Z del registro de indicadores de estado?

Se pone a 1 cuando el resultado de una operación es 0

574-Los buses de placa madre ("board level") son relativamente rápidos y no requieren terminadores.

VERDADERO

575-Dada la siguiente definición de datos:

```
lista: .int 0x10000000, 0x50000000,  
        0x10000000, 0x20000000  
longlista: .int (. -lista)/4  
resultado: .quad 0x123456789ABCDEF  
formato: .ascii "suma=%llu=%llx hex\n\0"
```

y suponiendo que hemos llamado a una función suma que devuelve un número de 64 bits en la pareja EDX:EAX, las instrucciones que copian ese número en resultado son:

```
movl %eax, resultado  
movl %edx, resultado+4  
little-endian => primero el menos significativo
```

576-Para construir una DRAM de 4GB con pastillas de 512Mx4bit hacen falta 16 pastillas

577-En una jerarquía de memoria, a medida que nos alejamos del procesador...

El tamaño de la unidad de transferencia entre dos niveles aumenta.

578-¿Cuál de las siguientes afirmaciones es falsa respecto al lenguaje C?

- En lenguaje C, al llamar a una
- a) subrutina o función se introducen los parámetros en la pila y después se realiza una llamada a la subrutina
 - b) Pasar a una función un puntero a una variable se traduce en introducir en la pila el valor de la dirección de memoria donde está almacenada la variable

-
- c) Antes de volver de la rutina llamada, el programa en C se encarga de quitar de la pila los parámetros de llamada realizando varios pop
- X
- d) Los parámetros se introducen en la pila en el orden inverso a como aparecen en la llamada de C, es decir, empezando por el último y acabando por el primero

579-Un computador emplea un sistema de memoria principal de 128 palabras y una memoria caché de 32 palabras. La organización de la memoria caché es totalmente asociativa y el tamaño de bloque es de 8 palabras. Se emplea el algoritmo de reemplazo FIFO. Si inicialmente la memoria caché está totalmente vacía, calcule el número de fallos cuando se lee la secuencia de direcciones de la memoria principal: 0000100, 1000001, 0000101, 0010011, 0100010, 1000100, 0000111.

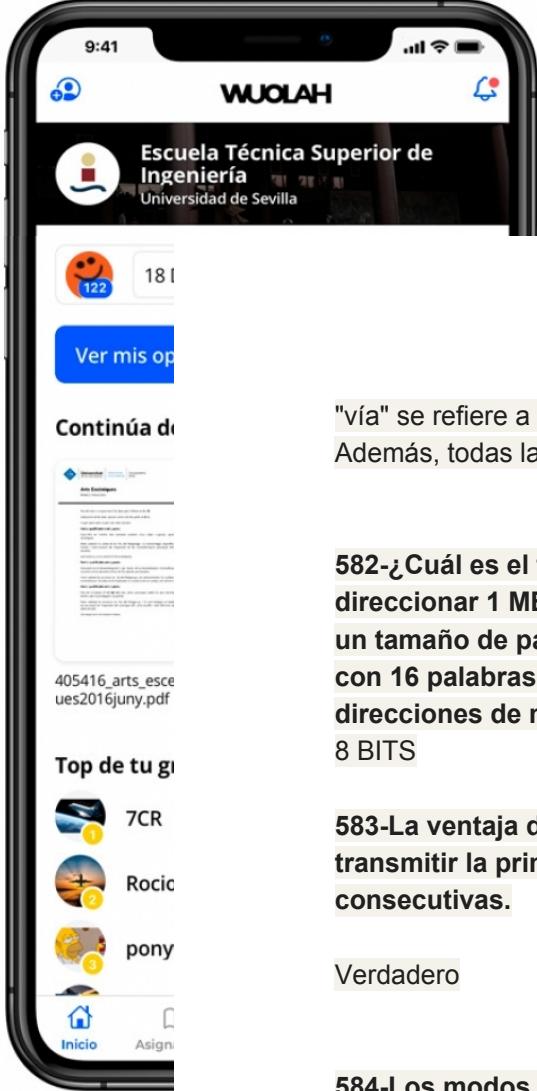
- 6
- a) fallos
- b) 3
fallos
- ✓ • c) 4
fallos
- d) 5
fallos

580-Respecto a direccionamiento a memoria en ensamblador IA32 (sintaxis AT&T), de la forma D(Rb, Ri, S), sólo una de las siguientes afirmaciones es FALSA. ¿Cuál?

Los registros base e índice (Rb y Ri) pueden ser cualesquiera de los 8 registros enteros (EAX...ESP)

581-La política de correspondencia de una memoria cache con la mitad de conjuntos que líneas es:

Asociativa por conjuntos de 2 vías



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

"vía" se refiere a cuántas "alternativas" (líneas) hay en un conjunto.
Además, todas las demás opciones tienen defectos

582-¿Cuál es el tamaño de la marca de caché en un ordenador capaz de direccionar 1 MB de memoria principal y 32 KB de memoria caché, que emplea un tamaño de palabra de 32 bits y correspondencia asociativa por conjuntos con 16 palabras por bloque y 8 bloques por conjunto, suponiendo que las direcciones de memoria utilizan 20 bits?

8 BITS

583-La ventaja de las transferencias de bloques en un bus es que sólo hay que transmitir la primera dirección y se ahorran las siguientes direcciones consecutivas.

Verdadero

584-Los modos de direccionamiento de una instrucción se especifican en ensamblador mediante las directivas.

Falso

585-En las arquitecturas del tipo registro-registro se usan instrucciones LOAD y STORE para transferir información entre registros y memoria principal.

Verdadero

586-El bus de direcciones contiene líneas para indicar el sentido de la transferencia de datos, por ejemplo una línea para distinguir entre lectura y escritura.

Falso

587-Si se necesitan 60 ns para escribir una palabra de datos de cache en memoria principal y cada bloque de cache tiene 8 palabras, ¿cuántas veces "seguidas" (sin que haya reemplazo) se tiene que escribir en un mismo bloque para que una cache de postescritura sea más eficiente que una de escritura inmediata?

No se puede responder con los datos proporcionados

588-Si el proceso de empaquetado de un producto (50 segundos de duración) puede segmentarse en 5 etapas, cada una de 10 segundos, de modo que 5 operarios puedan trabajar cada uno en una etapa, ¿cuál de las siguientes afirmaciones es falsa al aplicar la segmentación?

Cada 50 s saldrá un nuevo producto empaquetado, el mismo tiempo que cuando no había cadena de empaquetamiento

589-Son funciones de la unidad de control la lectura de memoria principal de la instrucción apuntada por el uPC y la codificación de esa instrucción.

Falso

590-Lo que se lee de una dirección de E/S siempre es lo mismo que lo último que se ha escrito en ella.

Falso

591-¿Cuál de las siguientes listas está correctamente ordenada temporalmente?

8088, 80386, Pentium, Pentium Pro,

a) Pentium 4, Pentium II.

X

b) 8088, 80386, Pentium, Pentium II, 80486, Pentium 4.

•

c) 8088, 80486, Pentium, Pentium Pro, Pentium II Pentium 4.

d) 80286, 8088, Pentium, Pentium Pro, Pentium III, Pentium 4.

592-En IA32, ¿cuál de las siguientes afirmaciones es incorrecta?

CALL ETIQUETA guarda en la pila la dirección de retorno y el registro de indicadores

593-Un ciclo de refresco en una memoria DRAM requiere un impulso RAS y un CAS.

FALSO

594-Un computador con 8 bits en el bus de direcciones puede direccionar como máximo:

256 palabras

595-¿Cuál de las siguientes afirmaciones es cierta?

- Los modos de
- a) direccionamiento de una instrucción se especifican en ensamblador mediante las directivas.
 - ✓ • b) El direccionamiento a registro es similar al directo. La diferencia es que el campo de direcciones referencia un registro en lugar de una dirección de memoria principal.
 - c) En el direccionamiento directo, la dirección efectiva se encuentra en un registro de uso general del procesador.
 - d) En el modo de direccionamiento implícito, el objeto direccionado es una constante contenida en la propia instrucción.

596-Un microprocesador es superescalar si tiene un número mayor de etapas y éstas son más cortas que las de un cauce ("pipeline") normal, permitiendo una velocidad de reloj mayor.

FALSO

597-Un mapa de direcciones de memoria es una representación pictórica del espacio de direcciones asignado a cada chip o tipo de memoria en el sistema.

VERDADERO

598-Se dice que las máquinas con arquitectura Von Neumann siguen un modelo de programa...

Almacenado.

599-¿Cuál de las siguientes afirmaciones respecto a la memoria RDRAM no es cierta?

El bus de datos suele ser muy ancho.

600-¿Cuál de las siguientes es una idea fundamental de la jerarquía de memoria?

Que dispositivos más pequeños y rápidos sirvan de cache para dispositivos más grandes y lentos

601-Sea un computador de 32 bits que dispone de una memoria cache de 512 KB y líneas de 64 bytes. ¿Cuántas líneas tiene la cache?

8192

602-Una placa madre de un 486 con un único SIMM de 30 contactos con 8 chips de 1M x 1, tiene 2 M palabras de memoria principal.

Falso

603-Con el repertorio IA32, para sumar %eax y %ebx dejando el resultado en %ecx se podría hacer lo siguiente:

lea (%eax, %ebx, 1), %ecx

604-El registro MDR/MBR...

contiene el valor que va a ser almacenado en la memoria, o bien se usa para recibir un valor procedente de la memoria

606-Respecto a requisitos de alineamiento de structs en gcc/IA32 x86 y x86-64, alguna de las siguientes afirmaciones es falsa

en x86-64 Linux alinea float a 8x (Windows también)

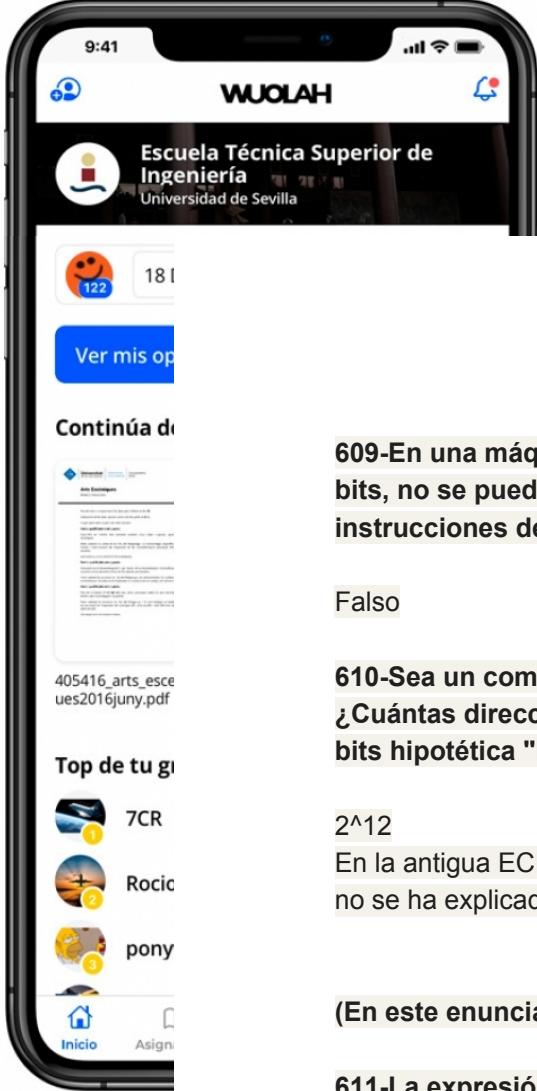
redacción original era "x86-64 Linux long double 8x, Windows no", motivo por el cual notamos errata transparencias (no dicen nada de x86-64 Windows, y dicen que x86-64 Linux long double 8x, cuando libro pág.325 dice que 16x). Posible corrección: "alinea long double a 12x (Windows no)". Redacción final es aún más claramente errónea.

607-¿En qué generación, dentro de la historia de los computadores digitales, aparece la microprogramación?

tercera

608-Respecto a los conceptos de interfaz de dispositivo, controlador(a), puerto de E/S:

La controladora o interfaz contiene los puertos necesarios para utilizar el dispositivo



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

609-En una máquina con 32 registros direccionables e instrucciones de 16 bits, no se pueden codificar 32 instrucciones de dos registros, 32 instrucciones de 1 registro y 32 instrucciones de 0 direcciones.

Falso

610-Sea un computador con 48 registros y 200 instrucciones máquina. ¿Cuántas direcciones de memoria permite el formato de la instrucción de 32 bits hipotética "beqz r1, r2, dir"?

2^{12}

En la antigua ECI se hacían estos cálculos, códigos Hamming, etc. En la nueva EC no se ha explicado con tanto detalle. Evitar esta pregunta.

(En este enunciado el símbolo \wedge representa potenciación)

611-La expresión que cumplen las direcciones de una memoria de bytes en las que queda mal alineada una palabra de 32 bits es Dirección mod 4 != 0.
VERDADERO

612-La instrucción NOP no se usa nunca, ya que no hace nada.

FALSO

613-Para que el modo de direccionamiento indirecto a través de registro coincida con el modo de direccionamiento indexado, el campo de desplazamiento en el direccionamiento indexado ha de ser igual al contenido del registro, pero cambiado de signo.

Falso

614-Respecto al sistema de Entrada / Salida, ¿cuál de las siguientes afirmaciones es incorrecta?

- a) La CPU se comunica con el periférico por medio del controlador y de software de E/S.
- b) Un protocolo sirve para “ponerse de acuerdo” en cosas como velocidad, paridad, nº de bits, etc.

- - c) Un controlador se encarga de la comunicación con la CPU.
 - d) La mayoría de los periféricos trabajan a velocidad muy superior a la CPU; por eso es necesario sincronizar.

615-En las transacciones de bloques del bus PCI se envía una dirección por cada dato, ya que direcciones y datos no están multiplexados en el tiempo.

Falso

616-La codificación Huffman utiliza menos bits para las instrucciones menos frecuentes y más bits para las instrucciones más utilizadas.

FALSO

617-La gestión distribuida del arbitraje consiste en un árbitro central del bus al que llegan múltiples líneas de petición de varios maestros potenciales y del que salen las correspondientes señales de concesión.

FALSO

618-Una memoria SRAM tiene una capacidad de 64 Kbits y utiliza 12 líneas para direccionamiento. Indique cuál es el tamaño de palabra de dicha memoria:

16 bits

619-Una dirección de memoria se refiere siempre a:

a)
una palabra

b)
16 bits

c)
un byte
 d)
ninguna de las anteriores

620-La idea fundamental en la que se basan los computadores RISC es minimizar el número de instrucciones necesarias para realizar una tarea determinada.

F

621-En la ejecución de una instrucción...

la ALU realiza las operaciones aritméticas y lógicas

622-¿Para qué se utiliza la función gettimeofday en la práctica de la "bomba digital"?

Para lanzar un error cuando el usuario tarde demasiado tiempo en introducir la contraseña o el PIN

623-Respecto a si un computador dispone de E/S independiente (separada) o usa E/S mapeada a memoria:

Si el repertorio del procesador tiene instrucciones del tipo IN y OUT, es que el computador dispone de E/S separada

624- Compilar .s → ejecutable, usando sólo as y ld, sin gcc...

Se puede, usando en as y ld los modificadores (switches) que corresponda

625-Si AX = 0xFA50 y ejecutamos XOR \$0xFF, AX

Se realiza el complemento a 1 de AL.

626-Dado un camino de datos concreto, un posible formato de microprogramación se caracteriza como horizontal o vertical según tenga más o menos (señalar la respuesta falsa)

microbifurcaciones

627-En una arquitectura Von Neumann cada palabra de memoria contiene un campo que permite diferenciar entre instrucción o dato.

F

628-Se puede construir procesadores de varios anchos de palabra usando varios chips de sección de 4 bits.

V

629-El 8086 sólo permite direccionar directamente 640 KB de memoria. Por eso los PCs basados en este chip no disponían de más memoria RAM.

F

630-Al llamar a una función de 2 argumentos foo(arg1, arg2), ¿cuál es el orden correcto de las operaciones? (suponiendo convención de llamada x86 cdecl, y que foo requiere ajustar marco de pila, esto es, salvar %ebp)
push arg2, push arg1, call foo, push %ebp

631-En la E/S controlada por interrupciones:

La CPU transfiere el control a una rutina de servicio cuando recibe una interrupción.

632-Los computadores con estructura Von Neumann utilizan el modelo de programa externo, frente a programa cableado y almacenado.

F

633-Un computador que utilice el sistema big-endian, almacena el número 2143h a partir de la dirección 0 como:

21h en el byte de la dirección 0 y 43h en el byte de la dirección

1

634-La microprogramación vertical se caracteriza por tener:
muchas codificaciones

635-Se dispone de un computador cuyo tiempo medio de acceso al sistema de memoria caché y memoria principal es de 18 ns. Si la tasa de fallo de la caché es de 0,2 y el tiempo de acceso a la memoria principal es 50 ns. ¿Cuál es el tiempo de acceso a la caché?

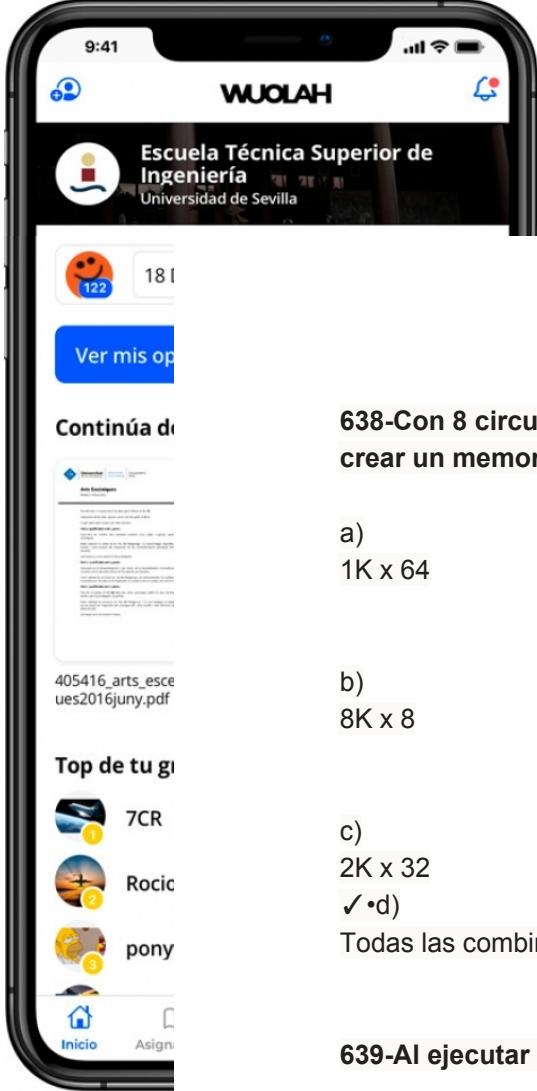
8 ns

636-Siempre que se utiliza "polling" la prioridad de los dispositivos interruptores queda fijada mediante encadenamiento ("daisy-chain").

Falso

637-¿Cuál de las siguientes afirmaciones es cierta?

El modo de direccionamiento indirecto a través de registro es el modo más sencillo de direccionamiento a memoria.



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

638-Con 8 circuitos de memoria RAM de 1K x 8 se puede crear un memoria de:

- a)
1K x 64

- b)
8K x 8

- c)
2K x 32
- ✓ d)
Todas las combinaciones anteriores son posibles

639-Al ejecutar el fragmento de código:

```
leal -48(%eax), %edx
cmpl $9, %edx
ja .L2
```

se salta a .L2 si el contenido del registro %eax:

está fuera del intervalo [48,57]

640-Cada bit de una RAM dinámica ha de ser refrescado cada pocos nanosegundos.

Falso

641-Cuál de los siguientes no es un modo de direccionamiento IA32?

- a)
Cache
no existe, ver [T2.1.3ConASM] tr.27
- b)

Registro

c)
Inmediato

d)
Memoria

642-¿Cuál de las siguientes afirmaciones acerca de las interrupciones en el PC es cierta?

Ninguna de las otras afirmaciones es cierta

643-Si se dice que en un sistema computador cada dirección especifica uno o dos puertos de E/S, se refiere a que:

Un puerto será de sólo lectura, otro de sólo escritura, y ambos se decodifican en la misma dirección

644-Un bus se compone de:

Líneas de control/estado, líneas de dirección y líneas de datos

645-El Intel 8086:

Incluía instrucciones de multiplicación.

646-El fragmento de código:

```
poll: in a, 0x20
      cmp a, $0
      jnz poll
      load a, 0x11
      out 0x21, a
```

corresponde a:

E/S programada con consulta de estado

647-La idea de la arquitectura RISC se debe, entre otros, a:

John Cocke

648-¿Qué necesitamos para construir una memoria de 1K x 8 bits?

8 memorias de 256 x 4 bits y un decodificador de 2 a 4

649-¿Cuál de los siguientes grupos de instrucciones sólo modifican los indicadores de estado sin almacenar el resultado de la operación?

CMP, TEST

650-Un TLB (buffer de traducción anticipada) tiene un número de entradas o elementos mucho menor que el número de páginas de la memoria virtual.

V

651-En una arquitectura little-endian el código de operación aparece al final de la instrucción.

F

652-Motivos que impiden que la ganancia (aceleración) de un cauce segmentado sea ideal (señalar la respuesta falsa)

cola de instrucciones (precaptación)

653-Nunca puede utilizarse "polling" para identificar la fuente de una interrupción en un sistema con "daisy-chain"

F

654-SCSI son las siglas de Small Computer System Interface (interfaz del sistema para computadores pequeños).

Verdadero

655-Un procesador cuya instrucción CALL guardara la

dirección de retorno en un registro RL (llamado "de enlace"):

No permitiría llamadas anidadas ni recursivas.

656-En un procesador RISC, las bifurcaciones no degradan las prestaciones del "pipeline".

F

657-Una de las funciones de una interfaz de E/S es generar señales de temporización cuando se necesiten, según el esquema que se utilice el subbus de control.

658-¿Cuál afirmación es FALSA al comparar las arquitecturas x86 y x86-64?

659-A medida que aumenta el tamaño de página en un sistema de memoria virtual, ¿qué ocurre con el tamaño de las tablas de páginas?

Disminuye

660-¿En qué tipo de memoria virtual es un problema la fragmentación externa?

661-Se tiene una memoria que emplea entrelazado. Si fallan varias celdas contiguas de uno de sus chips de memoria, ¿con qué tipo de entrelazado de memoria sería más fácil poder utilizarla?

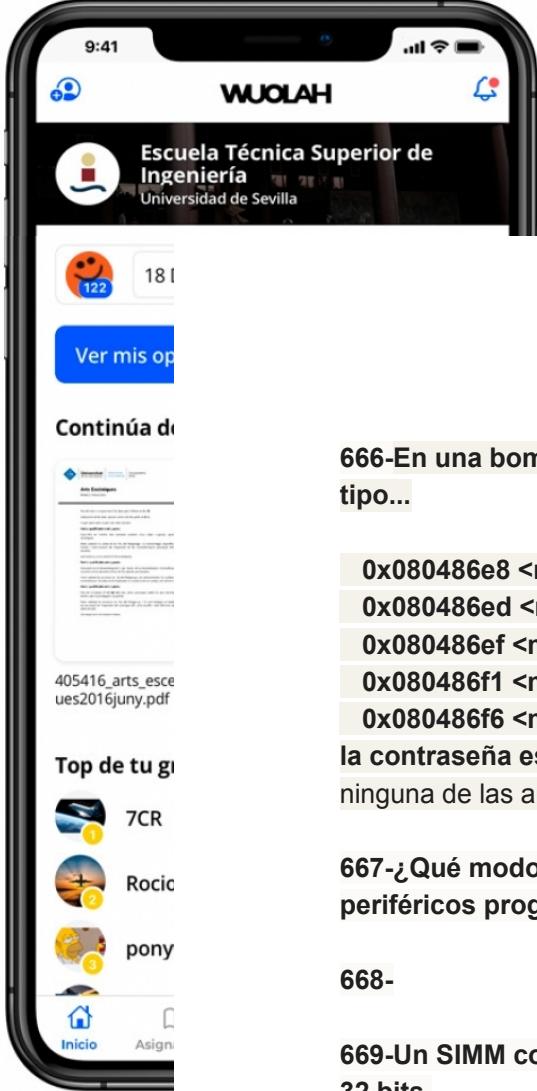
662-Una cache de 256 B asociativa por conjuntos de 4-vías con líneas de 16 B tendría

4 conjuntos

663-Para diseñar una memoria con ancho de palabra $k \cdot m$ (y mismo nº palabras que los módulos) a partir de módulos con ancho de palabra m , se utilizan k módulos (mediante _ se representa subíndice)

664-El Pentium III es superescalar porque permite "emitir" varias instrucciones por ciclo de reloj.

665-¿Cómo actúa el indicador N del registro de indicadores de estado?



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

666-En una bomba como las estudiadas en prácticas, del tipo...

```
0x080486e8 <main+120>: call 0x8048524 <strncmp>
0x080486ed <main+125>: test %eax,%eax
0x080486ef <main+127>: je 0x80486f6 <main+134>
0x080486f1 <main+129>: call 0x8048604 <boom>
0x080486f6 <main+134>: ...
```

la contraseña es...

ninguna de las anteriores

667-¿Qué modo de funcionamiento permite a la interfaz de periféricos programable 8255 utilizar un bus bidireccional?

668-

669-Un SIMM con 8 chips de 4Mx1 contiene 1 M palabras de 32 bits.

F

670-En el contexto de las DRAM, la activación de la señal CAS se emplea para realizar ciclos de refresco.

671-La "postescritura ("write-back") marcada" es más eficiente que la "postescritura siempre".

V

672-La cache es gestionada por:
unidades de "manejo" (gestión) hardware

673-Un modo de direccionamiento en el que se especifica un registro y una dirección de memoria cuyo contenido se suma al contenido del registro base para obtener la dirección efectiva, se conoce como:

- a)base con desplazamiento
- b)absoluto o directo
- c)indirecto a registro a través de memoria
- d)ninguna de las anteriores*****

674-Si la estructura struct a ocupa un espacio de 28 bytes

en memoria, ¿cuántos bytes ocupa la siguiente estructura struct b cuando se compila en 64 bits?

```
struct b {  
    struct a a1;  
    int i;  
    struct a a2;  
}  
60 bytes
```

675-En un sistema de memoria con entrelazado de orden inferior y M módulos de memoria la dirección de memoria a está en el módulo a mod M.

V

676-Alguna de las siguientes señales no sirve de entrada a la unidad de control. ¿Cuál?

contador de programa (bits del registro PC)

677-¿Cuál es la característica tecnológica principal de la tercera generación de computadores?

Los circuitos integrados

678-¿Cuál de las siguientes secuencias de instrucciones multiplica el (contenido del) registro EAX por 18?

```
leal (%eax,%eax,8), %eax  
leal (%eax,%eax), %eax
```

679-Un sistema no segmentado tarda 20 ns en procesar una tarea. La misma tarea puede ser procesada en un cauce (pipeline) de 4 segmentos con un ciclo de reloj de 6 ns. Cuando se procesan muchas tareas, la ganancia máxima de velocidad que se obtiene se aproxima a:

3,33

680-Al traducir la sentencia C r->i = val; gcc genera el código ASM movl %edx, 12(%eax). Se deduce que

el desplazamiento de i en *r es 12
y en concreto r está en %eax

681-El computador EDVAC, propuesto por John Von Neumann, presentaba dos importantes diferencias respecto al ENIAC:

Empleaba aritmética binaria y permitía trabajar con un programa almacenado.

682-Cada celda de un chip de memoria DRAM de 1M x 1, organizada en una matriz de 512 filas x 2048 columnas, necesita ser refrescada cada 16 ms. ¿Cada cuánto tiempo ha de realizarse una operación de refresco en el chip?

31,25 microsegundos

683-El lenguaje máquina...

a)

es un conjunto de nombres simbólicos o nemotécnicos.

b)

facilita la portabilidad de los programas.

c)

es el mismo para todos los computadores.

✓ •d)

Ninguna de las respuestas anteriores es correcta.

684-La instrucción negl:

realiza el complemento a dos

685-Una caché totalmente asociativa es equivalente a una asociativa por conjuntos de una vía.

Falso

686-En la secuencia de programa siguiente:

804854e:e8 3d 06 00 00 call 8048b90 <main>

8048553:50 pushl %eax

¿cuál es el valor que introduce en la pila la instrucción call?

0x8048553

687-Una máquina que almacene el número 4321h a partir de la dirección 0 como 21h, 43h utiliza el sistema big-endian.

Falso

688-Las instrucciones que asignan la dirección 0x78e00 a un puntero situado en la posición 0xff00 son:

mov \$0x78e00,%eax

mov %eax,0xff00

689-En general, un operación segmentada ("pipelined") requiere el mismo tiempo o más, desde el principio hasta el fin y para un único par de operandos, que la misma operación en una implementación no segmentada.

V

690-La penalización por una falta de página suele ser de cientos de miles de ciclos de reloj de la CPU.

Verdadero

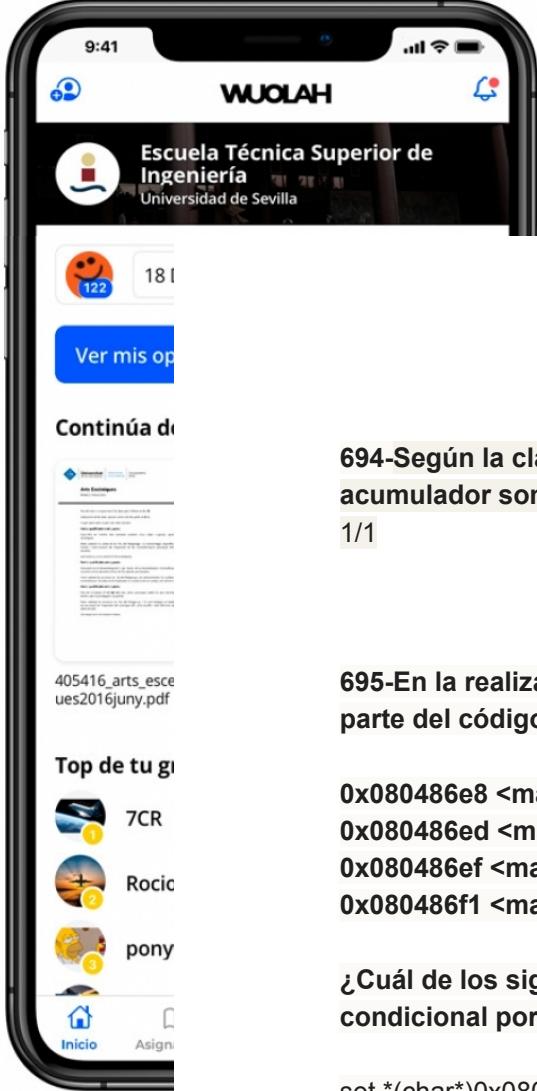
691-En el direccionamiento inmediato el operando reside en:

en la instrucción tras el código de operación

692-El bus del sistema es

el que conecta CPU-M, ya sea un sistema con bus único o con múltiples buses

693-La CPU guarda automáticamente el contexto del programa que ejecuta cuando le llega y atiende una petición de DMA.



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

694-Según la clasificación m/n, las máquinas de acumulador son de tipo

1/1

695-En la realización de la práctica de la bomba digital, una parte del código máquina es el siguiente:

```
0x080486e8 <main+120>: call 0x8048524 <strcmp>
0x080486ed <main+125>: test %eax,%eax
0x080486ef <main+127>: je 0x80486f6 <main+134>
0x080486f1 <main+129>: call 0x8048604 <boom>
```

¿Cuál de los siguientes comandos cambiaría el salto condicional por un salto incondicional?

set *(char*)0x080486ef=0xeb

696-Respecto a las bombas estudiadas en la práctica "bomba digital", ¿en cuál de los siguientes tipos de bomba sería más fácil descubrir la(s) contraseña(s)? Se distingue entre strings definidos en el código fuente de la bomba, y strings solicitados al usuario mediante scanf(). Por "cifrar" se entiende aplicar la cifra del César (sumar o restar una constante fija a los códigos ASCII).

1 string del fuente se cifra, y se compara con el string del usuario

aunque hubieran sido 2, aunque su hubieran cifrado y concatenado o al revés, sigue siendo igual de fácil, porque con el debugger se puede ver qué hay que poner de contraseña literalmente

697-Un archivo .o que contiene código objeto:

Contiene instrucciones máquina binarias.

698-¿En qué tipo de transferencias es necesario establecer un periodo de tiempo máximo después del cual se considera que ha fallado?

En las transferencias asíncronas

699-La instrucción MOVSB del 8086 transfiere el byte apuntado por ES:DI al acumulador AL.

FALSO

700-En el bus PCI las direcciones y los datos están multiplexados en el tiempo, y existen transacciones de bloques de datos en las que se envía una única dirección y se envían (o reciben) muchos datos.

Verdadero

701-¿Qué tipo de localidad de las referencias a memoria se define como: "si se referencia un elemento, volverá a ser referenciado pronto"?

Localidad temporal

702-Después de ejecutar una instrucción de suma sobre dos números con signo de la que sabemos que no provocará overflow (los dos números son pequeños en valor absoluto), queremos comprobar si el resultado de la suma es menor que 0. ¿Qué flag necesita comprobar la instrucción de salto condicional equivalente a “if (resultado<0) then goto label”?

SF

703-Un bus de ciclo partido está disponible para otro maestro en las ranuras entre la petición de una lectura y la contestación con el dato.

VERDADERO

704-En una caché con 64 bytes de longitud de línea, ¿qué bits de una dirección de memoria se utilizan para determinar a qué byte dentro de la línea se refiere dicha dirección?

Los 6 bits menos significativos

705-La memoria caché del computador es:

a)

Más rápida que la memoria principal

b)

De menor capacidad que la memoria principal

✓ •c)

a) y b) son correctas

d)

Ninguna de las anteriores es correcta

706-Una jerarquía de memoria consta de una cache de con una tasa de aciertos del 92% y 4 ns de tiempo de acceso y una memoria principal con una tasa de aciertos del 100% y 100 ns de tiempo de acceso. ¿Cuál es el tiempo promedio estimado de acceso a memoria?

12 ns

$0.92 \cdot 4\text{ns} + 0.08 \cdot (100+4)\text{ns}$

707-El bus GPIB utiliza handshake de 6 flancos. Esto es necesario para que un emisor pueda enviar datos a varios receptores distinguiéndose el hecho de que el dato haya sido leído por todos del hecho de que todos estén listos para recibir un nuevo dato.

VERDADERO

708-Respecto a registros salva-invocante y salva-invocado en GCC/Linux IA32, ¿cuál de éstos es de distinto tipo que el resto?

EAX

709-Para la memoria virtual se suele utilizar correspondencia directa

FALSO

710-Un programa de ordenador que convierte un programa fuente de alto nivel completo en lenguaje máquina se llama un:

compilador

711-¿Cómo se devuelve en ensamblador x86-64 Linux gcc el valor de retorno de una función long int al terminar ésta?

- a) La instrucción RET lo almacena en un registro especial de retorno.
- b) Por convención se guarda en %eax.
- c) Se almacena en pila justo encima de los argumentos de la función
- d) **CORRECTA->**Ninguna de esas formas es la correcta

712-¿Qué política de memoria virtual para colocar nuevos segmentos en los huecos libres de la memoria principal evita el que se generen huecos pequeños?

Peor ajuste

713-¿Cuántas señales de control se necesitan como mínimo para implementar un sistema de gestión de interrupciones?

2

714-El tiempo de acceso de un SIMM es mayor que el de un único chip, debido al retraso de los decodificadores que contiene para generar los distintos "Chip Select".

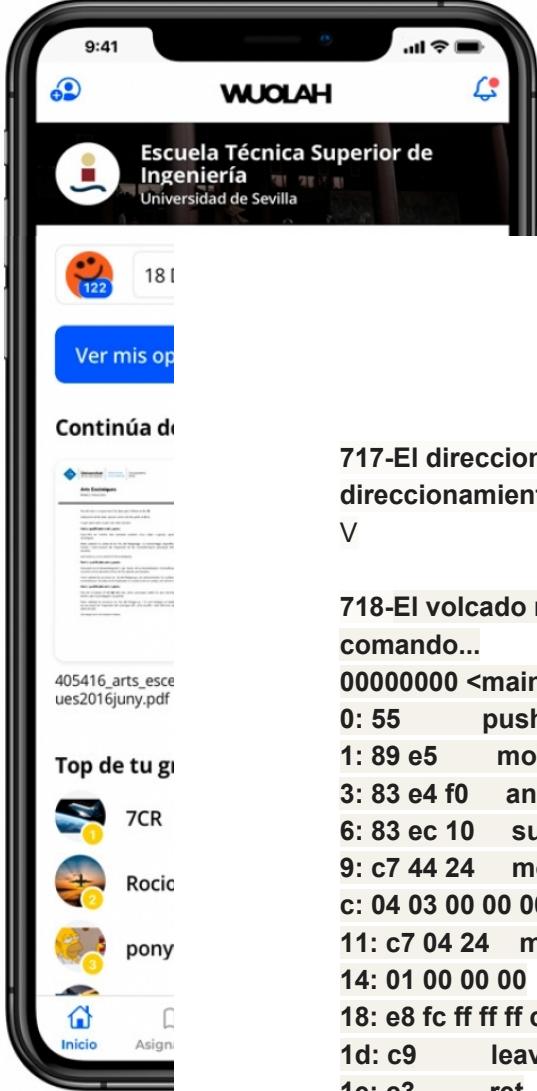
FALSO

715-Usar un procesador más rápido siempre implica un incremento proporcional en las prestaciones de un computador, incluso si la velocidad de la memoria principal permanece inalterada.

FALSO

716-El resultado de desplazar aritméticamente dos posiciones hacia la derecha el número de 8 bits en complemento a dos –32 es:

-8



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

Continúa di

717-El direccionamiento inmediato no es más lento que el direccionamiento directo.

V

718-El volcado mostrado abajo se ha obtenido con el comando...

00000000 <main>:

```
0: 55      push %ebp
1: 89 e5    mov %esp,%ebp
3: 83 e4 f0  and $-16,%esp
6: 83 ec 10  sub $0x10,%esp
9: c7 44 24  movl $3, 4(%esp)
c: 04 03 00 00 00
11: c7 04 24  movl $0x1,(%esp)
14: 01 00 00 00
18: e8 fc ff ff call <main+0x19>
1d: c9        leave
1e: c3        ret
```

objdump -d p2.o

-d porque es un desensamblado (no cabeceras -h ni tablas -t)

719-La conexión entre un dispositivo de E/S y el procesador mediante bus:

Permite conectar en paralelo varios dispositivos

720-Según el concepto de máquina virtual de Tanenbaum, el nivel de máquina convencional consiste en una computadora hipotética con un determinado lenguaje máquina, cuyos programas son interpretados por la máquina virtual del nivel de microprogramación, supuesto que éste existe.

Verdadero

721-Si desplazamos a la izquierda un registro 3 posiciones:

Lo multiplicamos por 8

722-Dada la siguiente definición de datos:

```
lista: .int 0x10000000, 0x50000000,  
      0x10000000, 0x20000000  
longlista: .int (. -lista)/4  
resultado: .quad 0x123456789ABCDEF  
formato: .ascii "suma=%llu=%llx hex\n\n\0"
```

```
push resultado+4  
push resultado  
push resultado+4  
push resultado  
push $formato  
call printf  
add $20, %esp  
Sí.  $20 = 8(\%llx) + 8(\%llu) + 4(\$formato)$ 
```

723-Alguno de los siguientes no es un nombre de registro en una máquina IA32 en modo 32 bits

sil
Sí lo sería en modo 64 bits

724-Las instrucciones máquina contienen toda la información necesaria para ejecutarse, y además su interpretación es independiente de la posición que ocupan en el programa.

Verdadero

725-Para realizar la microoperación MAR <- PC, habrá que activar:

EnPC y LdMAR

726-El primer nivel de una jerarquía de memoria tiene una tasa de aciertos del 75% y las peticiones de memoria tardan 12 ns en completarse si dicha posición se encuentra en ese nivel y 100 ns si no es así. ¿Cuál es el tiempo medio de acceso de la jerarquía?

34 ns

727-¿En qué registro se pasa el primer argumento a una función en Linux gcc x86-64?

edi

728-¿Cuál de las siguientes secuencias de instrucciones multiplica %eax por 10?

a)

leal(%eax,%eax,4), %eax
sal \$2, %eax
sería $x5x4=x20$

b)

imull \$0x10, %eax
sería x16

c)

addl %eax, %eax
shll \$5, %eax
sería $x2x32=x64$

✓ •d)

Varias o ninguna de las respuestas anteriores son correctas, no se puede marcar una y sólo una

729-¿Qué política de colocación en caché necesita más comparadores, la correspondencia asociativa por conjuntos o la correspondencia por sectores?

Depende de si es mayor el número de bloques por conjunto o el número de sectores

730-Dada la siguiente definición de datos:

lista: .int 0x10000000, 0x50000000,
 0x10000000, 0x20000000

longlista: .int (.lista)/4

resultado: .quad 0x123456789ABCDEF

formato: .ascii "suma=%llu=%llx hex\n\0"

la instrucción movl longlista, %ecx copia el siguiente valor:

4

Práct.2, Tut, pág.7

731-¿Cuál de las siguientes características es típica de la microprogramación horizontal?

Ninguna o escasa codificación.

732-HIJO DE PUTA

733-En una memoria con entrelazado de orden inferior, M módulos y acceso simultáneo, el tiempo de acceso a las $2M$ palabras cuyas direcciones son $0, 1, \dots, 2M-1$ es menor o igual que $3Ta$, siendo Ta el tiempo de acceso a cada módulo de memoria.

Verdadero

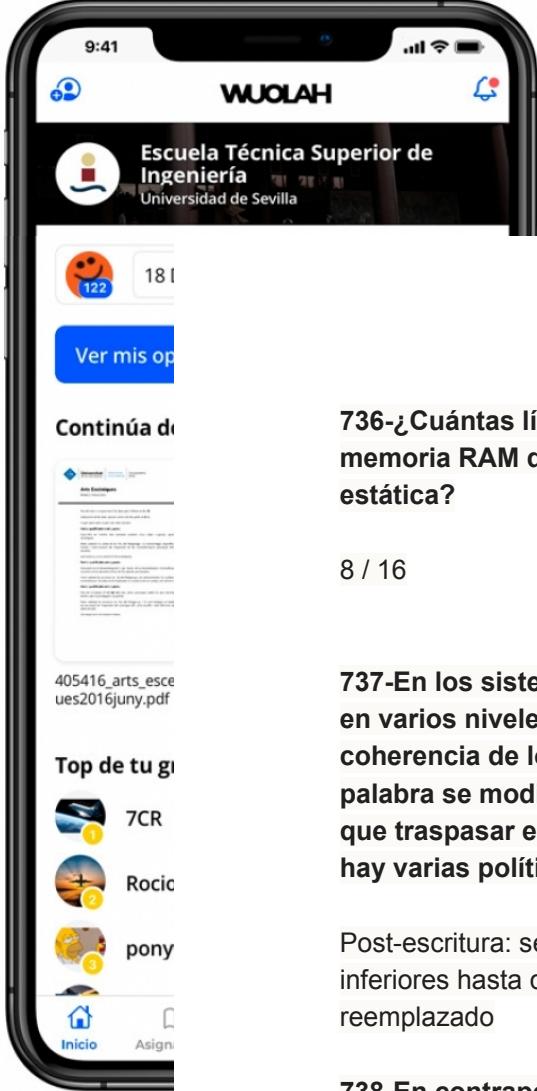
734-Los buses síncronos y semisíncronos disponen de una línea de reloj "maestro", y todas las actividades del bus tienen una duración igual a un número entero de ciclos de ese reloj.

Verdadero

735-Suponiendo que todos los registros inicialmente contienen el valor 1, ¿cuál es el valor de ECX tras la ejecución de la siguiente secuencia de instrucciones?

```
mov $4, %eax  
mov $3, %ebx  
lea (%eax, %eax), %ecx  
sub %ebx, %ecx  
imul %ecx, %eax
```

5



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

736-¿Cuántas líneas de dirección son necesarias en un memoria RAM dinámica de 64 K palabras? ¿Y en una estática?

8 / 16

737-En los sistemas con una jerarquía de memoria dividida en varios niveles se da el problema de la consistencia o coherencia de los datos entre los distintos niveles. Si una palabra se modifica en un nivel, en algún momento habrá que traspasar ese cambio a los niveles inferiores. Para ello hay varias políticas:

Post-escritura: se retrasa la actualización en los niveles inferiores hasta que el bloque modificado tenga que ser reemplazado

738-En contraposición a un ejecutable Linux ELF, un fichero objeto (obtenido con gcc -c)

ubica el código (y los datos) a partir de la posición 0x0, las direcciones definitivas sólo se calculan tras enlazar

739-Un Pentium 4 a 3,2 GHz dispone de 7 unidades de ejecución en paralelo, con 20 etapas de segmentación, y es capaz de emitir (comenzar a ejecutar) 3 instrucciones en cada ciclo de reloj. ¿Qué velocidad aproximada de ejecución de instrucciones será capaz de alcanzar (MIPS = millones de instrucciones por segundo)?

9000 MIPS

740-En IA32, ¿cuál de los siguientes fragmentos de programa tiene un efecto sobre los flags distinto al resto?

mov \$-1, %edi
mov no afecta a los flags

741-Un sistema no segmentado tarda 200 ns en procesar una tarea. La misma tarea puede ser procesada en un cauce segmentado de 20 etapas con un ciclo de reloj de 12 ns. Cuando se procesan muchas tareas, la máxima ganancia de velocidad que podría obtenerse se acerca a:

16,67

742-Las memorias caché con política de extracción anticipativa (prebúsqueda o preextracción) no soportan búsqueda por demanda.

Falso

743-Si el tiempo de acceso a la memoria cache es de 2 ns y el tiempo necesario para tratar un fallo de cache es de 80 ns, ¿cuál es la tasa de aciertos necesaria para que el tiempo medio de acceso al sistema de memoria sea de 10 ns?

0.9

744-La cache con correspondencia directa se puede considerar como un caso límite de la asociativa por conjuntos, en donde...

745-Un overflow nunca puede ocurrir cuando:

•d)
se suma un número positivo a un número negativo
resultado queda entre ambos => se puede representar

746-El algoritmo LRU ("Menos Recientemente Usado/a") se puede utilizar tanto en memoria caché como en sistemas de memoria virtual con paginación.

Verdadero

747-En una CPU de 32 bits con memoria de bytes, el problema es que.

Hay que respetar el ordenamiento de bytes y reglas de alineamiento con que se diseñó la CPU

748-La primera instrucción ensamblador de una subrutina compilada con gcc en Linux/x86 cdecl suele ser:

push %ebp

749-¿En qué generación, dentro de la historia de los computadores digitales, aparecieron la microprogramación, la segmentación de cauce, la memoria cache, los S.O. multiusuario y la memoria virtual?

3^a generación (1965-75)
en la 3^a generación se inventó casi todo

750-Un cauce ("pipeline") de instrucciones con 5 etapas tarda 7 ciclos de reloj o más en ejecutar 3 instrucciones si éstas utilizan las cinco etapas.

Verdadero

751-PC y SP son dos registros de uso general (GPR).

Falso

752-Si se necesitan 60 ns para escribir una palabra de datos de caché en memoria principal y cada bloque de caché tiene 8 palabras, ¿cuántas veces seguidas se tiene que escribir en un mismo bloque para que una caché de postescritura sea más eficiente que una de escritura inmediata?

Depende de la tasa de aciertos

753-¿Cuál de las siguientes afirmaciones acerca de las

interrupciones en el PC (modo real) es cierta?

a)

Cada vector de interrupción es una palabra de 16 bits

b)

Existen 1024 vectores de interrupción

c)

La tabla de interrupciones tiene un tamaño de 256 bytes

• d)

Todas las interrupciones se pueden generar por software



Legendario_115795

www.wuolah.com/student/Legendario_115795



Test EC tema3.pdf

Tests de EC



2º Estructura de Computadores



Grado en Ingeniería Informática



Escuela Técnica Superior de Ingenierías Informática y de
Telecomunicación
Universidad de Granada



Descarga la APP de Wuolah.
Ya disponible para el móvil y la tablet.





**KEEP
CALM
AND
ESTUDIA
UN POQUITO**

PREGUNTAS TEST EC TEMA 3

1. ¿Cómo actúa el indicador de signo?

- a) Se pone a 1 cuando el resultado es positivo
- > b) Se pone a 1 cuando el resultado es negativo
- c) Se pone a 1 cuando el resultado es distinto
- d) Se pone a 0 cuando el resultado es negativo

2. La microprogramación vertical se caracteriza por tener:

- a) escaso o ningún solapamiento entre campos
- b) capacidad para expresar un alto grado de paralelismo en las microoperaciones a ejecutar
- c) microinstrucciones largas
- > d) mucha codificación

3. Un procesador con una unidad de control microprogramada tiene una memoria de control de 256 palabras de 16 bits, de las que 128 son diferentes. ¿Qué ahorro en número de bits obtendríamos si usáramos nanoprogramación?

- > a) 256 bits
- b) No se produce ahorro
- c) 4096 bits
- d) 3840 bits

4. La conexión de las salidas de tres registros hacia un bus común en el camino de datos puede realizarse usando...

- > a) dos multiplexores de 2 a 1
- b) tres conexiones directas al bus común
- c) tres demultiplexores
- d) dos buffers triestado

5. Sea un formato de microinstrucción que incluye dos campos independientes de 9 bits cada uno. Si se rediseña de modo que se solapen los dos campos, ¿cuántos bits se ahorran en cada microinstrucción?

- a) 1
- b) 4
- > c) 8
- d) 9

6. ¿Cómo actúa el indicador N del registro de indicadores de estado?

- > a) Se pone a 1 cuando el resultado es negativo.
- b) Se pone a 1 cuando el resultado es positivo.
- c) Se pone a 1 cuando el resultado de una operación es 0.
- d) Se pone a 0 cuando el resultado es negativo.

7. Los procesadores comerciales con unidad de control microprogramada suelen almacenar los micropogramas...

- a) en una RAM.
- b) en un banco de registros.
- > c) en una ROM.
- d) en una PLA.

8. Respecto a MBR y MAR

- a) Ambos permiten guardar información sobre el marco de pila //sin relación directa
- b) Ambos son accesibles por el programador //ninguno
- c) MAR contiene el dato/instrucción que se leerá o escribirá en memoria //ese es MBR
- > d) MAR requiere menos señales de control que MBR //en Tema 3 tr.10, sólo una (LoadMAR),

//frente a 4 de MBR (Load/Enable Mem/Bus)

9. Un computador usa el formato vertical de codificación de instrucciones para parte de las señales de control y el formato horizontal para k señales de control. El formato vertical posee n campos codificados de m bits cada uno. ¿Cuál es el máximo número de señales de control que pueden usarse en este computador?

- a) $k + n \cdot 2^m$
- b) $k + n^m$
- > c) $k + n \cdot (2^m - 1)$
- d) Ninguno de los anteriores

10. Un computador tiene una memoria de control de 640 palabras de 70 bits, de las que 280 son diferentes. ¿Qué ahorro en número de bits obtendríamos si usáramos nanoprogramación en lugar de microprogramación?

- >a) 19440
- b) 42280
- c) 9840
- d) ninguno de los anteriores resultados es exacto.

11. Un computador tiene una memoria de control de 16000 palabras de 250 bits, de las que 447 son diferentes. ¿Cuántos bits ahorramos usando nanoprogramación en lugar de micropogramación?

- > a) 3744250
- b) 259206
- c) 287935
- d) ninguno de los resultados anteriores es exacto

12. En un camino de datos con un solo bus, para realizar la operación de copia de un registro r1 en un registro r2, es decir $r2 \leftarrow r1$, es necesario:

- a) Activar la carga del registro r1 y habilitar la salida triestado del registro r2
- b) Habilitar la salida triestado del registro r2 y activar la carga de los registros r1 y r2
- c) Habilitar las salidas triestado de los registros r1 y r2 y activar la carga del registro r2
- > d) Habilitar la salida triestado del registro r1 y activar la carga del registro r2

13. Respecto a las unidades de control nanoprogramadas:

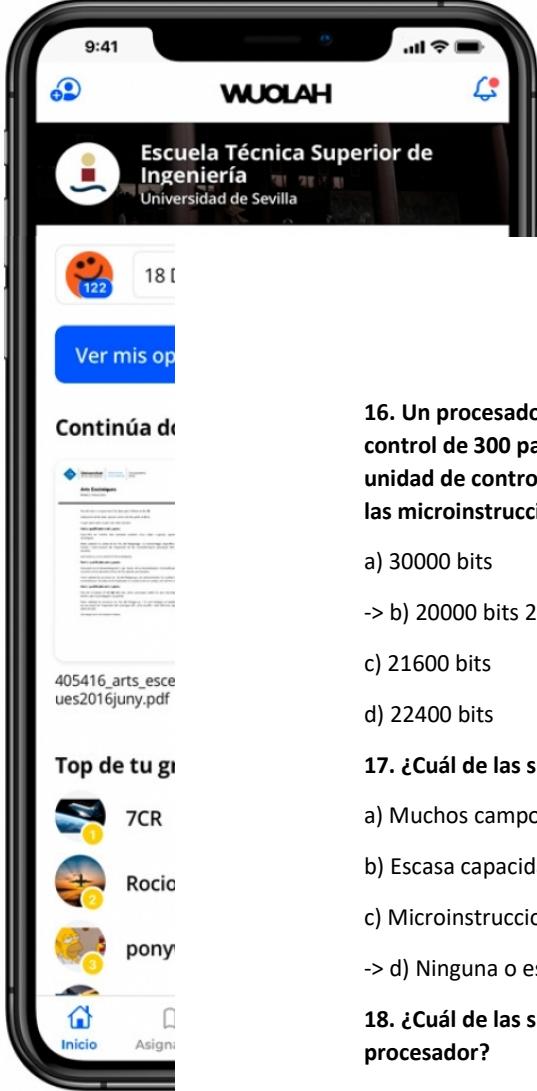
- > a) La anchura de la memoria de nanoprograma es la misma que la de memoria de micropograma en un diseño de la misma unidad de control que no usara nanoprogramación.
- b) La realización nanoprogramada de una unidad de control es más rápida que la microprogramada.
- c) El diseño de las unidades de control nanoprogramadas debe ser vertical.
- d) Suponiendo una memoria de micropograma con n microinstrucciones de w bits cada una, de las cuales 2^m son distintas, el ahorro en bits si se utiliza nanoprogramación es $(n \cdot m + 2^m \cdot w) - n \cdot w$.

14. En el pseudocódigo usado para representar las microinstrucciones, la expresión “goto f(IR)”:

- a) Permite saltar a la dirección de memoria de control del principio de un microbucle.
- b) Se utiliza para realizar un microsalto condicional en función del registro de estado.
- c) Realiza una llamada a una microsubrutina.
- > d) Salta a una dirección de memoria de control que depende de la instrucción máquina actual.

15. Para conectar las salidas de dos registros hacia un bus común en el datapath...

- > a) se pueden usar dos buffers triestado.
- b) no se puede usar un multiplexor.
- c) se puede realizar una conexión directa.
- d) se puede usar un demultiplexor.



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

16. Un procesador con una unidad de control microprogramada tiene una memoria de control de 300 palabras de 100 bits, de las que 200 son diferentes. Si se rediseñara como unidad de control nanoprogramada, ¿qué tamaño ocuparía la nanomemoria que contiene las microinstrucciones completas sin repeticiones?

- a) 30000 bits
- > b) 20000 bits 200 uinstr. x 100 bits
- c) 21600 bits
- d) 22400 bits

17. ¿Cuál de las siguientes características es típica de la microprogramación horizontal?

- a) Muchos campos solapados.
- b) Escasa capacidad para expresar paralelismo entre microoperaciones.
- c) Microinstrucciones cortas.
- > d) Ninguna o escasa codificación.

18. ¿Cuál de las siguientes funciones no corresponde a la unidad de control de un procesador?

- > a) Calculo de operaciones de coma flotante
- b) Decodificación de las instrucciones
- c) Secuenciamiento de las instrucciones
- d) Generación de las señales de control que provocan la ejecución de cada instrucción

19. Para el procesador con unidad de control microprogramada estudiado en clase, Tanenbaum define 15 codops de 4bits para instrucciones con un operando dirección de 12bits (LODD M, STOD M, ADDD M, ...), y en lugar de gastar el último codop 1111 en otra instrucción con una dirección, realiza una extensión de codop de 3bits para definir ocho instrucciones más. En general esas instrucciones no tienen ningún operando, salvo INSP Y (sp+=Y) y DESP Y (sp-=Y), que tienen un operando inmediato.

a) Y es un operando de 8 bits y no sobra espacio de codificación (huecos) para modificaciones sin aprovechar bit8

-> b) Y es un valor de 8 bits pero podría definirse que fuera de 9 bits sin ninguna dificultad

c) Y es una dirección corta de 9 bits y podría definirse que fuera de 10 u 11 bits cambiando la codificación por extensión

// 11bits imposible, sólo cabrían INSP/DESP, no habría sitio para PUSH/POP etc

d) Y ocupa 10 bits, pero podría ser de 11 bits cambiando la codificación por extensión y uniendo ambas instrucciones en una sola ADDSP Y (sp+=Y) en donde Y se interpretara como número con signo en complemento a dos

// Y no ocupa 10bits.

//Sí que sería posible ADDSP codop 1111 1yyy yyyy yyyy con 11bit para Y, y entonces codop //1111 0ccc 0000 0000 permite de sobra los 6 codops restantes. [T3.4Tnbaum] tr.79

20. Una unidad de control microprogramada con secuenciamiento explícito con dos direcciones por microinstrucción, tiene una memoria de control con 35 bits de longitud de palabra. Si las microinstrucciones emplean 15 bits en total para los campos de control y de tipo y condición de salto, el número máximo de palabras de la memoria de control de esta unidad de control microprogramada es de:

- a) 10
- b) 2^{20}
- > c) 2^{10}
- d) 20

21. ¿Qué circuito suele utilizarse para traducir el código de operación de una instrucción máquina a dirección de comienzo en la memoria de control del microprograma correspondiente?

- > a) Una memoria.
- b) Un registro.
- c) Un multiplexor.
- d) Un contador.

22. En una arquitectura RISC típica:

- a) no puede usarse segmentación.
- b) la programación resulta mucho más simple que en una arquitectura CISC.
- > c) se usa un porcentaje elevado de las instrucciones del repertorio.
- d) la UC es más compleja que en una arquitectura CISC.

//Evitar esta pregunta (de la antigua titulación), las transparencias que hablan de RISC están //ahora repartidas por los distintos temas

23. Sea un formato de microinstrucción que incluye dos campos independientes de 10 bits cada uno. Si se rediseña de modo que se solapen los dos campos, ¿cuántos bits se ahorran en cada microinstrucción?

- a) 14
- b) 13
- c) 10
- >d) 9

24. La salida de un campo del registro de microinstrucción que solapa dirección de salto y algunas señales de control han de conectarse a:

- a) una ROM o PLA
- >b) un demultiplexor controlado por el tipo de salto
- c) el registro de instrucción
- d) la memoria de control

25. Una unidad de control microprogramada se denomina "con secuenciamiento de microinstrucciones explícito" según tenga o no tenga

- a) ROM/PLA para traducir el codop en dirección de inicio de microprograma (goto f(IR))
//todos los diseños implícitos en las transparencias usan ROM traducción, pero también se
//podría añadir ROM al explícito
- b) microcódigo de decodificación que analice el codop bit a bit de izquierda a derecha
//con ROM traductora (goto f(IR)) no haría falta ese análisis
- > c) micro-contador de programa atacando a las líneas de dirección de la memoria de control
// si y sólo si tiene micro-PC, sería implícito
- d) un multiplexor para seleccionar la fuente de la dirección de la memoria de control
//todos los diseños en las transparencias tienen un MUX en la dirección de la memoria de
//control (o en el micro-PC)

26. ¿Cuál de las siguientes afirmaciones es verdadera?

- > a) La unidad de control necesita como entrada el registro de estado para poder controlar la ejecución de las instrucciones de salto condicional.
- b) El registro puntero de pila es un registro de propósito general que suele contener tanto direcciones como datos.
- c) Las únicas instrucciones en las que algunas de sus fases de ejecución llevan un acceso a memoria son las instrucciones load y store.
- d) El registro de instrucción es un registro de propósito específico que contiene la dirección de la siguiente instrucción a ejecutar.

27. En una unidad de control microprogramada con formato de microinstrucciones vertical, un subcampo que deba especificar 16 señales de control codificadas de tal forma que pueda activarse sólo una o ninguna habrá de tener una anchura mínima de

- a) 17 bits
- b) 16 bits
- > c) 5 bits
- d) 4 bits

28. Para realizar la microoperación MAR <- PC, habrá que activar:

- a) EnPC y EnMAR
- b) LdPC y EnMAR
- > c) EnPC y LdMAR
- d) LdPC y LdMAR

29. Una posible codificación en microinstrucciones de la instrucción CALL X es:

- a) SP=SP-1 ; m[SP]=PC ; PC=PC+1
- > b) SP=SP-1 ; m[SP]=PC ; PC=X
- c) PC=X ; SP=SP-1 ; m[SP]=PC
- d) SP=PC-1 ; m[SP]=PC ; PC=X

30. Dado un camino de datos concreto, un posible formato de microprogramación se caracteriza como horizontal o vertical según tenga más o menos (señalar la respuesta falsa)

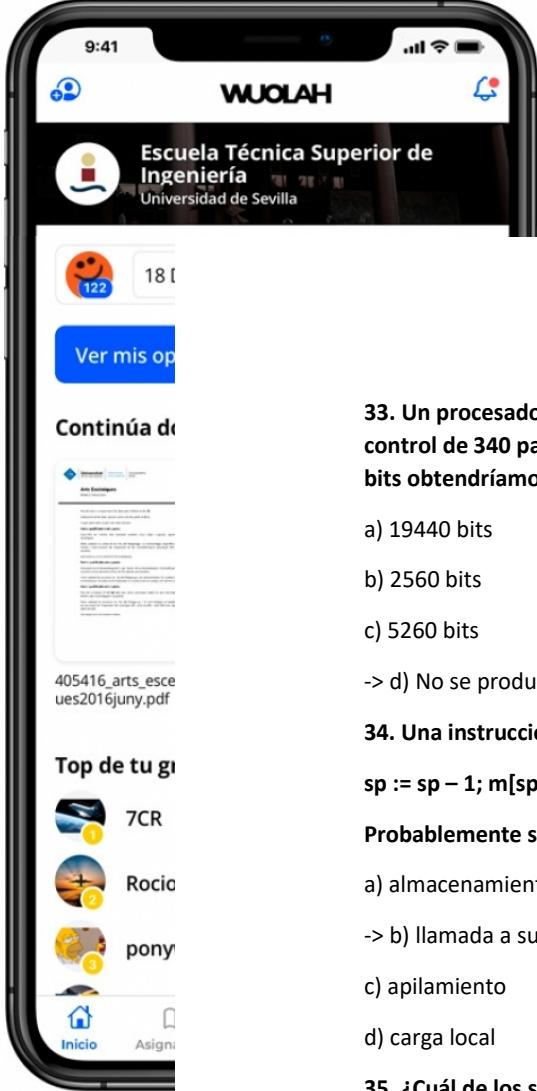
- a) longitud relativa de microinstrucción
- > b) microbifurcaciones
- c) solapamiento
- d) codificación

31. Alguna de las siguientes *no* es una operación básica de la unidad de control:

- a) realizar operación ALU y guardar resultado en registro
- b) transferir un registro a otro
- > c) (guardar o recuperar) un registro (en / de) la pila
- d) (leer o escribir) un registro (de / a) memoria

32. Un computador tiene una memoria de control de 16 Kpalabras de 250 bits, de las que 447 son diferentes. ¿Cuántos bits ahorramos usando nanoprogramación en lugar de microprogramación?

- a) 3928652
- b) 259206
- c) 287935
- > d) ninguno de los resultados anteriores es exacto



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

33. Un procesador con una unidad de control microprogramada tiene una memoria de control de 340 palabras de 16 bits, de las que 180 son diferentes. ¿Qué ahorro en número de bits obtendríamos si usáramos nanoprogramación?

a) 19440 bits

b) 2560 bits

c) 5260 bits

-> d) No se produce ahorro

34. Una instrucción máquina puede desglosarse en las siguientes operaciones elementales:

$sp := sp - 1; m[sp] := pc; pc := x$

Probablemente se trate de una instrucción de:

a) almacenamiento local

-> b) llamada a subrutina

c) apilamiento

d) carga local

35. ¿Cuál de los siguientes grupos de señales son entradas a la unidad de control?

a) Las señales de carga/incremento/desplazamiento de registros

b) Las señales de selección de entradas de multiplexores del datapath

-> c) Los bits del registro de indicadores (flags)

d) Los bits de las opciones b y c

36. Respecto a los términos microinstrucción y microcódigo:

a) Son equivalentes, llamamos microcódigo o microinstrucción a una palabra de la memoria de control

b) Una microinstrucción está programada en microcódigo, que es un lenguaje para programar señales de control

c) Un microcódigo controla una serie de señales de control relacionadas (por ejemplo, el código 000 para que la ALU realice la suma), y varios juntos forman una microinstrucción

-> d) Microcódigo es el contenido de la memoria de control, y una microinstrucción es una palabra de dicha memoria

37. El control residual se utiliza para:

a) reducir el tiempo de ejecución de las instrucciones máquina

b) eliminar los bits residuales de la ejecución de los microinstrucciones

-> c) reducir el tamaño de la memoria de control

d) ninguna de las anteriores es cierta

38. Son funciones de la unidad de control:

- a) la codificación de las instrucciones máquina
- b) la lectura de memoria principal de la instrucción apuntada por el µPC
- > c) el secuenciamiento de las instrucciones máquina
- d) todas las respuestas son ciertas

39. En cuanto al control microprogramado:

- a) se guardan en una ROM las instrucciones máquina del conjunto de instrucciones.
- b) la UC se construye con puertas lógicas, biestables, etc.
- > c) se usa en CISC para facilitar el diseño de la UC tan compleja.
- d) la lentitud en la ejecución de las instrucciones máquina la impone directamente la tecnología hardware usada.

40. Alguna de las siguientes señales no sirve de entrada a la unidad de control. ¿Cuál?

- a) señal de reloj (CLK)
- b) instrucción actual (bits del registro IR)
- c) estado de la unidad de proceso (flags Z, S, C, O...)
- > d) contador de programa (bits del registro PC)

41. [T3.1] ¿Cuál de las siguientes funciones es una tarea propia de la unidad de control en la CPU?

- > a) decodificar las instrucciones del programa
- b) almacenar instrucciones del programa
- c) realizar operaciones lógicas
- d) almacenar datos del programa

42. En una arquitectura RISC típica:

- a) se usan pocas instrucciones de las disponibles en el conjunto de instrucciones
- > b) suele usarse segmentación
- c) la programación resulta mucho más simple que en una arquitectura CISC
- d) la UC es más compleja que en una arquitectura CISC

43. ¿Cómo actúa el indicador Z del registro de indicadores de estado?

- a) Se pone a 1 cuando el resultado es negativo
- b) Se pone a 0 cuando el resultado es negativo
- > c) Se pone a 1 cuando el resultado de una operación es 0
- d) Se pone a 1 cuando el resultado es positivo

44. Alguna de las siguientes señales no es salida de la unidad de control. ¿Cuál?

- a) señales de lectura y escritura en memoria (RD, WR)
- > b) dirección de la siguiente microinstrucción (bits del campo DIR o Memoria B de Wilkes)
- c) códigos de selección en multiplexores, decodificadores, ALU, etc (00, 01, 10, 11...)
- d) señales de carga, habilitación y/o desplazamiento de registros (Load, Enable, ShiftL, ShiftR)

45. Sea un formato de microinstrucción que incluye dos campos independientes de 8 bits cada uno. Si se rediseña de modo que se solapen los dos campos, ¿cuántos bits se ahorran en cada microinstrucción?

- a) 1
- b) 8
- c) 9
- > d) 7

WUOLAH



CAZZ

www.wuolah.com/student/CAZZ

23184

testsswad.pdf

Pack de test resueltos SWAD



2º Estructura de Computadores



Grado en Ingeniería Informática



**Escuela Técnica Superior de Ingenierías Informática y de
Telecomunicación
Universidad de Granada**



**Descarga la APP de Wuolah.
Ya disponible para el móvil y la tablet.**





**KEEP
CALM
AND
ESTUDIA
UN POQUITO**

RESPUESTAS TEST SWAD

1 ¿Qué circuito suele utilizarse para traducir el código de operación de una instrucción máquina a dirección de comienzo en la memoria de control del microprograma correspondiente?

Elección única Usuario Profesores

- a) Una memoria.
- b) Un contador.
- X c) Un multiplexor.
- d) Un registro.

Puntuación: -0,33

2 [T5.1]

Elección única Alguna de las siguientes NO es una ventaja de la E/S independiente (separada, aislada)

Usuario Profesores

- ✓ • a) Diseño del procesador más sencillo (E/S mapeada añade complejidad al diseño)
- b) Mayor aprovechamiento del espacio de memoria (E/S mapeada resta espacio a la memoria)
- c) Protección de E/S más fácil (E/S mapeada añade dificultad a la protección de E/S)
- d) Decodificación de memoria más elegante, limpia, sencilla (E/S mapeada añade complejidad a la decodificación)

Puntuación: 1,00

[T5.1FunE/S]

[E14SepTeo09]

3 [P2.2]

Elección única En la práctica "media" se programa la suma de una lista de 32 enteros de 4 B para producir un resultado de 8 B, primero sin signo y luego con signo. Si la lista se rellena con el valor que se indica a continuación, ¿en qué caso ambos programas producen el mismo resultado?

Usuario Profesores

- X a) 0xFFFF FFFF

- 0x0000 001f ffff ffe0 != 0xffff ffff ffff ffe0
- b) 0xAAAA AAAA
 0x0000 0015 5555 5540 != 0xffff fff5 5555 5540
- c) 0x9999 9999
 0x0000 0013 3333 3320 != 0xffff fff3 3333 3320
- 0x1111 1111
- resultado 0x0000 0002 2222 2220
- porque es positivo incluso en complemento a 2
 - d) todos los demás valores se interpretan como negativos, lo primero que hace la suma con signo es extenderlos a 64bit de manera que se activan los 32 bits superiores... resultado radicalmente distinto

Puntuación: -0,33

[P2.2SumSgn]

[E16SepPra06]

Recordar que multiplicar por 32 es desplazar 5 posiciones a la izquierda

- 4 [T3.3]
 Elección ¿Cómo actúa el indicador Z del registro de indicadores de estado?
 única Usuario Profesores
- ✓ • a) Se pone a 1 cuando el resultado es negativo
 b) Se pone a 0 cuando el resultado es negativo
 c) Se pone a 1 cuando el resultado de una operación es 0
 d) Se pone a 1 cuando el resultado es positivo

Puntuación: 1,00

[T3.3CtrlUp]

[E12SepTeo19]

- 5 ¿Es posible utilizar 4 GB de memoria en un sistema cuya CPU emplea E/S mapeada en memoria, cuyo bus de direcciones es de 32 bits y que tiene al menos un puerto de E/S? Supondremos que no se puede emplear ninguna técnica de extensión del bus de direcciones.
 Elección
 única Usuario Profesores
- X a) Sí
 • b) No

- c) Depende de si el número puertos de E/S es muy elevado
- d) Ninguna de las respuestas anteriores es correcta

Puntuación: -0,33

6 Si R0=2, R1=5 y M[3]=3 ¿Qué valor toman R0, R1 y M[3] tras ejecutarse la instrucción XOR 1h[R0],R1?
 Elección única Usuario Profesores

- a) R0=6 , R1=2 , M[3]=5
- b) R0=6 , R1=5 , M[3]=2
- c) R0=2 , R1=5 , M[3]=6
- d) R0=5 , R1=6 , M[3]=2

Puntuación: 0,00

7 [T4.3]
 Elección única En un procesador con segmentación de cauce, aumentar el número de etapas (p.ej. de 2 a 4, o de 4 a 8), tiene en general como consecuencia:
 Usuario Profesores

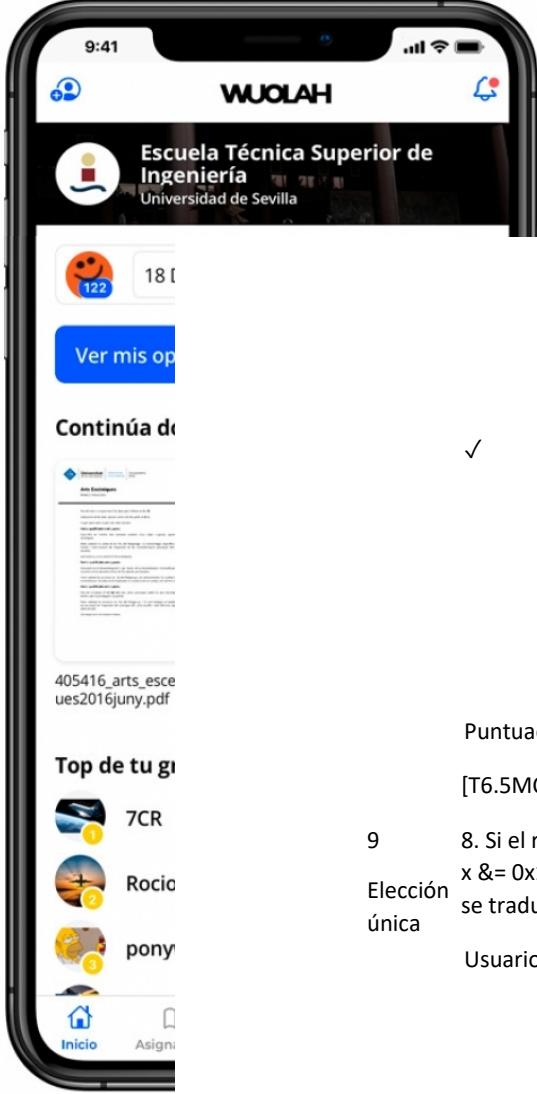
- a) Una disminución de la máxima frecuencia de reloj a la que puede operar el cauce
- b) Una disminución en la posible dependencia de datos
- c) Un mayor retraso en la ejecución de los programas debido al incremento del número de etapas
- ✓ • d) Un incremento de las prestaciones

Puntuación: 1,00

[T4.3Aceler]
 [E13FebTeo30]
 [E17JulTeo14]

8 [T6.5]
 Elección única En los sistemas con una jerarquía de memoria dividida en varios niveles se da el problema de la consistencia o coherencia de los datos entre los distintos niveles. Si una palabra se modifica en un nivel, en algún momento habrá que traspasar ese cambio a los niveles inferiores (más lejanos al procesador). Para ello hay varias políticas:

Usuario Profesores



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

- Post-escritura: se retrasa la actualización en los niveles inferiores hasta que el bloque modificado tenga que ser reemplazado
- a) Escritura indirecta: si se modifica una palabra, inmediatamente se modifican los niveles superiores
 - b) Las respuestas a y b son ciertas
 - c) Las respuestas a y b son falsas

Puntuación: 1,00

[T6.5MCache]

- 9 8. Si el registro EAX contiene X, la sentencia en C
x &= 0x1;
se traducirá a ensamblador como:

Elección Usuario Profesores

- a) sarl %eax
- b) shrl %eax
- c) orl \$0x1, %eax
- d) andl \$1, %eax

✓ •

Puntuación: 1,00

- 10 Respecto a las políticas de arbitraje de buses...

Elección Usuario Profesores
única

- a) pueden ser dinámicas o estáticas
- b) las políticas estáticas reparten el uso del bus por turnos de forma fija entre los dispositivos activos
- c) las políticas dinámicas permiten cambiar el maestro del bus en función de la situación en cada instante
- d) Todas las repuestas anteriores son ciertas

✓

•

Puntuación: 1,00

- 11 ¿Qué tipo de direccionamiento se usa para el registro destino en la instrucción mov bx, 5h?

Elección Usuario Profesores

X

- a) Direccionamiento relativo a registro base
- b) Direccionamiento inmediato

- c) Direccionamiento implícito
- d) Direccionamiento directo a registro

Puntuación: -0,33

12 [T5.3]
Elección única ¿A qué tipo de interrupciones pertenecen las condiciones de tiempo real y los fallos hardware?

Usuario Profesores

- a) Enmascarables
- ✓ • b) No enmascarables
- c) Puede ser tanto enmascarables como no enmascarables
- d) Ninguna de las respuestas anteriores es correcta

Puntuación: 1,00

[T5.3ES_IRQ]

13 [T.2.3.1]
Elección única Para crear espacio en la pila para variables locales sin inicializar suele realizarse la siguiente operación:

Usuario Profesores

- a) Sumar una cantidad positiva a ESP
- ✓ • b) Restar una cantidad positiva a ESP
- c) Sumar una cantidad positiva a EBP
- d) Restar una cantidad positiva a EBP

Puntuación: 1,00

[T2.3.1MarcoP]
[E16FebTeo12]

14 Respecto al sistema de Entrada / Salida, ¿cuál de las siguientes afirmaciones es incorrecta?
Elección única Usuario Profesores

- a) La CPU se comunica con el periférico por medio del controlador y de software de E/S.
- b) Un controlador se encarga de la comunicación con la CPU.
- c) Un protocolo sirve para “ponerse de acuerdo” en cosas como velocidad, paridad, nº de bits, etc.

- ✓ • d) La mayoría de los periféricos trabajan a velocidad muy superior a la CPU; por eso es necesario sincronizar.

Puntuación: 1,00

15 En el contexto del lenguaje máquina, el acrónimo ISA suele referirse a:

Elección única Usuario Profesores

- X a) Intel Standard Architecture
b) Industry Standard Architecture
c) Information Security Architecture
• d) Instruction Set Architecture

Puntuación: -0,33

16 [T5.1]

Elección única ¿Cuál de las siguientes funciones no corresponde a un controlador (interfaz) de E/S?
Usuario Profesores

- ✓ • a) Almacenamiento temporal de datos
b) Almacenamiento de programas
c) Comunicación con el dispositivo
d) Comunicación con el microprocesador

Puntuación: 1,00

[T5.1FunE/S]

[E12FebTeo21]

[E12SepTeo12]

17 [T5.1]

Respecto a los conceptos de interfaz de dispositivo, controlador(a), puerto de E/S:
Elección única Usuario Profesores

- a) El interfaz contiene las controladoras necesarias para conectar los puertos con el procesador
b) El puerto, o interfaz, contiene los controladores necesarios para comunicar el dispositivo con el procesador
c) Cada puerto o interfaz es una línea de comunicación con el procesador. El conjunto de ellos forma el controlador.
✓ • d) La controladora o interfaz contiene los puertos necesarios para utilizar el dispositivo

Puntuación: 1,00

[T5.1FunE/S]

[E17FebTeo19]

18 [T2.2.2]

Elección única La extensión de signo a m bits de un número original N de n bits, con m > n, consiste en:

Usuario Profesores

- a) Incrementar la cantidad de bits a m rellenando con unos por la izquierda.
- b) Incrementar la cantidad de bits a m preservando el signo y el valor del número.
- c) Realizar la operación $2^m - N - 1$
- d) Realizar la operación $2^m - N$

Puntuación: 1,00

[T2.2.2OpArit]

[P2Apendice2]

[E17JulTeo02]

19 ¿Cuál de las siguientes afirmaciones acerca del concepto de interrupción es falsa?

Elección única Usuario Profesores

- a) Ninguna de las afirmaciones es falsa

- b) Es una bifurcación normalmente externa al programa en ejecución

- c) Su objetivo es reclamar la atención del procesador

- d) Solicita que se ejecute un programa específico para tratarla

Puntuación: 1,00

20 La conexión de las salidas de tres registros hacia un bus común en el camino de datos puede realizarse usando...

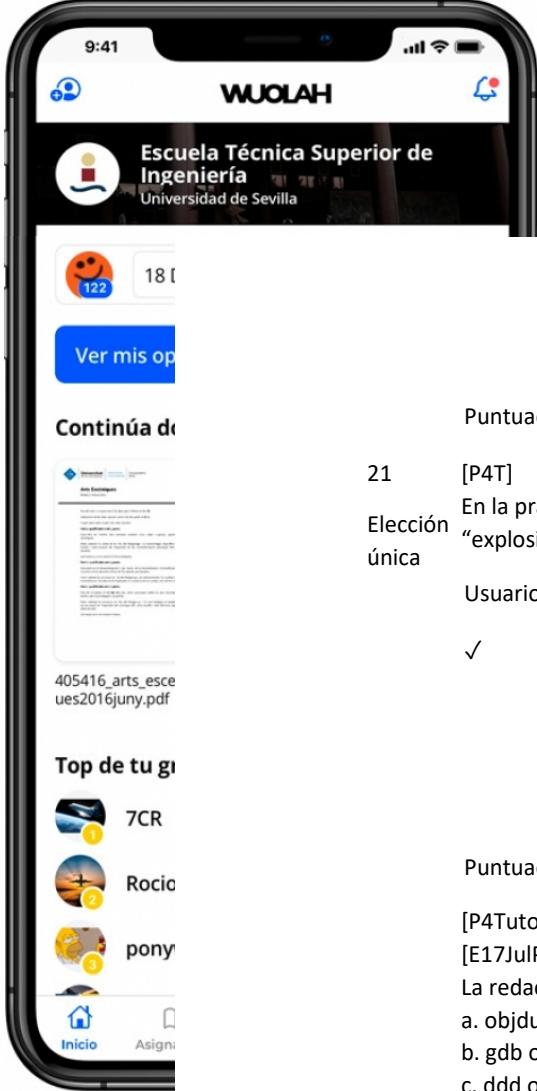
Elección única Usuario Profesores

- a) dos multiplexores de 2 a 1

- b) dos buffers triestado

- c) tres conexiones directas al bus común

- d) tres demultiplexores



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

Continúa d'

Puntuación: -0,33

21 [P4T]

Elección única En la práctica de la bomba, el segundo ejercicio consistía en crear un ejecutable sin "explosiones", para lo cual se puede utilizar... (marcar la opción *falsa*)

Usuario Profesores

- ✓ • a) objdump
b) gdb
c) ddd
d) hexedit

405416_arts_escue2016juny.pdf

Top de tu grupo

| | | |
|--|---|--------|
| | 1 | 7CR |
| | 2 | Rocio |
| | 3 | pony |
| | | Inicio |

Puntuación: 1,00

[P4Tutorial]

[E17JulPra16]

La redacción original era: ...para lo cual se puede utilizar...

- a. objdump o gdb
b. gdb o ddd
c. ddd o hexedit
d. hexedit u objdump

Con esa redacción, hubo que dar por válidas b. y c. Con depuradores se usaría el método explicado en P4T ejercicio 2, y con hexedit el método explicado en ejercicio 3. Es cierto que hexedit a secas (sin objdump ni depurador) no permite saber dónde hay que modificar nada, pero todo el mundo dice (correctamente) que con hexedit se puede crear el ejecutable sin explosiones, así que esta redacción es menos problemática.

22 [T6.2]

Elección única ¿Cuántas líneas de dirección (patillas) son necesarias para direccionar un chip de memoria DRAM de 4096 x 4?

Usuario Profesores

- X a) 12
b) 11
c) 10
• d) 6

Puntuación: -0,33

[T6.2RAMROM]

[E14SepTeo28]

- 23 [T2.1.2]
 Elección Un archivo .o que contiene código objeto:
 única Usuario Profesores
- a) Puede ejecutarse directamente.
 - b) Incluye el código de las funciones de biblioteca a las que llame.
 - c) Contiene las direcciones definitivas de las variables globales.
 - d) Contiene instrucciones máquina binarias.

Puntuación: 1,00

[T2.1.2Lngjes]
 [P2Tutorial]

- 24 [T5.1]
 Elección Parecidos y diferencias entre los métodos de E/S (señalar la opción incorrecta)
 única Usuario Profesores
- a) sólo E/S por DMA libera a la CPU de realizar la transferencia de los datos de E/S
 - b) sólo E/S por DMA libera a la CPU de realizar la consulta de estado del dispositivo
 - c) no cubre el caso de E/S programada para dispositivos "sin estado" (ej: display 7 segmentos en las diapositivas)
 - d) se suele avisar a la CPU (con una IRQ) de que debe realizar alguna tarea, tanto en E/S por IRQ como en E/S por DMA
 - e) cubre los casos de IRQ de dispositivo e IRQ de DMA
 - f) la consulta del estado del dispositivo por parte de la CPU se suele/puede incluir en E/S programada y en E/S por IRQ
 - g) originalmente redactado "se suele/puede hacer con E/S programada y con E/S IRQ" - redacción actual más clara, cubre los casos de bucle de espera ocupada y varios dispositivos conectados a la misma IRQ

Puntuación: 1,00

[T5.1FunE/S]
 [E15SepTeo22]

- 25 [T6.3]
 ¿Qué conjunto de componentes permite construir una memoria 256Mx32? (sin que sobren componentes)

Elección Usuario Profesores
única

- a) 16 chips 64Mx4
- b) 32 chips 64Mx4
- c) 16 chips 64Mx16
- d) Ninguna de las anteriores

Puntuación: 1,00

[T6.3Diseño]

[E14FebTeo26]

26 Una instrucción de salto si igual tiene que comprobar el valor de:

Elección Usuario Profesores
única

- a) los bits de signo y desbordamiento
- b) el bit de signo
- c) el bit de cero
- d) el bit de acarreo

Puntuación: 1,00

27 [T6.2]

Elección Usuario Profesores
única Una SRAM de 32Kx8bit (256Kbit) puede venir organizada en 512 filas, dedicando por tanto al decodificador de columnas...

Usuario Profesores

- a) 9 bits
- b) 8 bits
- c) 7 bits
- d) 6 bits

Puntuación: -0,33

[T6.2RAMROM]

[E13SepTeo014]

28 [T4.3]

Elección Usuario Profesores
única Un sistema no segmentado tarda 10 ns en procesar una instrucción. Las instrucciones pueden ser procesadas en un cauce (pipeline) de 5 segmentos con un ciclo de reloj de 4 ns. Cuando se procesan muchas instrucciones, la ganancia máxima de velocidad que se obtiene se acerca a:

Usuario Profesores

- a) 2,5
- b) 50
- c) 4
- X d) 5

Puntuación: -0,33

[T4.3Aceler]

- 29 [T5.1]
 Elección Señale cuál de las siguientes opciones es una técnica habitual para llevar a cabo la transferencia de datos entre el computador y los dispositivos de E/S externos:

Única Usuario Profesores

- a) Acceso indirecto a memoria (IMA)
- b) Acceso directo a memoria (DMA)
- X c) E/S por flanco
- d) E/S por nivel

Puntuación: -0,33

[T5.1FunE/S]

[E13FebTeo17]

- 30 [P4T]
 Elección En una bomba como las estudiadas en prácticas, del tipo...

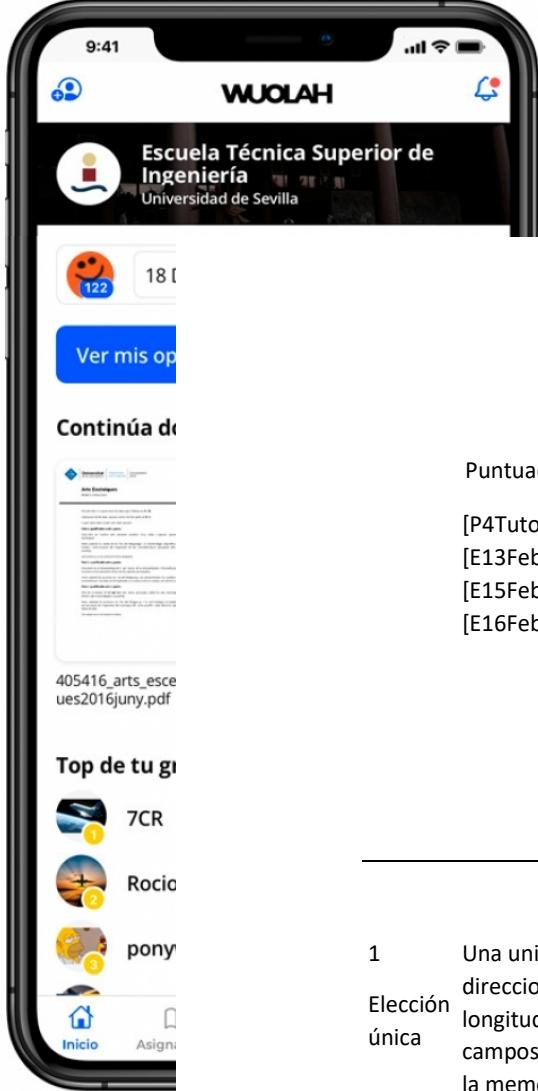
Única

```
0x0804873f <main+207>: call 0x8048504 <scanf>
0x08048744 <main+212>: mov 0x24(%esp),%edx
0x08048748 <main+216>: mov 0x804a044,%eax
0x0804874d <main+221>: cmp %eax,%edx
0x0804874f <main+223>: je 0x8048756 <main+230>
0x08048751 <main+225>: call 0x8048604 <boom>
0x08048756 <main+230>: ...
```

la contraseña es...

Usuario Profesores

- a) el entero 0x804a044
- ✓ • b) el entero almacenado a partir de la posición de memoria
0x804a044
- c) el string almacenado a partir de la posición de memoria
0x24(%esp)



Available on the
App Store

GET IT ON
Google Play

Continúa d

d) ninguna de las anteriores

Puntuación: 1,00

[P4Tutorial]
[E13FebPra11]
[E15FebPra17]
[E16FebPra14]

405416_arts_esce
ues2016juny.pdf

Top de tu gr

7CR
Rocio
pony
Inicio Asign

- 1 Una unidad de control microprogramada con direccionamiento explícito con dos direcciones por microinstrucción, tiene una memoria de control con 35 bits de longitud de palabra. Si las microinstrucciones emplean 15 bits en total para los campos de control y de tipo y condición de salto, el número máximo de palabras de la memoria de control de esta unidad de control microprogramada es de:

Usuario Profesores

- a) 10
- b) 2^{10}
- X c) 2^{20}
- d) 20

Puntuación: -0,33

- 2 Con 8 circuitos de memoria RAM de 1K x 8 se puede crear un memoria de:

Elección Usuario Profesores
única

- a) 1K x 64
- b) 8K x 8
- c) 2K x 32
- ✓ • d) Todas las combinaciones anteriores son posibles

Puntuación: 1,00

- 3 [P4T]
En una bomba como las estudiadas en prácticas, del tipo...
Elección única
0x080486e8 <main+120>: call 0x8048524 <strcmp>

```
0x080486ed <main+125>: test %eax,%eax  
0x080486ef <main+127>: je 0x80486f6 <main+134>  
0x080486f1 <main+129>: call 0x8048604 <boom>  
0x080486f6 <main+134>:
```

...la contraseña es...

Usuario Profesores

- a) el valor que tenga %eax
- b) el string almacenado a partir de 0x80486f6
- X c) el entero almacenado a partir de donde apunta %eax
- d) ninguna de las anteriores

Puntuación: -0,33

[P4Tutorial]

[E14FebPra12]

4 [T2.2.1]

Elección única Si %edx contiene 0xf000 y %ecx contiene 0x0100, el direccionamiento 0x80(%ecx,%edx,2) se refiere a la posición

Usuario Profesores

- a) 0xf182
- b) 0xf280
- c) 0x1e180
- d) Ninguna de las respuestas anteriores es correcta

Puntuación: 0,00

[T2.2.1ModDir]

[E15SepPra03]

5 ¿Cuál es el contenido de una pila al terminar de ejecutarse la siguiente secuencia de operaciones push y pop:

```
push #1  
push #2  
push #3  
pop a  
push #4  
pop a  
pop a
```

Usuario Profesores

- a) 1, 2, 3 y 4
- ✓ • b) 1
- c) 10
- d) 1 y 2

Puntuación: 1,00

6 [T2.4.1]
 Elección ¿Cómo se devuelve en ensamblador x86-64 Linux gcc el valor de retorno de una única función long int al terminar ésta?

Usuario Profesores

- a) La instrucción RET lo almacena en un registro especial de retorno.
- X b) Por convención se guarda en %eax.
- c) Se almacena en pila justo encima de los argumentos de la función.
- d) Ninguna de esas formas es la correcta.

Puntuación: -0,33

[T2.1.4x86-64]
 [T2.2.3CodCon]
 [T2.4.1x86-64]
 [P3Tutorial]
 [E12SepTeo02]
 [E17JulTeo05]

7 [T4.4]
 Elección La predicción de saltos está relacionada con...
 única Usuario Profesores

- a) Los riesgos de transferencia (intenta agrupar las posibles transferencias de un conjunto de instrucciones)
- b) Los riesgos estructurales (intenta evitar el efecto de un fallo de cache)
- c) Los riesgos de (dependencia de) datos (intenta que el dato esté disponible anticipadamente)
- ✓ • d) Los riesgos de control (intenta determinar de antemano el flujo de control)

Puntuación: 1,00

[T4.4Riesgs]
[E15FebTeo13]

8 [T2.1.2]

Elección ¿Cuál de las siguientes afirmaciones sobre la instrucción leave es cierta?
única Usuario Profesores

- a) No es obligatorio usarla. En su lugar puede realizarse una secuencia explícita de operaciones mov y pop

Se ejecuta justo después de retornar de un procedimiento
- b) Lo típico después de retornar (de una función C) es pop/add \$n,%esp y/o mov %eax,...

Equivale a pop %ebp seguida de mov %ebp,%esp
- c) en realidad es mov %ebp,%esp; pop %ebp

Equivale a mov %esp,%ebp seguida de pop %ebp
- d) mov al revés

Puntuación: 0,00

[T2.1.2Lngjes]
[T2.3.1MarcoP]
[E16FebTeo11]

9 [T2.2.3]

Elección En IA32, ¿cuál de los siguientes fragmentos de programa tiene un efecto sobre los flags distinto al resto?
única

Usuario Profesores

- ```
 mov $0, %edi
```

a) 

```
sub $1, %edi
```

  
sub afecta
- ```
        sub %edi, %edi
```

b)

```
adc $0xFFFFFFFF, %edi
```


adc afecta
- c)

```
        mov $-1, %edi
```


mov no afecta a los flags
- ```
 mov $-1, %edi
```

d) 

```
add $0, %edi
```

  
add afecta



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

Continúa d'

Puntuación: 0,00

[T2.2.3CodCon]  
[E17JulTeo03]

10 En el contexto del lenguaje máquina, el acrónimo ISA suele referirse a:

Elección Usuario Profesores  
única

X

a) Intel Standard Architecture

• b) Instruction Set Architecture

c) Information Security Architecture

d) Industry Standard Architecture

405416\_arts\_escue2016juny.pdf

Top de tu gr...

7CR

Rocio

pony

Inicio

Asign.

Puntuación: -0,33

11 [T2.2.1]

Respecto a direccionamiento a memoria en ensamblador IA32 (sintaxis AT&T), de la forma D(Rb, Ri, S), sólo una de las siguientes afirmaciones es FALSA. ¿Cuál?

Elección Usuario Profesores

X

a) El factor de escala S puede ser 1, 2, 4, 8

b) El desplazamiento D puede ser una constante literal (1, 2 ó 4 bytes)

• c) EBP no se puede usar como registro base

X d) ESP no se puede usar como registro índice

Puntuación: -0,33

[T2.2.1ModDir]  
[E12SepTeo07]

12 ¿Cuál de las siguientes afirmaciones es incorrecta?

Elección Usuario Profesores  
única

a) No siempre es necesario indicar la dirección de la siguiente instrucción

b) El formato de una instrucción nos indica el significado de cada bit de la instrucción

c) Todas las instrucciones deben tener código de operación

✓ • d) Todas las instrucciones deben tener operando fuente y operando destino

Puntuación: 1,00

13        El tamaño del registro indicador de una memoria asociativa de  $n$  palabras x  $m$  bits/palabra es:  
Elección única    Usuario Profesores

- X              a)  $n/m$  bits
- b)  $m$  bits
- c)  $n$  bits
- d) un bit

Puntuación: -0,33

14        [T3.1]  
Elección única    ¿Cuál de las siguientes funciones no corresponde a la unidad de control de un procesador?

Usuario Profesores

- ✓              a) Calculo de operaciones de coma flotante
- b) Decodificación de las instrucciones
- c) Secuenciamiento de las instrucciones
- d) Generación de las señales de control que provocan la ejecución de cada instrucción

Puntuación: 1,00

[T3.1CamDat]  
[E14FebTeo13]

15        [T2.4.1]  
Elección única    ¿Cuál de las siguientes listas menciona registros x86-64 del mismo tipo respecto a convenio de uso? (salva-invocante, invocado, etc)

Usuario Profesores

- a) CL, DX, R8d, R9
- b) RAX, RBX, RCX, RDX
- c) RBX, RSI, RDI
- d) RSP, RBP

Puntuación: 0,00

[T2.4.1x86-64]  
[E13SepTeo09]

16 ¿Cuántas líneas de dirección son necesarias en un memoria RAM de 64 K palabras  
dinámica? ¿Y estática?  
Elección única Usuario Profesores

- a) 8 / 8  
b) 16 / 8  
X c) 16 / 16  
• d) 8 / 16

Puntuación: -0,33

17 [P5.1]

Abajo se ofrece el listado de una función para multiplicar matrices  $C = A \times B$ .

Elección única

```
void mult_matr(float A[N][N], float B[N][N], float C[N][N]){
 /* Se asume valor inicial C = {0,0...} */
 int i,j,k;
 for (i=0; i<N; i++)
 for (j=0; j<N; j++)
 for (k=0; k<N; k++)
 C[i][j] += A[i][k] * B[k][j];
}
```

Suponer que:

- El computador tiene una cache de datos de 8 MB, 16-vías, líneas de 64 bytes.
- N es grande, una fila o columna no cabe completa en cache.
- El tamaño de los tipos de datos es como en IA32.
- El compilador optimiza el acceso a  $C[i][j]$  en un registro.

Aproximadamente, ¿qué tasa de fallos se podría esperar de esta función para valores grandes de N?

Usuario Profesores

- a) 1/8  
✓ b) 1/2  
• 1/2 por cada  $B[k][j]$  + 1/32 por cada 16-ésimo  $A[i][k]$   
c) 1/4  
d) 1/16

Puntuación: 1,00

[P5.1Line]  
[E14SepPra09]

[E16FebPra19]

[E17JulPra19]

18 [P5.1]

Elección  
única

Abajo se ofrece el listado de una función para multiplicar matrices  $C = A \times B$ .

```
void mult_matr(float A[N][N], float B[N][N], float C[N][N]){
 /* Se asume valor inicial C = {0,0...} */
 int i,j,k;
 for (i=0; i<N; i++)
 for (j=0; j<N; j++)
 for (k=0; k<N; k++)
 C[i][j] += A[i][k] * B[k][j];
}
```

Suponer que:

- El computador tiene una cache de datos de 8 MB, 16-vías, líneas de 64 bytes.
- N es grande, una fila o columna no cabe completa en cache.
- El tamaño de los tipos de datos es como en IA32.
- El compilador optimiza el acceso a  $C[i][j]$  en un registro.

Imaginar que se modifica la última sentencia (el cuerpo anidado) por esta otra

$C[i][j] += A[i][k] * B[j][k];$

de manera que se calcule  $C = A \times B'$  (A por traspuesta de B). Aproximadamente, ¿qué tasa de fallos se podría esperar de esta nueva función para valores grandes de N?

Usuario Profesores

a) 1/2

- b)  $\frac{1}{16}$   
uno de cada 16  $A[i][k]$  y otro de cada 16  $B[i][k]$
- c) 1/8
- d) 1/4

Puntuación: 0,00

[P5.1Line]

[E14SepPra10]

[E16FebPra20]

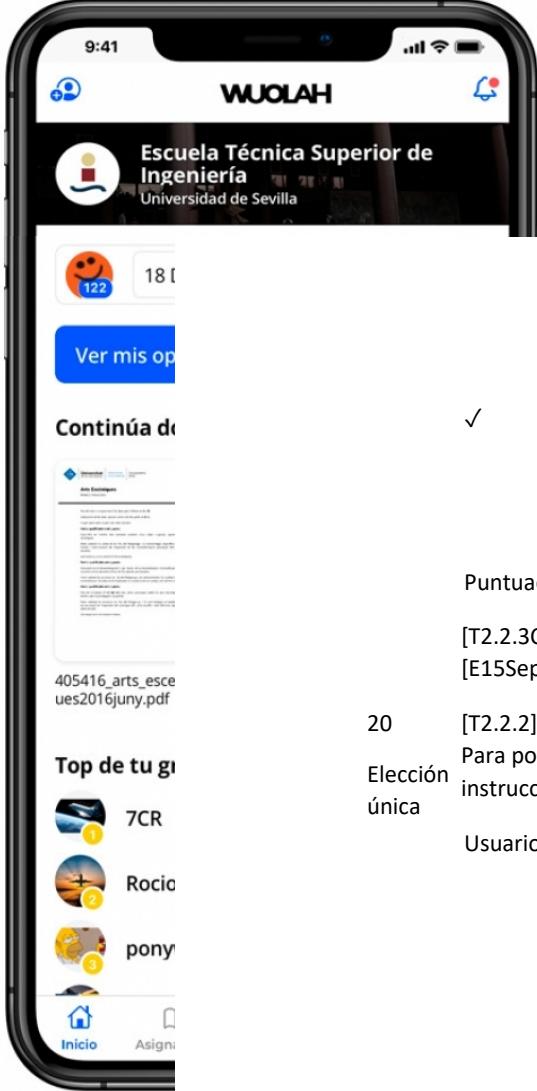
19 [T2.2.3]

Elección  
única

¿Qué combinación de flags aritmético-lógicos corresponde al código de condición b (below)?

Usuario Profesores

a) OF xor SF



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

Continúa d

- ✓ • b) CF

c) OF

d) CF xor OF

Puntuación: 1,00

[T2.2.3CodCon]

[E15SepTeo08]

20 [T2.2.2]

Eleción única Para poner a 1 el bit 5 del registro %edx sin cambiar el resto de bits podemos usar la instrucción máquina:

Usuario Profesores

- a) or \$0x20, %edx
- b) and \$32, %edx
- c) and \$0x5, %edx
- d) or \$0b101, %edx

Puntuación: 0,00

[T2.2.2OpArit]

[E17FebTeo7]

21 En la ejecución de una instrucción...

Eleción única Usuario Profesores

- a) la ALU realiza las operaciones aritméticas y lógicas
- b) la UC activa las señales de control que envía por el bus de direcciones
- c) siempre se altera el registro de estado
- X d) el registro de instrucción se va incrementando para apuntar a la siguiente instrucción

Puntuación: -0,33

22 [T6.1]

Eleción única La tasa de aciertos  $A_{-i}$  del nivel  $i$  de una jerarquía de memoria no depende de:  
(mediante  $_$  se representa subíndice)

Usuario Profesores

- ✓ • a) El ancho de banda  $b_{-i}$  del nivel  $i$ .

Todas las demás se mencionan explícitamente en Tema 6  
tr.24

- b) La capacidad (tamaño)  $s_i$  del nivel  $i$ .
- c) La estrategia de administración de memoria.
- d) La unidad de la transferencia de información  $x_i$  entre el nivel  $i$  y el  $i+1$ .

Puntuación: 1,00

[T6.1ConLoc]

[T6.5MCache]

[E17JulTeo26]

23 ¿Cuál de las siguientes afirmaciones sobre el direccionamiento absoluto es falsa?

Elección Usuario Profesores  
única

- a) El tamaño del operando direccionado queda limitado por el nº de bits del campo dirección de memoria.
- b) La instrucción contiene la dirección de memoria exacta en la que se encuentra el objeto.
- X c) El objeto está en una posición de la memoria principal.
- d) El rango de posiciones direccionables queda limitado por el tamaño del campo de operando.

Puntuación: -0,33

24 [T2.1.3]

Elección ¿Qué valor contendrá el registro edx tras ejecutar las dos instrucciones siguientes?  
única

movl \$-1, %edx  
movb \$1, %dl

Usuario Profesores

- a) 00000001 00000000 00000000 00000000
- b) 11111111 11111111 11111111 11111111
- c) 11111111 11111111 11111111 00000001
- d) 00000000 00000000 00000000 00000001

Puntuación: 0,00

[T2.1.3ConASM]

[E14SepPra15]

- 25 [T4.3]  
Elección única Si un procesador no segmentado necesita 5 ns para leer una instrucción de memoria, 2 ns para decodificar la instrucción, 3 ns para leer del banco de registros, 3 ns para realizar el cálculo requerido por la instrucción, y 2 ns para escribir el resultado en el banco de registros, ¿cuál es la frecuencia máxima del procesador?

Usuario Profesores

- a) 500 MHz
- b) 40 MHz
- c) 66,67 MHz
- d) 200 MHz

Puntuación: 0,00

[T4.3Aceler]

- 26 [T2.2.4]  
Elección única La instrucción JGE / JNL provoca un salto si...  
Usuario Profesores

OF = SF

- a) basta recordar que "Less" no era un flag solo (es OF^SF)  
recordar también que "Below" comprueba CF
- X b) SF = 1  
sería js
- c) CF = 1  
sería jc/jb
- d) SF = 0  
sería jns

Puntuación: -0,33

[T2.2.4SalCon]

[E16FebTeo07]

- 27 En las arquitecturas RISC hay...

Elección única Usuario Profesores

- a) muchos registros y pocos modos de direccionamiento.
- X b) pocas instrucciones muy rápidas con muchos modos de direccionamiento.

- c) pocos modos de direccionamiento y muchos formatos de instrucción.
- d) pocos registros y muchos tipos de instrucciones.

Puntuación: -0,33

28 [T6.5]  
 Elección única ¿En qué se diferencian las estrategias de mantenimiento de coherencia en memoria "escritura directa" y "post-escritura"?

Usuario Profesores

- ✓ • a) En cuándo tiene lugar la actualización
- b) En cómo tiene lugar la actualización
- c) Tanto en a) como en b)
- d) Ni en a) ni en b)

Puntuación: 1,00

[T6.1ConLoc]

[T6.5MCache]

29 El 8086 tiene:

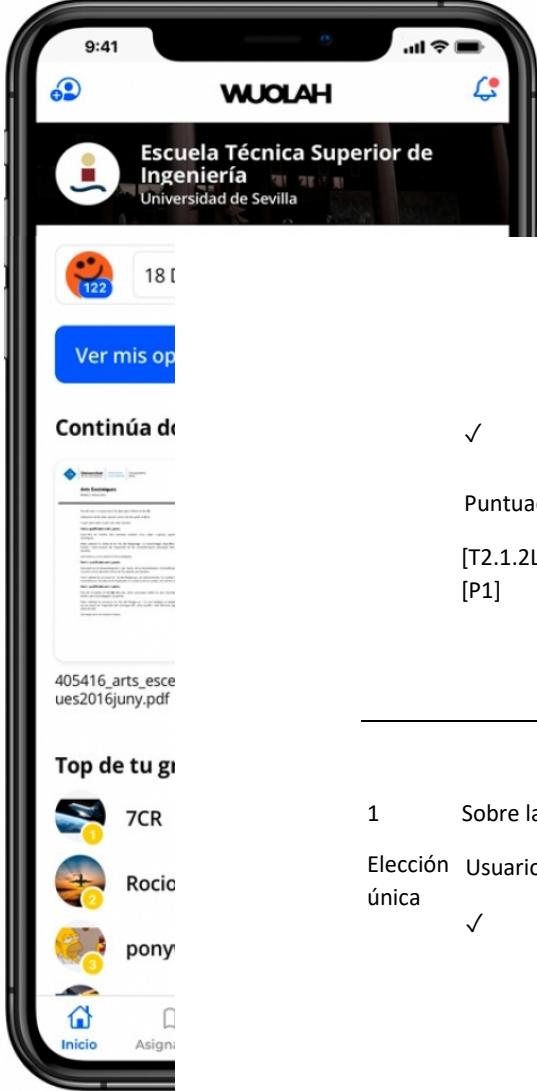
Elección única Usuario Profesores

- a) 14 registros de 16 bits más 4 registros de segmento
- b) 16 registros de 16 bits
- c) 16 registros de 20 bits
- d) 14 registros de 16 bits y un bus de direcciones de 20 bits

Puntuación: 0,00

30 [T2.1.2]  
 Elección única ¿Cuál de las siguientes afirmaciones es incorrecta?  
 Usuario Profesores

- En lenguajes de alto nivel no se tiene acceso a detalles del
- a) procesador como las tablas de páginas, el paso de modo usuario a modo supervisor, el bit de paridad, etc.
  - b) El lenguaje de alto nivel es más portable que el lenguaje máquina.
  - c) En lenguaje ensamblador cada instrucción se corresponde con una instrucción máquina.



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

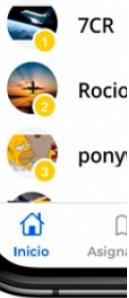
- ✓ • d) En lenguaje ensamblador las instrucciones se escriben en binario.

Puntuación: 1,00

[T2.1.2Lngjes]  
[P1]

405416\_arts\_escuela2016juny.pdf

## Top de tu grupo



- 1 Sobre la E/S mapeada en memoria podemos decir que:

Elección Usuario Profesores única

- ✓ • a) Usa direcciones de memoria para acceder a puertos de E/S  
b) La CPU necesita el pin IO/M#  
c) Dispone de instrucciones especiales de E/S  
d) Todas las respuestas anteriores son falsas

Puntuación: 1,00

- 2 [T2.2.4]

Elección Despues de ejecutar una instrucción de suma sobre dos números con signo de la que sabemos que no provocará overflow (los dos números son pequeños en valor absoluto), queremos comprobar si el resultado de la suma es menor que 0. ¿Qué flag necesita comprobar la instrucción de salto condicional equivalente a... ?

if (resultado<0) then goto label

Usuario Profesores

- a) OF  
b) CF  
c) ZF  
d) SF

Puntuación: 1,00

[T2.2.3CodCon]  
[T2.2.4SalCon]

[E14SepTeo19]

- 3 ¿Qué tipo de localidad de las referencias se define como: "si se referencia un elemento, volverá a ser referenciado pronto"?

Elección Usuario Profesores  
única

- a) Localidad secuencial
- b) Localidad espacial
- c) Localidad iterativa
- ✓      •      d) Localidad temporal

Puntuación: 1,00

4 [T2.2.2]

Elección Si EAX=2, EBX=5 y M[3]=3, ¿qué valores quedan tras ejecutarse la instrucción XOR  
única %BL, 1(%EAX)?

Usuario Profesores

- a) EAX=2 , EBX=5 , M[3]=6
- b) EAX=5 , EBX=6 , M[3]=2
- c) EAX=6 , EBX=2 , M[3]=5
- d) EAX=6 , EBX=5 , M[3]=2

Puntuación: 0,00

[T2.2.1ModDir]

[T2.2.2OpArit]

5 [T2.2.1]

Elección Si EBX=0x00002a00 y EDI=0x0000000a, ¿cuál es la dirección efectiva en la  
única instrucción add 0x64(%ebx,%edi,2),%eax?

Usuario Profesores

- a) 0x00002a78
- b) 0x00002a70
- c) 0x000054dc
- d) 0x00005478

Puntuación: 0,00

[T2.2.1ModDir]

[E15FebPra05]

6 Sobre un sistema que utiliza un esquema de memoria virtual con segmentación  
Elección paginada podemos decir que:

única Usuario Profesores

- a) Es un sistema de correspondencia entre direcciones virtuales y direcciones en memoria caché
- b) Entre disco y memoria principal se transfieren segmentos completos
- c) Las respuestas a y b son ciertas
- d) Las respuestas a y b son falsas

Puntuación: 0,00

7 [T2.2.4]

Elección Al ejecutar el fragmento de código:

única

```
leal -4(%eax), %edx
cmpl $9, %edx
ja .L2
```

se salta a .L2 si el contenido del registro %eax:

Usuario Profesores

- a) es mayor o igual que 48
- X b) está dentro del intervalo [48,57]
- c) está fuera del intervalo [48,57]
  - d) es mayor o igual que 57

Puntuación: -0,33

[T2.2.1ModDir]

[T2.2.2OpArit]

[T2.2.3CodCon]

[T2.2.4SalCon]

[E14SepTeo20]

8 Si d es un desplazamiento, r un registro índice e i una constante apropiada, el modo de direccionamiento indexado con postautoincremento realiza...

Elección

única Usuario Profesores

- a)  $r = r + i$  ; dirección efectiva =  $r + d$
- ✓ • b) dirección efectiva =  $r + d$  ;  $r = r + i$
- c)  $r = r - i$  ; dirección efectiva =  $r + d$
- d) dirección efectiva =  $r + i$  ;  $r = r + d$

Puntuación: 1,00

9 ¿Cuál de los siguientes grupos de instrucciones podrá pertenecer a un procesador con E/S mapeada en memoria?  
Elección única Usuario Profesores

- a) IN, LOAD, OUT
- b) IN, LOAD, MOV
- ✓ • c) LOAD, MOV, STORE
- d) Ninguno de los anteriores

Puntuación: 1,00

10 [T2.3.1]  
Elección única Al llamar a una función de 2 argumentos foo(arg1, arg2), ¿cuál es el orden correcto de las operaciones? (suponiendo convención de llamada x86 cdecl, y que foo requiere ajustar marco de pila, esto es, salvar %ebp)

Usuario Profesores

- a) push arg1, push arg2, push %ebp, call foo
- b) push arg1, push arg2, call foo, push %ebp
- X • c) push arg2, push arg1, push %ebp, call foo
- d) push arg2, push arg1, call foo, push %ebp

Puntuación: -0,33

[T2.3.1MarcoP]  
[E14SepTeo03]  
[E16FebTeo17]

11 [T5.3]  
Elección única ¿Cuál de las siguientes afirmaciones acerca del concepto de interrupción es cierta?  
Usuario Profesores

- a) Su objetivo es incrementar el ancho de banda con el dispositivo
- ✓ • b) Es una bifurcación normalmente externa al programa en ejecución
- c) Permite realizar transferencias sin el control de un programa
- d) Sigue la ejecución del programa

Puntuación: 1,00



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

## Continúa d



## Top de tu grupo



[T5.3ES IRQ]  
[E13FebTeo18]

- 12 [T2.1.4]  
Elección Cuál de las instrucciones máquina siguientes es incorrecta en x86-64:  
única Usuario Profesores

- ✓ • a) movl %r8, %eax  
b) testl %edx, %edx  
c) addq \$1, %rcx  
d) movl (%rdi,%rcx,4), %edx

Puntuación: 1,00

[T2.1.4x86-64]  
[T2.2.0OpArit]  
[T2.2.3CodCon]  
[E17FebTeo5]

- 13 [T2.3.2]  
Elección Es responsabilidad del procedimiento llamado salvaguardar los registros:  
única Usuario Profesores

- ✓ • a) %eax, %edx, %ecx  
b) %ebx, %esi, %edi  
c) %esi, %edi  
d) %eax, %ebx, %ecx, %edx

Puntuación: 1,00

[T2.3.2Conven]  
[E16FebTeo16]  
El enunciado asume convención cdecl

- 14 [T2.1.4]  
Elección Diferencias gcc Linux IA32/x86-64: marcar la respuesta falsa  
única Usuario Profesores

- ✓ • a) los punteros (void\*) pasan de 32 a 64 bits  
b) los enteros largos (long) pasan de 32 a 64 bits  
c) long double pasa de 10/12 B a 16 B  
d) el tipo double pasa de 4 B a 8 B

siempre 8 B

Puntuación: 1,00

[T2.1.4x86-64]

[E16SepTeo17]

15 [T4.3]

Elección En un procesador con segmentación de cauce, aumentar el número de etapas (p.ej. de 2 a 4, o de 4 a 8), tiene en general como consecuencia:

única Usuario Profesores

a) Un mayor retraso en la ejecución de los programas debido al incremento del número de etapas

✓ • b) Un incremento de las prestaciones

c) Una disminución de la máxima frecuencia de reloj a la que puede operar el cauce

d) Una disminución en la posible dependencia de datos

Puntuación: 1,00

[T4.3Aceler]

[E13FebTeo30]

16 [T5.1]

Elección ¿En qué tipo de transferencias es necesario establecer un periodo de tiempo máximo después del cual se considera que ha fallado?

única Usuario Profesores

a) En las transferencias síncronas

• b) En las transferencias asíncronas

X c) En ambas

d) En ninguna

Puntuación: -0,33

[T5.1FunE/S]

[E12FebTeo25]

17 ¿Cuál de los siguientes elementos no forma parte del canal de un controlador de acceso directo a memoria?

Elección Usuario Profesores

• a) Registro de prioridades.

b) Registro contador.

X c) Todos los elementos de las otras respuestas forman parte de un canal de DMA.

d) Registro de dirección.

Puntuación: -0,33

- 18 Un computador usa el formato vertical de codificación de instrucciones para parte de las señales de control y el formato horizontal para k señales de control. El formato vertical posee n campos codificados de m bits cada uno. ¿Cuál es el máximo número de señales de control que pueden usarse en este computador?

Elección única Usuario Profesores

a)  $k + n \cdot 2^m$

X b)  $k + n^m$

• c)  $k + n \cdot (2^m - 1)$

d) Ninguno de los anteriores

Puntuación: -0,33

- 19 El tamaño del registro indicador de una memoria asociativa de n palabras x m bits/palabra es:

Elección única Usuario Profesores

• a) n bits

b) m bits

X c)  $n/m$  bits

d) un bit

Puntuación: -0,33

- 20 ¿En qué técnica para determinar la dirección de comienzo de la rutina de servicio de interrupción se fija dicha dirección en los circuitos de la CPU?

Elección única Usuario Profesores

a) Direccionamiento absoluto.

• b) Direcciones fijas.

c) Envío de instrucción de bifurcación completa.

X d) Direccionamiento relativo.

Puntuación: -0,33

- 21 La convención de llamada Linux/GCC IA32 considera, respecto a convenios de uso de registros:
- Elección única Usuario Profesores
- a) 8 registros salva-invocante, 6 registros salva-invocado, y 2 especiales
  - b) 3 registros salva-invocante, 3 registros salva-invocado, y 2 especiales
  - X c) Algunos registros para pasar argumentos, otros salva-invocante, otros salva-invocado, dos especiales
  - ✓ d) Algunos registros salva-invocante, otros salva-invocado, uno especial

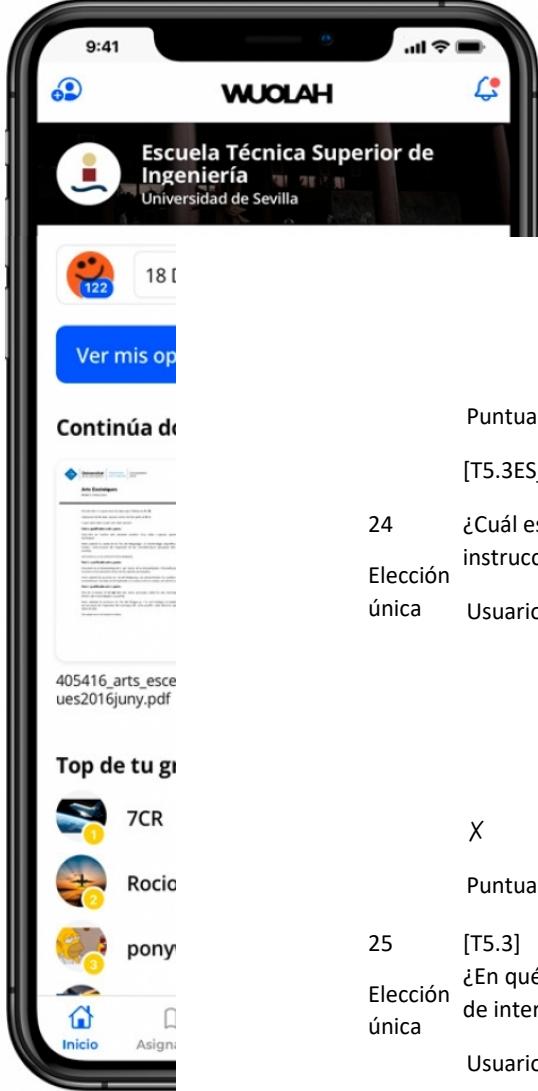
Puntuación: -0,33

- 22 [T6.1]
- Elección única Si el tiempo de acceso a la memoria cache es de 2 ns y el tiempo necesario para tratar un fallo de cache es de 80 ns, ¿cuál es la tasa de aciertos necesaria para que el tiempo medio de acceso al sistema de memoria sea de 10 ns?
- Usuario Profesores
- a) 0.8
  - b) 0.75
  - c) 0.95
  - ✓ d) 0.9

Puntuación: 1,00

[T6.1ConLoc]  
[T6.5MCache]  
[E14FebTeo16]

- 23 [T5.3]
- Elección única La técnica de sondeo, escrutinio o "polling"...
- Usuario Profesores
- a) Se utiliza para identificar el destino de una interrupción
  - b) Permite establecer un mecanismo de asignación de prioridades a los distintos dispositivos
  - X c) En caso de utilizarse, es necesario emplear varias líneas para que los dispositivos soliciten una interrupción
  - d) Todas las respuestas anteriores son falsas



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

## Continúa d

Puntuación: -0,33

[T5.3ES IRQ]

- 24 ¿Cuál es el contenido de la pila al terminar de ejecutarse la siguiente secuencia de instrucciones de una arquitectura de pila: push #4; push #7; add; push #10; sub?  
Elección única Usuario Profesores

a) 4, 11, 1

- b) 1

c) 11, 1

d) 4, 7, 10

405416\_arts\_esce ues2016juny.pdf

## Top de tu grupo

7CR

Rocio

pony

Inicio

Asign

Puntuación: -0,33

- 25 [T5.3]  
¿En qué método para determinar la dirección de comienzo de una rutina de servicio de interrupción se envía parte de dicha dirección?  
Elección única Usuario Profesores

a) Direccionamiento absoluto

b) Direccionamiento mediante instrucción de bifurcación

- c) Direccionamiento relativo

d) Direccionamiento indirecto

Puntuación: -0,33

[T5.3ES IRQ]

- 26 [T6.2]  
La memoria DRAM:  
Elección única Usuario Profesores

X

a) Es menos densa que la memoria SRAM

Se denomina dinámica porque para mantener almacenado un

- b) dato hay que recargarlo cada cierto tiempo en un ciclo de refresco

c) Necesita 6 transistores por cada celda

d) Se inventó en la década de los 90

Puntuación: -0,33

[T6.2RAMROM]  
[E13SepTeo29]

27 La conexión de las salidas de tres registros hacia un bus común en el camino de datos puede realizarse usando...  
Elección única Usuario Profesores

- a) tres conexiones directas al bus común
- b) tres demultiplexores
- ✓ • c) dos multiplexores de 2 a 1
- d) dos buffers triestado

Puntuación: 1,00

28 [T2.2.3]  
Un overflow nunca puede ocurrir cuando:  
Elección única Usuario Profesores

- a) se suman dos números negativos  
0x80000000 + 0xffffffff
- b) se suman dos números positivos  
0xfffffff + 0x1
- c) se resta un número positivo de un número negativo  
0x80000000 - 0x00000001
- ✓ • d) se suma un número positivo a un número negativo  
resultado queda entre ambos => se puede representar

Puntuación: 1,00

[T2.2.3CodCon]  
[E16FebTeo08]

29 [T2.4.1]  
Comparando las convenciones de llamada de gcc Linux IA32 con x86-64 respecto a registros  
Elección única Usuario Profesores

- En IA32 %ecx es salva-invocante, y en x86-64 %rcx es salva-invocante también
- a) %rcx es 4º argumento - salva-invocante, ok

- En IA32 %ebx es salva-invocante, pero en x86-64 %rbx es
- b) salva-invocado  
 %ebx es salva-invocado (verde en tr.2.3.36)
- En IA32 %ebp es especial (marco de pila), y en x86-64 %rbp
- c) también  
 %rbp es salva-invocado (verde en tr.2.4.5)
- En IA32 %esi es salva-invocado, y en x86-64 %rsi es salva-
- d) invocado también  
 %rsi es 2º argumento - salva-invocante (amarillo en tr.2.4.5)

Puntuación: 0,00

[T2.4.1x86-64]  
 [E15SepPra11]  
 [E17JulTeo06]

- 30      [T6.5]
- Elección    En las políticas anticipativas de extracción de cache, ¿cuál de ellas se caracteriza por preextraer el bloque  $i+1$  si se referencia al bloque  $i$  y se produce falta de bloque?
- única
- Usuario Profesores

- a) Preextracción indexada
  - b) Preextracción marcada
  - c) Preextracción por falta
  - X       d) Preextracción siempre
- 

- 1       Respecto a las unidades de control nanoprogramadas:

- Elección    Usuario Profesores
- única
- X
- a) La realización nanoprogramada de una unidad de control es más rápida que la micropogramada.
  - b) El diseño de las unidades de control nanoprogramadas debe ser vertical.
  - c) Suponiendo una memoria de microprograma con  $n$  microinstrucciones de  $w$  bits cada una, de las cuales  $2^m$  son distintas, el ahorro en bits si se utiliza nanoprogramación es  $(n \cdot m + 2^m \cdot w) - n \cdot w$ .

- La anchura de la memoria de nanoprograma es la misma que d) la de memoria de microprograma en un diseño de la misma unidad de control que no usara nanoprogramación.

Puntuación: -0,33

2 ¿Cuántos bytes puede transmitir como máximo el controlador de acceso directo a memoria 8237 de forma consecutiva?  
 Elección única Usuario Profesores

- a) 128 KB
- b) Existe al menos un modo de funcionamiento sin límite máximo
- c) 32 KB
- X d) 64 KB

Puntuación: -0,33

3 [T2.2.3]  
 Elección única ¿Cuál de las siguientes parejas de mnemotécnicos de ensamblador IA32 corresponden a la misma instrucción máquina?  
 Usuario Profesores

- ✓ • a) JZ (saltar si cero), JE (saltar si igual)
- b) SAR (desplazamiento aritmético a la derecha) / SHR (desplazamiento lógico a la derecha)
- c) CMP (comparar), SUB (restar)
- d) JC (saltar si acarreo), JL (saltar si menor, para números con signo)

Puntuación: 1,00

[T2.2.2OpArit]  
 [T2.2.3CodCon]  
 [T2.2.4SalCon]  
 [E12FebTeo15]

4 [T3.3CtrlUp]  
 Elección única En una unidad de control microprogramada con formato de microinstrucciones vertical, un subcampo que deba especificar 16 señales de control codificadas de tal forma que pueda activarse sólo una o ninguna habrá de tener una anchura mínima de

Usuario Profesores



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

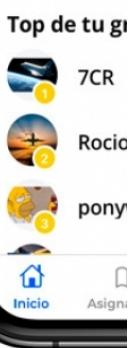
## Continúa d



- X                    a) 16 bits  
                      b) 17 bits  
                      c) 4 bits  
                      •      d) 5 bits

Puntuación: -0,33

[T3.3CtrlUp]



- 5                    [P3.1]  
Elección            ¿Cuál es el popcount (peso Hamming, nº de bits activados) del número 19?  
única              Usuario Profesores

- ✓                    a) 2  
                      •      3  
                      b)  $19 = 0x13 = 16+3 = 0b0001\ 0011 \rightarrow \text{popcount } 3$   
                      c) 4  
                      d) 5

Puntuación: 1,00

[P3.1Pcount]  
[T2.2.5BWhile]  
[E17FebPra10]

- 6                    [T5.4]  
Elección            Si el acceso directo a memoria se realiza mediante robo de ciclo:  
única              Usuario Profesores

- ✓                    a) es posible que la ejecución de una operación elemental de transferencia en el bus sea temporalmente detenida  
                      •      b) es posible que la ejecución de una instrucción máquina sea temporalmente detenida  
                      c) es necesario que termine de ejecutarse la instrucción máquina actual para comenzar una transferencia por DMA  
                      d) ninguna de las anteriores es cierta

Puntuación: 1,00

[T5.4ES\_DMA]

- 7                    Una memoria SRAM tiene una capacidad de 64 Kbits y precisa 12 líneas de dirección para su manejo. Indique cuál es el tamaño de palabra de dicha memoria:

Elección Usuario Profesores  
única

- ✓ • a) 16 bits  
b) 8 bits  
c) 64 bits  
d) 32 bits

Puntuación: 1,00

8 [T6.2]

Elección ¿Cuál de las siguientes afirmaciones es falsa?  
única Usuario Profesores

- a) La lectura de un bit de la matriz de almacenamiento de una memoria DRAM proporciona una señal mucho más débil que la suministrada por los inversores de una celda de memoria SRAM.
- b) Las memorias DRAM son en general mucho más rápidas que las SRAM
- c) Las memorias DRAM presentan generalmente una capacidad de almacenamiento mucho mayor que las SRAM.
- d) Una celda DRAM sólo necesita un transistor y un condensador.

Puntuación: 1,00

[T6.2RAMROM]

9 [T2.1.2]

Elección En Linux IA32, si gcc usa la instrucción leave se puede asegurar que en ese punto del  
única programa

Usuario Profesores

ya no hay variables locales que destruir

- a) puede usarse leave para destruir rápidamente todas las variables locales, siempre que no haya que recuperar registros salva-invocado
- ya no se hacen llamadas anidadas y por tanto no hay parámetros que ocupen espacio en pila
- b) puede usarse leave habiendo espacio reservado en pila para argumentos de llamadas anidadas, siempre que no haya que recuperar registros salva-invocado

- X                    correspondería emitir la secuencia de salida pop/ret, pero
- c) leave hace lo mismo y ocupa menos espacio
  - leave no equivale a ret, y sólo hace pop %ebp
  - d) ya no hay registros salva-invocado que recuperar
  - tal vez porque no había ninguno, para empezar

Puntuación: -0,33

[T2.1.2Lngjes]

[T2.3.1MarcoP]

[E16SepTeo13]

- 10                  [T2.2.1]  
 Elección            ¿Cuál de las siguientes instrucciones convierte `%eax = 5 * %eax`?  
 única                1) `mov (%eax, %eax, 4), %eax`  
                        2) `lea (%eax, %eax, 4), %eax`

Usuario Profesores

- a) Sólo 2
- b) Ambas 1 y 2
- c) Ninguna

X                    d) Sólo 1

Puntuación: -0,33

[T2.1.3ConASM]

[T2.2.1ModDir]

[E14FebTeo02]

- 11                  Elección            Los valores de los registros Argumento y Máscara de una memoria asociativa son los  
                       única                siguientes:  
                       Argumento: 010010  
                       Máscara: 101011  
                       Si en la primera posición de la memoria está almacenado el valor 0, y las siguientes  
                       celdas de memoria tienen el valor de la celda inmediatamente anterior  
                       incrementado en 1, siendo el valor de la última celda el 7, ¿cuál sería el valor del  
                       registro indicador o de marca?

Usuario Profesores

- a) 00100010
- b) 01001011
- c) 10101100
- d) Ninguno de los anteriores

Puntuación: 0,00

12 Con 8 circuitos de memoria RAM de 1K x 8 se puede crear un memoria de:

Elección Usuario Profesores  
única

a) 1K x 64

b) 8K x 8

c) 2K x 32

✓ • d) Todas las combinaciones anteriores son posibles

Puntuación: 1,00

13 ¿Cuál de las siguientes afirmaciones sobre los lenguajes ensambladores es falsa?

Elección Usuario Profesores  
única

✓ • a) A cada sentencia le corresponde un conjunto preestablecido de instrucciones máquina.

b) A cada sentencia le corresponde una única instrucción de lenguaje máquina.

c) Usan símbolos nemotécnicos para los códigos de operación de las instrucciones.

d) Usan etiquetas para las direcciones de memoria.

Puntuación: 1,00

14 [T2.3.1]

Elección Si el contenido de ESP es 0xAC00 antes de ejecutar la instrucción push %ebx. ¿Cuál será su contenido tras ejecutarla?

Única Usuario Profesores

a) 0xAC02

b) 0xAC04

✓ • c) 0xABFC

d) 0xABFE

Puntuación: 1,00

[T2.3.1MarcoP]

[E15FebPra08]

15 [T2.1.4]

En un sistema IA32 Linux, ¿cuál es el tamaño de un long?



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

Elección única Usuario Profesores

- a) 6 bytes
- X b) 8 bytes
- c) 2 bytes
- d) 4 bytes

Puntuación: -0,33

[T2.1.4x86-64]  
[E14FebTeo06]

16 [T2.2.1]

En el direccionamiento indirecto a través de registro, la dirección efectiva...

Elección única Usuario Profesores

- a) se encuentra en un registro general del procesador.
- b) se encuentra en el registro de instrucción.
- c) se encuentra en una dirección de memoria.
- d) se calcula como la suma del contenido de dos registros.

Puntuación: -0,33

[T1.2ConBas]  
[T2.2.1ModDir]

17 [T5.1]

Alguna de las siguientes NO es una técnica de E/S de las estudiadas en clase:  
Elección única Usuario Profesores

- a) E/S controlada por interrupciones
- b) E/S mediante Acceso Directo a Memoria
- c) E/S programada
- ✓ • d) E/S asíncrona

Puntuación: 1,00

[T5.1FunE/S]  
[E14SepTeo10]

18 [T2.1.4]

La arquitectura x86-64 tiene:  
Elección única Usuario Profesores

WUOLAH

- a) 32 registros RPG de 64 bits
  - b) 64 registros RPG de 64 bits
  - c) 8 registros de propósito general (RPG) de 64 bits (%rax, %rbx, ... %rsp, %rbp)
  - ✓ d) 16 registros RPG de 64 bits

Puntuación: 1,00

[T2.1.4x86-64]

[E13SepTeo08]

19 ¿En qué tipo de política de arbitraje de buses se desperdicia ancho de banda cuando alguno de los posibles maestros no tiene nada que transmitir?  
Elección única Usuario Profesores

- - a) Política dinámica FIFO
    - b) Política estática
    - c) Política dinámica de prioridad fija
    - d) Ninguna de las respuestas anteriores es cierta

Puntuación: 0,00

20 [T3.1] ¿Cuál de las siguientes funciones no corresponde a la unidad de control de un procesador?

Elección única

Usuario Profesores

- ✓     • a) Generación de las señales de control que provean la ejecución de cada instrucción

      b) Calculo de operaciones de coma flotante

      c) Decodificación de las instrucciones

      d) Secuenciamiento de las instrucciones

Puntuación: 1.00

[T3.1CamDat]

[F14FebTeo13]

21 [T2.1.2] Un programa de ordenador que convierte un programa fuente de alto nivel completo en lenguaje máquina se llama un:

Elección única

Usuario Profesores

- ✓ • a) compilador  
b) ensamblador  
c) intérprete  
d) simulador

Puntuación: 1,00

[T2.1.2Lngjes]

[E14SepTeo14]

22 Respecto a registros salva-invocante y salva-invocado en GCC/Linux IA32, ¿cuál de éstos es de distinto tipo que el resto?

Elección única Usuario Profesores

- a) esi  
b) edi  
✓ • c) eax  
d) ebx

Puntuación: 1,00

23 [T2.1.2]

Elección única ¿Cómo se devuelve en ensamblador x86 Linux gcc el valor de retorno de una función al terminar ésta?

Usuario Profesores

- a) Se almacena en pila justo encima del (%ebp) del invocado  
b) Se almacena en pila justo encima de los argumentos de la función  
c) La instrucción RET lo almacena en un registro especial de retorno  
✓ • d) Por convención se guarda en %eax

Puntuación: 1,00

[T2.1.2Lngjes]

[P3Tutorial]

[E12FebTeo02]

24 ¿Cuál de las siguientes afirmaciones acerca de un sistema con un bus único es falsa?

Elección única Usuario Profesores

- X            a) El procesador y los dispositivos pueden funcionar a diferentes velocidades si funciona de forma asíncrona
- b) El procesador y los dispositivos pueden funcionar a diferentes velocidades si funciona de forma síncrona
- c) Sólo un dispositivo puede escribir en el bus en un instante dado
- d) Se utilizan las mismas líneas de control para conectar todos los dispositivos

Puntuación: -0,33

25        ¿Cuál de las siguientes tareas no es responsabilidad de un circuito de interfaz o controlador de periféricos sencillo?

Elección única    Usuario Profesores

- ✓            a) Recibir señales de control desde el procesador
- b) Ejecutar el programa de transferencia de información entre el procesador y los dispositivos de E/S
- c) Adaptar el formato de las señales
- d) Ajustar la temporización entre el procesador y los dispositivos de E/S

Puntuación: 1,00

26        [P3T]

Elección única    Usuario Profesores

- a) ecx
- b) edx
- ✓            c) edi
- d) esi

Puntuación: 1,00

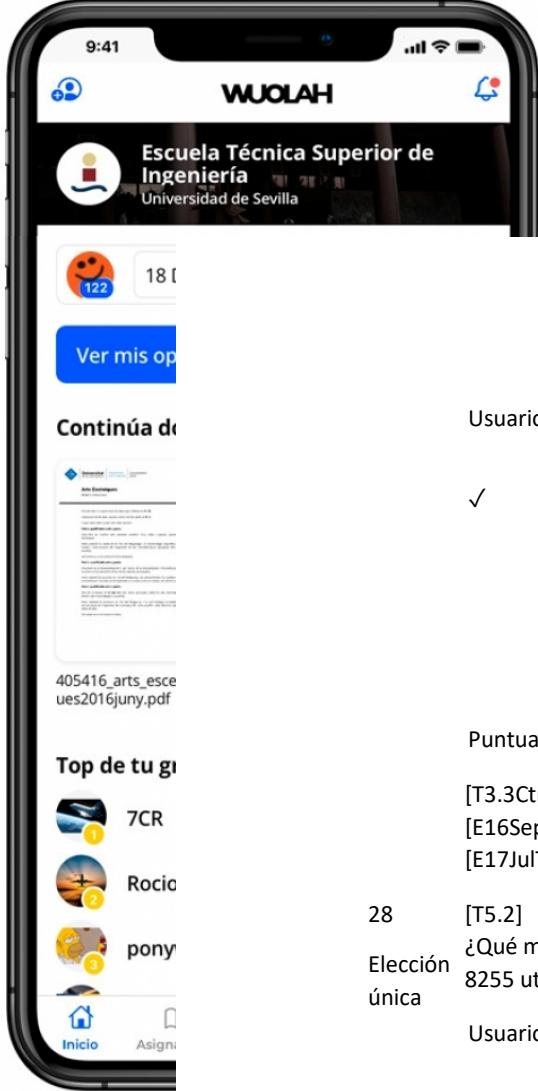
[P3Tutorial]

[T2.4.1x86-64]

[E17JulPra12]

27        [T3.3]

Elección única    Dado un camino de datos concreto, un posible formato de micropogramación se caracteriza como horizontal o vertical según tenga más o menos (señalar la respuesta falsa)



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

## Continúa d

### Usuario Profesores

- ✓ • microbifurcaciones  
a) T3 tr.49

- b) longitud relativa de microinstrucción  
c) codificación  
d) solapamiento

Puntuación: 1,00

[T3.3CtrlUp]  
[E16SepTeo23]  
[E17JulTeo11]

28 [T5.2]

Elección única ¿Qué modo de funcionamiento permite a la interfaz de periféricos programable 8255 utilizar un bus bidireccional?

### Usuario Profesores

- a) 2  
b) 3  
c) 0  
d) 1

Puntuación: 0,00

[T5.2ESProg]

29 Indique cuál de las siguientes características no es cierta en el direccionamiento indirecto a memoria a través de memoria:

### Elección única Usuario Profesores

Permite una gran capacidad de direccionamiento al poderse  
a) utilizar todos los bits de la palabra de memoria como dirección.

- ✓ • b) La instrucción contiene la dirección de memoria exacta en que se encuentra el objeto.  
c) No se requieren cálculos previos para conocer la dirección final.  
d) Son necesarios dos accesos a memoria (aparte de la fase de búsqueda de instrucción) para acceder al objeto.

Puntuación: 1,00

30 El 8086 tiene:

Elección Usuario Profesores  
única

- a) 14 registros de 16 bits y un bus de direcciones de 20 bits
- b) 14 registros de 16 bits más 4 registros de segmento
- c) 16 registros de 16 bits
- d) 16 registros de 20 bits

Puntuación: -0,33

---

**1** [P2T]

¿Cuál de los siguientes es el orden correcto en el ciclo de compilación de un programa en lenguaje C? (el fichero sin extensión es un ejecutable):  
Elección única

Usuario Profesores

- a) fich.c → fich.s → fich → fich.o
- b) fich → fich.s → fich.o → fich.c
- c) fich.c → fich.o → fich.s → fich
- d) fich.c → fich.s → fich.o → fich

Puntuación: **1,00**

[P2Tutorial]

[T2.1.2Lngjes]

[E13FebPra01]

**2** [P2A2]

Alguno de los siguientes no es un nombre de registro en una máquina x86-64 en modo 64 bits  
Elección única

Usuario Profesores

sih

- a) %sil en todo caso, no %sih  
Ver libro Hallaron Figura 3.35
- b) r12w
- c) r8d
- d) spl

Puntuación: **1,00**

[P2Apendice2]  
[T2.4.1x86-64]  
[E16SepPra11]

### 3 [P1]

Sobre el programa ensamblador:

Elección Usuario Profesores  
única

X

- La calidad de un programa ensamblador afectará
- a) menos al tiempo de ejecución de los programas generados por él que la calidad de un compilador.
  - Las etiquetas permiten que el programador especifique el destino de un salto de forma que éste no tenga que modificarse manualmente cuando el programa varíe de tamaño.
  - b) El lenguaje ensamblador elimina la posibilidad de errores en la generación de la representación en lenguaje máquina de cada instrucción.
  - c) d) Todas las respuestas son ciertas.

Puntuación: -0,33

[P1]

[P2Tutorial]

[T2.1.2Lngjes]

### 4 [P4T]

En la práctica de la bomba, el segundo ejercicio consistía en crear un ejecutable sin “explosiones”, para lo cual se puede utilizar... (marcar la opción \*falsa\*)

Elección  
única

Usuario Profesores

✓

•

- a) gdb
- b) ddd
- c) hexedit
- d) objdump

Puntuación: 1,00

[P4Tutorial]

[E17JulPra16]

La redacción original era: ...para lo cual se puede utilizar...

- a. objdump o gdb
- b. gdb o ddd
- c. ddd o hexedit
- d. hexedit u objdump

Con esa redacción, hubo que dar por válidas b. y c. Con depuradores se usaría el método explicado en P4T ejercicio 2, y con hexedit el método explicado en ejercicio 3. Es cierto que hexedit a secas (sin objdump ni depurador) no permite saber dónde hay que modificar nada, pero todo el mundo dice (correctamente) que con hexedit se

puede crear el ejecutable sin explosiones, así que esta redacción es menos problemática.

## 5 [P3.2]

La práctica "parity" debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la Elección siguiente versión de parity5:

única

```
int parity5(unsigned * array,
 int len){
 int i,j, result = 0;
 unsigned x;
 for(i = 0; i<len; i++){
 x=array[i];
 for(j=1; j<8*sizeof(int); j*=2)
 x ^= x >> j;
 result += x & 0x1;
 }
 return result;
}
```

Esta función sólo se diferencia de la versión "oficial" recomendada en clase, en las condiciones del bucle for interno.

Esta función parity5:

Usuario Profesores

Produce siempre el resultado correcto

- a) Efectivamente, con este for también se cumple que el LSB (bit menos significativo) sigue siendo el XOR lateral de todos los bits.  
b) Fallaría con array={0,1,2,3}  
c) Fallaría con array={1,2,4,8}  
d) No es correcta pero el error no se manifiesta en los ejemplos propuestos, o se manifiesta en ambos

Puntuación: 0,00

[P3.2Parity]

[E17FebPra14]

El bucle sugerido por el guión de prácticas hubiera sido:

for (j=8\*sizeof(unsigned); j>0; j>>=1)

## 6 [P2T]

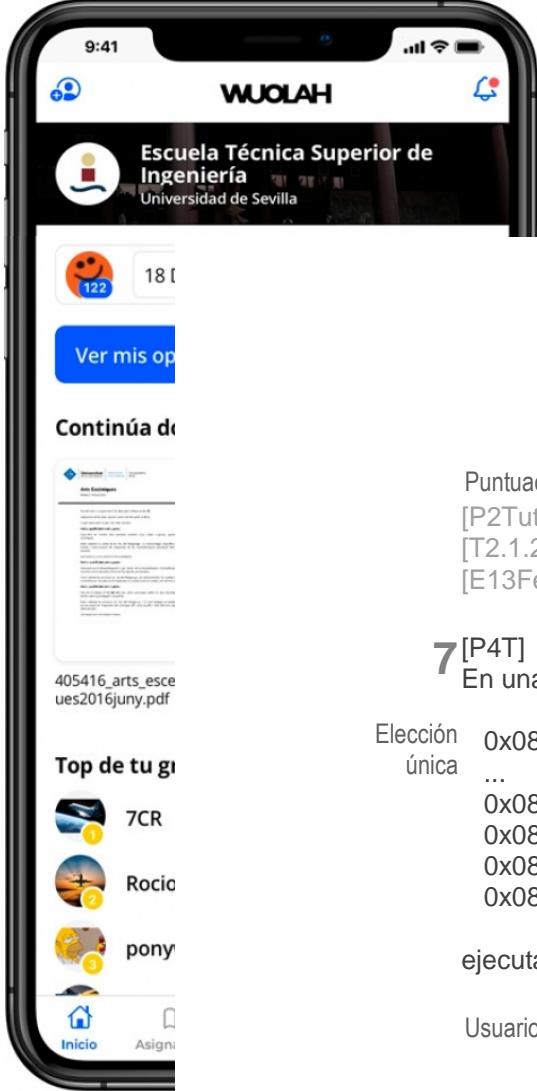
¿Qué hace gcc -O?

Elección  
única

Usuario Profesores



- a) Compilar con optimización suave  
b) Compilar .c→.o (fuente C a objeto)  
c) Compilar .s→.o (fuente ASM a objeto)



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

- d) Ambas (b) y (c), según la extensión de los ficheros que se usen como argumentos

Puntuación: 1,00

[P2Tutorial]  
[T2.1.2Lngjes]  
[E13FebPra02]

7 [P4T]

En una bomba como las estudiadas en prácticas, del tipo...

Elección única 0x08048705 <main+149>: call 0x80484c4 <gettmeofday>  
...

0x08048718 <main+168>: cmp \$0x5,%eax  
0x0804871b <main+171>: jle 0x8048722 <main+178>  
0x0804871d <main+173>: call 0x8048604 <boom>  
0x08048722 <main+178>: ...

ejecutada paso a paso con el depurador ddd, interesaría...

Usuario Profesores

- a) ejecutar hasta jle, ajustar %eax a 6, y continuar ejecutando paso a paso
- b) ejecutar hasta jle, ajustar %eax a 4, y continuar ejecutando paso a paso
- c) cambiar jle por jmp usando ddd o un editor hex,
- d) Ninguna de las opciones anteriores es de interés
- (bien porque no se pueda hacer eso o porque no sirva para evitar la bomba)

Puntuación: 1,00

[P4Tutorial]  
[E13FebPra12]

8 [P5.1]

En la práctica de la cache, el código de line.cc incluye la sentencia

Elección única for (unsigned long long line=1;  
line<=LINE; line<<=1) { ... }

¿Qué objetivo tiene la expresión line<<=1?

Usuario Profesores

- a) volver al principio del vector cuando el índice excede la longitud del vector
- b) duplicar el tamaño del salto en los accesos al vector respecto a la iteración anterior

- c) salir del bucle si el tamaño de línea se volviera menor o igual que 1 para algún elemento del vector
- d) sacar un uno (1) por el stream line

Puntuación: **1,00**

[P5.1Line]

[E15FebPra18]

### **9** [P2.1]

En la práctica "media" se pide sumar una lista de 32 enteros SIN signo de 32bits en una plataforma de 32bits sin perder precisión, esto es, evitando perder acarreos. ¿Cuál es el mínimo valor entero que único repetido en toda la lista causaría acarreo con 32bits (sin signo)?

Usuario Profesores

- a) 0x0800 0000
- b) 0xfbff ffff
- c) 0xfc00 0000
- d) 0x07ff ffff

Puntuación: **1,00**

[P2.1SumUns]

[E14SepPra03]

[E15FebPra14]

[E16FebPra06]

### **10** [P3.2]

La práctica "parity" debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. La siguiente función contiene un único error (en realidad se han editado 2 líneas de código sobre la versión correcta) pero produce resultado correcto cuando se usa como test el array...

```
int parity4(unsigned* array, int len){
 int i;
 unsigned x;
 int val=0, result=0;

 for (i=0; i<len; i++){
 x = array[i];
 asm("\n"
 "ini: \n\t"
 "xor %[x], %[v] \n\t"
 "shr %[x] \n\t"
 "jnz ini \n\t"
 : [v]"+r" (val)
 : [x] "r" (x)
);
 result += val & 0x1;
 }
}
```

```
 return result;
}
```

Usuario Profesores

- array={1, 16, 256, 1024}
- a) debería salir 4 (todos impares)  
sale 2 (1,0,1,0, cada otro impar deshace el anterior)
- array={1, 2, 4, 8}
- b) debería salir 4 (1,1,1,1, todos impares)  
sale 2 (1,0,1,0, cada otro impar deshace el anterior)
- array={0, 1, 2, 3}
- c) debería salir 2 (0,1,1,0, impares el 1 y el 2)  
sale 1 (0,1,0,0, el 2 deshace el impar del 1)
- array={5, 4, 3, 2}
- d) debería salir 2 (impares el 4 y el 2)  
sale 2 (0,1,1,0, el 3 cuenta como impar, el 2  
deshace el impar)

Puntuación: **0,00**

[P3.2Parity]

[E15SepPra17]

El error consiste en que val=0 se inicializa al principio, en lugar de tras cada x=array[i], de manera que si el LSB de val se queda activado la siguiente paridad se calcula mal. En concreto, tras calcular un impar, todos los pares que sigan cuentan como impar hasta que llegue un impar que deshaga el impar (y entonces el nuevo impar cuenta como par).

## 11 [P4T]

Respecto a las bombas estudiadas en la práctica "bomba digital", ¿en cuál de los siguientes tipos de bomba sería más difícil descubrir la(s) Elección contraseña(s)? Se distingue entre enteros definidos en el código fuente única de la bomba, y enteros solicitados al usuario mediante scanf(). Por "procesar" podemos entender calcular el n-ésimo elemento de la serie de Fibonacci, por ejemplo.

Usuario Profesores

- ✓ • a) 1 entero del fuente se procesa, y se compara con el entero del usuario
- b) 2 enteros del usuario se suman, se procesa la suma, y se compara con el entero del fuente
- c) 2 enteros del usuario se procesan, se suman los resultados, y se compara con el entero del fuente  
Las 2 (o 3) opciones más difíciles son de la misma dificultad, así que no se puede marcar ninguna como la más difícil

Puntuación: **1,00**

## 12 [P4T]

En una bomba como las estudiadas en prácticas, del tipo...

Elección única

```
0x0804873f <main+207>: call 0x8048504 <scanf>
0x08048744 <main+212>: mov 0x24(%esp),%edx
0x08048748 <main+216>: mov 0x804a044,%eax
0x0804874d <main+221>: cmp %eax,%edx
0x0804874f <main+223>: je 0x8048756 <main+230>
0x08048751 <main+225>: call 0x8048604 <boom>
0x08048756 <main+230>:
```

...el código numérico (pin) es...

Usuario Profesores

- a) el entero almacenado a partir de la posición de memoria 0x24(%esp)
- b) el entero cuya dirección está almacenada en la posición de memoria 0x804a044
- c) el entero 0x804a044
- d) el entero almacenado a partir de la posición de memoria 0x804a044

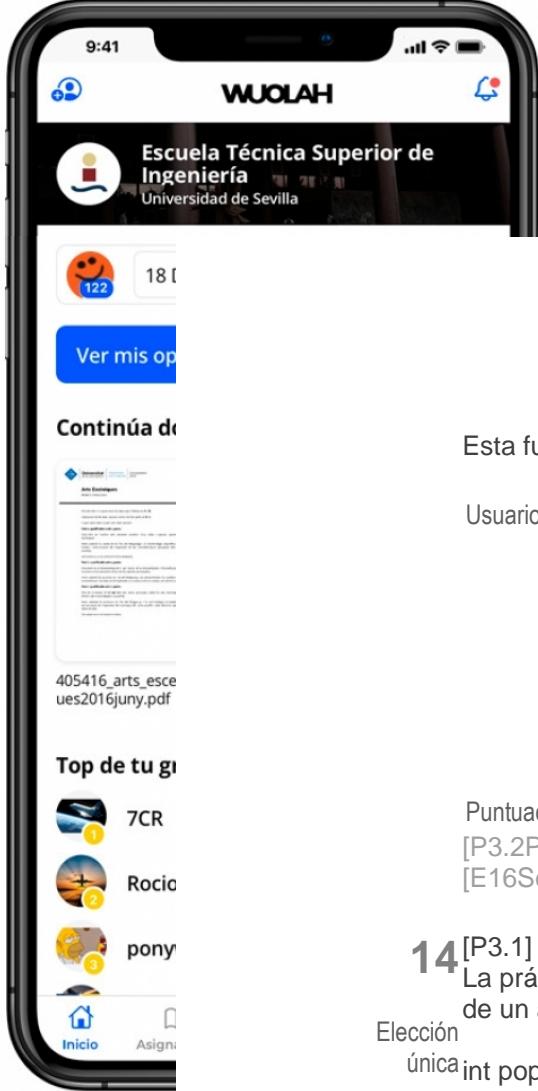
Puntuación: 1,00

## 13 [P3.2]

La práctica "parity" debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la siguiente versión de parity3:

Elección única

```
int parity3(unsigned* array, int len){
 int i,res=0,val;
 unsigned x;
 for (i=0; i<len; i++){
 x=array[i];
 val=0;
 do {
 val += x;
 x >>= 1;
 } while (x);
 val &= 0x1;
 res+=val;
 }
 return res;
}
```



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

Esta función parity3:

Usuario Profesores

- produce siempre el resultado correcto
- a) solemos escribir res+=val&0x1, en lugar de ponerlo en 2 sentencias C
- b) fallaría con array={0,1,2,3}
- c) fallaría con array={1,2,4,8}
- d) no siempre produce el resultado correcto, pero el error no se manifiesta en los ejemplos propuestos, o se manifiesta en ambos

Puntuación: 0,00

[P3.2Parity]  
[E16SepPra13]

**14** [P3.1]

La práctica "popcount" debía calcular la suma de bits de los elementos de un array. Un estudiante entrega la siguiente versión de popcount4:

Elección única

```
int popcount4(unsigned* array, int len)
{
 int i, j, res = 0;
 for(i = 0; i < len; ++i) {
 unsigned x = array[i];
 int n = 0;
 do {
 n += x & 0x01010101L;
 x >>= 1;
 } while(x);
 for(j = 16; j == 1; j /= 2){
 n ^= (n >>= j);
 }
 res += n & 0xff;
 }
 return res;
}
```

Esta función popcount4:

Usuario Profesores

produce el resultado correcto

Caso real, entregado en prácticas. La máscara está pensada para `for(j=0;j<8;j++)`. En lugar de eso, se hace `do...while(x)`, con lo cual no se ahorran iteraciones y todo el resultado queda acumulado en el LSB de `n`. El `for(j)` es absurdo, no itera ninguna vez. El resultado se extrae y acumula con `n&0xFF`. Es correcto, pero no mejora la eficiencia. popcount2

- es igual de eficiente y más elegante, porque no tiene código superfluo.
- b) fallaría con array={0,1,2,3}
  - c) fallaría con array={1,2,4,8}
  - d) no es correcta pero el error no se manifiesta en los ejemplos propuestos, o se manifiesta en ambos

Puntuación: **1,00**

[P3.1Pcount]  
[E13SepPra07]  
[E14FebPra13]  
[E16FebPra09]

En Feb14 se etiquetó incorrectamente como d. Se consideraron válidas a y d.

## 15 [P2T]

Los switches --32 y --64 para trabajar en 32bit/64bit corresponden a la herramienta...

Elección  
única Usuario Profesores

X

- a) ld sería -melf\_i386 y -melf\_xx86-64
- b) as
- c) gcc sería -m32 y -m64
- d) nm no tiene 32/64 bits

Puntuación: **-0,33**

[P2Tutorial]  
[E16SepPra02]

## 16 [P2T]

Dada la siguiente definición de datos:

Elección lista: .int 0x10000000, 0x50000000,  
única 0x10000000, 0x20000000  
longlista: .int (.lista)/4  
resultado: .quad 0x123456789ABCDEF  
formato: .ascii "suma=%llu=%llx hex\n\0"

La instrucción para copiar la dirección de memoria donde comienza lista en el registro EBX es:

Usuario Profesores

✓

- a) movl \$lista, %ebx  
Práct.2, Tut, págs.8
- b) movl (lista), %ebx
- c) movl \$lista, (%ebx)
- d) movl lista, %ebx

Puntuación: **1,00**

[P2Tutorial]

[E17JulPra05]

## 17 [P2.2]

En la práctica “media” se pide sumar una lista de 32 enteros \*con\* signo de 32bits en una plataforma de 32bits sin perder precisión, esto es, evitando overflow. ¿Cuál es el mayor valor negativo (menor en único valor absoluto) que repetido en toda la lista causaría overflow con 32bits?

Usuario Profesores

- 0xfbff ffff
- a) en cuanto sea algo más grande que 0xfc00 0000 sale overflow
  - 0xfc00 0000
  - b)  $nx32 == n << 5$  y quedaría 0x8000 0000 !!! justo para que no haya overflow
  - c) 0xffff ffff  
      eso es -1, y  $-1 \times 32 == -32$  sin problemas
  - d) 0xf000 0000  
      se pierde el signo con  $<<4$ , mucho más con  $<<5$

Puntuación: **0,00**

[P2.2SumSgn]

[E15SepPra16]

## 18 [P4T]

En una bomba como las estudiadas en prácticas, del tipo...

Elección única

```
0x0804873f <main+207>: call 0x8048504 <scanf>
0x08048744 <main+212>: mov 0x24(%esp),%edx
0x08048748 <main+216>: mov 0x804a044,%eax
0x0804874d <main+221>: cmp %eax,%edx
0x0804874f <main+223>: je 0x8048756 <main+230>
0x08048751 <main+225>: call 0x8048604 <boom>
0x08048756 <main+230>: ...
```

la contraseña es...

Usuario Profesores

- ✓     •     a) el entero 0x804a044
- b) el entero almacenado a partir de la posición de memoria 0x804a044
- c) el string almacenado a partir de la posición de memoria 0x24(%esp)
- d) ninguna de las anteriores

Puntuación: **1,00**

[P4Tutorial]  
[E13FebPra11]  
[E15FebPra17]  
[E16FebPra14]

## 19 [P5.1]

Sea un computador de 32 bits con una memoria cache L1 para datos de 32 KB y líneas de 64 bytes asociativa por conjuntos de 2 vías. Dado el siguiente fragmento de código:

única

```
int v[262144];
for (i = 0; i < 262144; i += 8)
 v[i] = 9;
```

¿Cuál será la tasa de fallos aproximada que se obtiene en la ejecución del bucle anterior?

Usuario Profesores

- 1/2 (mitad aciertos, mitad fallos)  
es un array de ints (4B) y se salta  $i+=8$ , los accesos
- a) están separados 32B y las líneas son de 64B, los accesos con índice  $i$  par son fallos, con índice impar son aciertos.
  - b) 1/8 (un fallo por cada 8 accesos)
  - c) 1 (todo son fallos)
  - d) 0 (ningún fallo)

Puntuación: 0,00

[P5.1Line]  
[E17JulPra20]

## 20 [P4T]

En la práctica de la bomba, el primer ejercicio consistía en "saltarse" las "explosiones", para lo cual se puede utilizar...

Elección

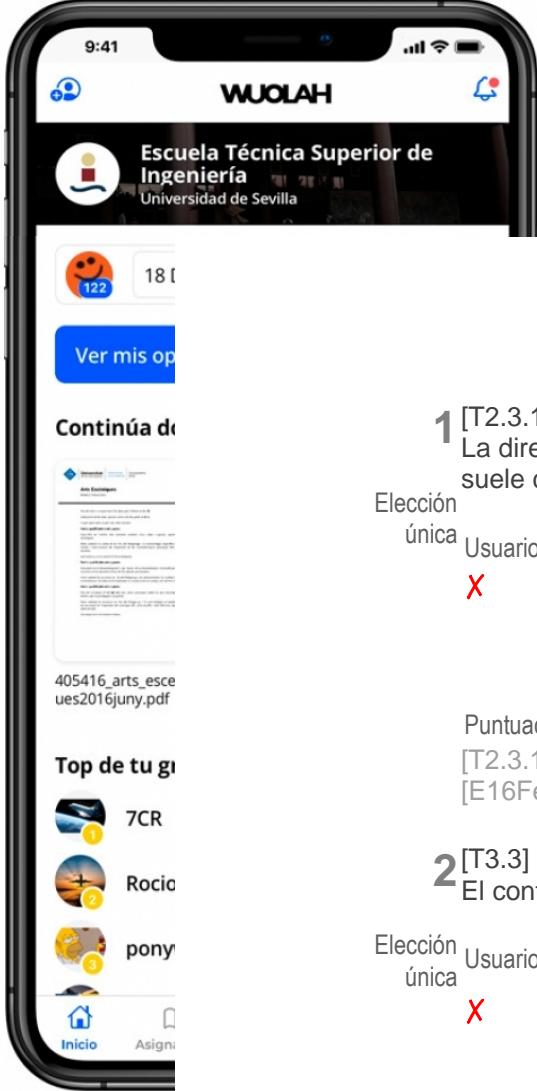
única

Usuario Profesores

- ✓      •      gdb o ddd
- a) un depurador cualquiera sirve para forzar que se cumpla la condición para "saltarse" la bomba
  - b) ddd o hexedit
  - c) hexedit u objdump
  - d) objdump o gdb

Puntuación: 1,00

[P4Tutorial]



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

1 [T2.3.1]

La dirección efectiva del primer parámetro de llamada a una función suele calcularse desde el código de la función como:

Elección  
única  
Usuario Profesores

X

- a) EBP+4
- b) EBP-8
- c) EBP+8
- d) EBP-4

Puntuación: -0,33

[T2.3.1MarcoP]  
[E16FebPra01]

2 [T3.3]

El control residual se utiliza para:

Elección  
única  
Usuario Profesores

X

- a) reducir el tiempo de ejecución de las instrucciones máquina
- b) eliminar los bits residuales de la ejecución de las microinstrucciones
- c) reducir el tamaño de la memoria de control
- d) ninguna de las anteriores es cierta

Puntuación: -0,33

[T3.3CtrlUp]  
[E17JulTeo12]

3 La codificación "Huffman" del código de operación...

Elección  
única  
Usuario Profesores

•

- a) emplea un campo de tamaño fijo para el código de operación de todas las instrucciones.
- b) permite obtener un tamaño promedio del código de operación mínimo.
- c) es la más utilizada.
- d) permite una decodificación muy sencilla de la instrucción.

Puntuación: 0,00

4 [T2.1.4]

En un sistema IA32 Linux, ¿cuál es el tamaño de un long?

Elección  
única  
Usuario Profesores

✓

•

- a) 6 bytes
- b) 4 bytes
- c) 2 bytes
- d) 8 bytes

Puntuación: **1,00**

[T2.1.4x86-64]

[E14FebTeo06]

## 5 [T6.5]

Un computador emplea un sistema de memoria principal de 128 palabras y una memoria cache de 32 palabras. La organización de la memoria cache es totalmente asociativa y el tamaño de bloque es de 8 palabras. Se emplea el algoritmo de reemplazo FIFO. Si inicialmente la memoria cache está totalmente vacía, calcule el número de fallos cuando se lee la secuencia de direcciones de la memoria principal: 0000100, 1000001, 0000101, 0010011, 0100010, 1000100, 0000111.

Elección única Usuario Profesores

- a) 5 fallos
- b) 4 fallos
- c) 3 fallos
- d) 6 fallos

Puntuación: **0,00**

[T6.5MCache]

## 6 [T6.2]

¿En qué tipo de refresco de memoria DRAM CAS# permanece a 0 después del ciclo de lectura o escritura precedente?

Elección

única Usuario Profesores

- a) RAS# antes de CAS#
- b) Sólo RAS#
- c) Refresco transparente
- d) Ninguna de las anteriores respuestas es correcta

Puntuación: **0,00**

[T6.2RAMROM]

## 7 La instrucción movl %esp,%ebp

Elección

única Usuario Profesores

- a) Intercambia los contenidos de los registros ESP y EBP.
- b) Mueve el contenido del registro ESP al registro EBP, poniendo a 0 el registro ESP.
- c) Copia el contenido del registro ESP en el registro EBP.
- d) Introduce en la pila el contenido del registro EBP.

X

Puntuación: **-0,33**

## 8 En las arquitecturas RISC hay...

Elección Usuario Profesores  
única

X

•

- a) pocos registros y muchos tipos de instrucciones.
- b) pocos modos de direccionamiento y muchos formatos de instrucción.
- c) pocas instrucciones muy rápidas con muchos modos de direccionamiento.
- d) muchos registros y pocos modos de direccionamiento.

Puntuación: -0,33

9 Suponiendo que varios dispositivos comparten una única línea de solicitud de interrupción y que varios de ellos solicitan una interrupción al mismo tiempo, ¿qué dispositivo tendría mayor prioridad a la hora de ser atendidas sus peticiones?

única

Usuario Profesores

✓

•

- a) Si se emplea una técnica de sondeo ("polling"), cualquiera de ellos
- Si se emplea una técnica de encadenamiento
- b) ("daisy-chain"), el dispositivo que este más cercano eléctricamente a la CPU
- c) Las respuestas a y b son ciertas
- d) Las respuestas a y b son falsas

Puntuación: 1,00

10 Un Pentium funcionando en modo protegido...

Elección Usuario Profesores  
única

•

- a) siempre tiene activa la unidad de segmentación
- b) siempre tiene activa la unidad de paginación
- c) siempre tiene activas la segmentación y la paginación
- d) puede tener desactivadas la segmentación o la paginación

Puntuación: 0,00

11 ¿Cuál de las siguientes afirmaciones es falsa?

Elección Usuario Profesores  
única

✓

•

- a) La operación de lectura de una celda DRAM es destructiva
- b) Una celda DRAM sólo necesita un transistor y un condensador
- Las memorias DRAM presentan generalmente una
- c) capacidad de almacenamiento mucho mayor que las SRAM
- d) Las memorias DRAM son en general mucho más rápidas que las SRAM

Puntuación: 1,00

## 12 ¿Cuál de las siguientes afirmaciones sobre el direccionamiento absoluto es falsa?

Elección Usuario Profesores  
única

- a) El rango de posiciones direccionables queda limitado por el tamaño del campo de operando.
- b) El objeto está en una posición de la memoria principal.
- c) La instrucción contiene la dirección de memoria exacta en la que se encuentra el objeto.
- d) El tamaño del operando direccionado queda limitado por el nº de bits del campo dirección de memoria.

Puntuación: 1,00

## 13 [T5.3]

¿En qué técnica para determinar la dirección de comienzo de la rutina de servicio de interrupción se fija dicha dirección en los circuitos de la Elección CPU?  
única

Usuario Profesores

- a) Direccionamiento relativo.
- b) Direccionamiento absoluto.
- c) Direcciones fijas.
- d) Envío de instrucción de bifurcación completa.

Puntuación: 1,00

[T5.3ES\_IRQ]

## 14 En el Pentium, el TLB permite buscar rápidamente...

Elección Usuario Profesores  
única

- a) El contenido de un registro caché a partir de un selector
- b) La dirección física a partir de la dirección lineal
- c) La dirección lineal a partir de la dirección virtual
- d) El descriptor de un segmento a partir de un selector

Puntuación: 0,00

## 15 ¿En qué método para determinar la dirección de comienzo de una rutina de servicio de interrupción se envía parte de dicha dirección?

Elección Usuario Profesores  
única

- a) Direccionamiento indirecto
- b) Direccionamiento absoluto
- c) Direccionamiento mediante instrucción de bifurcación
- d) Direccionamiento relativo

Puntuación: 1,00



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

Continúa d

**16** Un diseño vertical de una unidad de control...

Elección única **X** Usuario Profesores

- a) siempre es más rápido que uno horizontal
- b) en general es más lento que uno horizontal
- c) siempre supone un ahorro considerable de bits respecto a uno horizontal
- d) en general desperdicia bits respecto a uno horizontal

Puntuación: **-0,33**

**17** [T6.1]

En una jerarquía de memoria, a medida que nos alejamos del procesador:

Elección única Usuario Profesores

- ✓ • a) el tamaño de la memoria disminuye
- b) el tiempo de transferencia disminuye
- c) el tamaño de la unidad de transferencia entre dos niveles aumenta
- d) el coste por byte aumenta

Puntuación: **1,00**

[T6.1ConLoc]

[E12SepTeo29]

**18**

El programador de lenguaje ensamblador necesita conocer:

Elección única **✓** Usuario Profesores

- a) la arquitectura del ordenador.
- b) el diseño RTL del procesador.
- c) todas las respuestas son ciertas.
- d) la microarquitectura del procesador.

Puntuación: **1,00**

**19** [T6.2]

¿Cuál de las siguientes afirmaciones es falsa?

Elección única Usuario Profesores

- a) La operación de lectura de una celda DRAM es destructiva
- b) Una celda DRAM sólo necesita un transistor y un condensador
- c) Las memorias DRAM presentan generalmente una capacidad de almacenamiento mucho mayor que las SRAM
- d) Las memorias DRAM son en general mucho más rápidas que las SRAM

Puntuación: **1,00**

[T6.2RAMROM]

**20** Si queremos almacenar la palabra de 16 bits 9660h en memoria según "little-endian", quedará almacenada a partir de la posición 1000h como:

Elección única Usuario Profesores

- ✓   •   a) en el byte 1000h se guarda 96h y en el 1001h se guarda 60h
- b) en el byte 1000h se guarda 06h y en el 1001h se guarda 69h
- c) en el byte 1000h se guarda 69h y en el 1001h se guarda 06h
- d) en el byte 1000h se guarda 60h y en el 1001h se guarda 96h

Puntuación: **1,00**

**21** A medida que aumenta el tamaño de página en un sistema de memoria virtual, ¿qué ocurre con el tamaño de las tablas de páginas?

Elección única Usuario Profesores

- ✓   •   a) Crece
- b) Disminuye
- c) Permanece constante
- d) Todas las respuestas anteriores son falsas

Puntuación: **-0,33**

**22** [T3.3]

El control residual se utiliza para:

Elección única Usuario Profesores

- X   a) reducir el tiempo de ejecución de las instrucciones máquina
- b) eliminar los bits residuales de la ejecución de las microinstrucciones
- c) reducir el tamaño de la memoria de control
- d) ninguna de las anteriores es cierta

Puntuación: **-0,33**

[T3.3CtrlUp]

**23** [T6.5]

En un sistema con memoria de bytes y líneas de cache de 64 bytes, ¿dónde empieza el bloque de memoria que contiene la posición 0xBEE3DE72?

Elección única

Usuario Profesores

- ✓   •   a) 0xBEE3DE6E
- b) 0xBEE3DE70
- c) 0xBEE3DE40

d) 0x0EE3DE72

Puntuación: **1,00**

[T6.5MCache]

**24** [T2.2.2]

La instrucción xor \$3, %eax tiene como resultado:

Elección única Usuario Profesores



- a) Poner a 0 los últimos 3 bits del registro EAX
- b) Cambiar 0<->1 (complemento a 1 de) los últimos 2 bits del registro EAX
- c) Poner a 1 el último bit del registro EAX
- d) Ninguno de los anteriores resultados

Puntuación: **1,00**

[T2.2.2OpArit]

[E13FebTeo04]

**25** [T2.1.1]

¿Cuál de los siguientes microprocesadores no es de 64 bits?

Elección única Usuario Profesores



- a) Core 2
- b) Itanium
- c) Pentium III
- d) Core i7

Puntuación: **-0,33**

[T2.1.1Histor]

**26**

En una caché asociativa por conjuntos, la vía i está constituida por:

Elección única Usuario Profesores



- a) todos los bloques i-ésimos de cada conjunto
- b) todos los bloques del conjunto i
- c) todos los conjuntos del bloque i
- d) ninguna de las anteriores es cierta

Puntuación: **-0,33**

**27** [T5.3]

La técnica de sondeo, escrutinio o "polling"...

Elección única Usuario Profesores



- a) Se utiliza para identificar el destino de una interrupción
- b) Permite establecer un mecanismo de asignación de prioridades a los distintos dispositivos

- En caso de utilizarse, es necesario emplear varias
- Líneas para que los dispositivos soliciten una interrupción
  - Todas las respuestas anteriores son falsas

Puntuación: **1,00**

[T5.3ES IRQ]

**28** Con una línea de interrupción organizada en colector abierto:

Elección única Usuario Profesores

- a) se pueden conectar varios dispositivos de manera sencilla sin necesidad de circuitos combinacionales
- b) a la patilla del circuito integrado que genera esa línea hay que conectarle un transistor
- c) un dispositivo conectado a ella puede solicitar la interrupción poniendo a nivel bajo la línea
- d) a) y c) son ciertas

Puntuación: **1,00**

**29** [T2.4.1]

Comparando las convenciones de llamada de gcc Linux IA-32 con x86\_64 respecto a registros

Elección única Usuario Profesores

- X
- a) En IA-32 %esi es salva-invocado, y en x86\_64 %rsi es salva-invocado también
  - a) %rsi es 2º argumento - salva-invocante (amarillo en tr.2.4.5)
  - b) En IA-32 %ecx es salva-invocante, y en x86\_64 %rcx es salva-invocante también
  - b) %rcx es 4º argumento - salva-invocante, ok
  - c) En IA-32 %ebx es salva-invocante, pero en x86\_64 %rbx es salva-invocado
  - c) %ebx es salva-invocado (verde en tr.2.3.36)
  - d) En IA-32 %ebp es especial (marco de pila), y en x86\_64 %rbp también
  - d) %rbp es salva-invocado (verde en tr.2.4.5)

Puntuación: **-0,33**

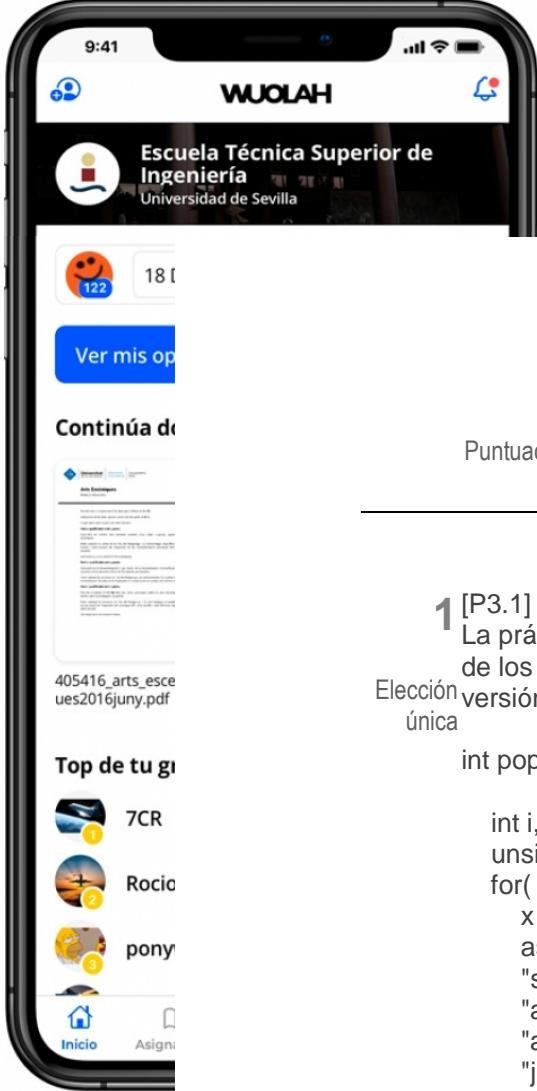
[T2.4.1x86-64]

[E15SepPra11]

**30** ¿En qué orden debería ejecutarse en una máquina de tipo pila la operación aritmética  $(a + b) / (c - d)$ ?

Elección única Usuario Profesores

- $a + b / c - d$
- $a b c d - + /$
- $a b + / c d -$



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

- d)  $a b + c d - /$
- Puntuación: 0,00
- 
- 1 [P3.1]**  
La práctica "popcount" debía calcular la suma de bits (peso Hamming) de los elementos de un array. Un estudiante entrega la siguiente Elección única versión de popcount3:

```
int popcount3(unsigned* array,
 int len){
 int i, res = 0;
 unsigned x;
 for(i = 0; i < len; i++) {
 x = array[i];
 asm("ini3: \n"
 "shr %[x] \n"
 "adc $0, %[r] \n"
 "add $0, %[x] \n"
 "jne ini3 \n"
 : [r] "+r" (res)
 : [x] "r" (x));
 }
 return res;
}
```

Esta función sólo tiene una diferencia con la versión "oficial" recomendada en clase. En concreto, una instrucción máquina en la sección `asm()` es distinta.

Esta función popcount3:

Usuario Profesores

- produce siempre el resultado correcto
- ✓ • a) Si sumar  $x+0$  activa ZF sólo puede ser porque ya era  $x==0$ , así que la lógica es equivalente a la deseada.  
b) fallaría con  $array=\{0,1,2,3\}$   
c) fallaría con  $array=\{1,2,4,8\}$   
d) no es correcta pero el error no se manifiesta en los ejemplos propuestos, o se manifiesta en ambos

Puntuación: 1,00

[P3.1Pcount]  
[E17FebPra11]

Para acabar el bucle cuando  $x==0$  hubiéramos esperado que se usara

test `%[x]`, `%[x]`, que activa el flag ZF si el resultado es 0. Notar que `and(x,x)` sólo puede ser 0 si ya era `x==0`

## 2 [P5.2]

En la práctica de cache hemos hecho una gráfica con el código `size.cc`  
¿Qué forma tiene la gráfica que se debe obtener?

Elección

única Usuario Profesores

- a) Forma de U (o V) con un tramo descendente y otro ascendente
- b) Una escalera con varios tramos horizontales
- c) Forma de U (o V) invertida, con un tramo ascendente y otro descendente
- d) Forma de /, una gráfica siempre creciente y sin escalones

Puntuación: 1,00

[P5.2Size]

[E14FebPra18]

## 3 [P3T]

En 80x86, los parámetros a las subrutinas se pueden pasar:

Elección

única Usuario Profesores

- a) a través de variables globales
- b) a través de los registros
- c) a través de la pila
- d) todas las anteriores son ciertas

Puntuación: 1,00

[P3Tutorial]

[P2Tutorial]

## 4 [P2T]

¿Qué hace gcc -O?

Elección

única Usuario Profesores

- a) Compilar sin optimización, igual que -O0
- b) Compilar con optimización, igual que -O1
- c) Compilar .c → .o
- d) Compilar .s → .o

Puntuación: 1,00

[P2Tutorial]

[T2.1.2Lngies]

[E13SepPra12]

**5** [P2T]

¿Qué modificador (switch) de gcc hace falta para compilar una aplicación de 32 bits en un sistema de 64 bits?

Elección

única Usuario Profesores



- a) -m32
- b) -64
- c) -m64
- d) -32

Puntuación: **1,00**

[P2Tutorial]

[E13SepPra14]

**6** [P3T]

¿Cuál de los siguientes registros tiene que ser salvaguardado (si va a modificarse) dentro de una subrutina según la convención cdecl para

Elección

IA32?

única

Usuario Profesores

- a) eax
- b) ebx
- c) ecx
- X d) edx

Puntuación: **-0,33**

[P3Tutorial]

[E14FebPra15]

**7** [P3.1]

¿Cuál es el popcount (peso Hamming, nº de bits activados) del número 19?

Elección

única Usuario Profesores



- a) 2
- b) 3  
19 = 0x13 = 16+3 = 0b0001 0011 -> popcount 3
- c) 4
- d) 5

Puntuación: **1,00**

[P3.1Pcount]

[T2.2.5BWhile]

[E17FebPra10]

**8** [P2T]

Dada la siguiente definición de datos:

lista: .int 0x10000000, 0x50000000,  
0x10000000, 0x20000000

Elección longlista: .int (-lista)/4  
única resultado: .quad 0x123456789ABCDEF  
formato: .ascii "suma=%llu=%llx hex\n\n0"

la instrucción movl longlista, %ecx copia el siguiente valor:

Usuario Profesores

- a) 32
- b) 4  
Práct.2, Tut, pág.7
- c) 16
- d) 8

Puntuación: **0,00**

[P2Tutorial]  
[E17JulPra06]

**9** [P3T]

¿Cuál de las siguientes afirmaciones es cierta respecto al lenguaje C?

Elección  
única

Usuario Profesores



- Pasar a una función un puntero a una variable se traduce en introducir en la pila el valor de la variable `ptr=val`
- En lenguaje C, al llamar a una subrutina o función se introducen los parámetros en la pila y después se realiza una llamada a la subrutina suponiendo convención cdecl x86, porque x86-64 usa regs.
- Antes de volver de la rutina llamada, el programa en C se encarga de quitar de la pila los parámetros de llamada realizando varios pop no, quita parámetros el invocante, tras retornar de la llamada
- Los parámetros se introducen en la pila en el orden en el que aparecen en la llamada de C, es decir, empezando por el primero y acabando por el último no, de derecha a izquierda

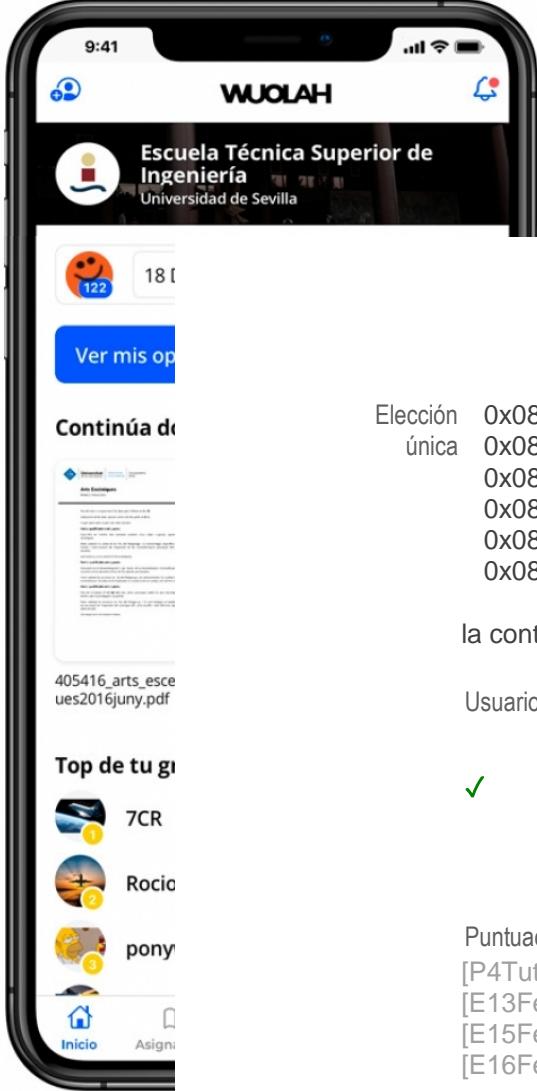
Puntuación: **1,00**

[P2Tutorial]  
[P3Tutorial]  
[E13SepPra16]  
[E17JulPra10]

**10** [P4T]

En una bomba como las estudiadas en prácticas, del tipo...

0x0804873f <main+207>: call 0x8048504 <scanf>



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the App Store GET IT ON Google Play

Elección única 0x08048744 <main+212>; mov 0x24(%esp),%edx  
 0x08048748 <main+216>; mov 0x804a044,%eax  
 0x0804874d <main+221>; cmp %eax,%edx  
 0x0804874f <main+223>; je 0x8048756 <main+230>  
 0x08048751 <main+225>; call 0x8048604 <boom>  
 0x08048756 <main+230>; ...

la contraseña es...

Usuario Profesores

- a) el entero 0x804a044
- b) el entero almacenado a partir de la posición de memoria 0x804a044
- c) el string almacenado a partir de la posición de memoria 0x24(%esp)
- d) ninguna de las anteriores

Puntuación: 1,00

[P4Tutorial]  
 [E13FebPra11]  
 [E15FebPra17]  
 [E16FebPra14]

11 [P2.1]

Elección única En la práctica "media" se pide sumar una lista de 32 enteros SIN signo de 32 bits en una plataforma de 32 bits sin perder precisión, esto es, evitando perder acarreos. De entre los siguientes, ¿cuál es el mínimo valor entero que repetido en toda la lista causaría acarreo con 32 bits (sin signo)? Se usa notación decimal y espacios como separadores de millares/millones/etc.

Usuario Profesores

- a) 10 000 000
- b) 100 000 000  
No llega, 100 millones < 128M
- c) 1 000 000 000  
Se pasa, 1000 millones >> 128M
- d) 10 000 000 000

Puntuación: -0,33

[P2.1SumUns]  
 [E17FebPra08]

32bit sin signo llega a  $2^{32}-1 = 4G-1 \sim 4\ 000\ 000\ 000$ . Habría acarreo si llegáramos a 4G. Con 32 elementos iguales, tendrían que valer  $4G/32 = 1G/8 = 128M$ .

12 [P3T]

¿En qué registro se pasa el primer argumento a una función en Linux gcc x86-64?

Elección Usuario Profesores

única ✓

- a) edi
- b) edx
- c) esi
- d) ecx

Puntuación: **1,00**

[P3Tutorial]

[T2.4.1x86-64]

[E17JulPra12]

**13** [P4T]

En una bomba como las estudiadas en prácticas, del tipo...

Elección 0x08048705 <main+149>: call 0x80484c4 <gettimeofday>

única ...

0x08048718 <main+168>: cmp \$0x5,%eax

0x0804871b <main+171>: jle 0x8048722 <main+178>

0x0804871d <main+173>: call 0x8048604 <boom>

0x08048722 <main+178>: ...

ejecutada paso a paso con el depurador ddd, interesaría...

Usuario Profesores

- a) ejecutar hasta jle, ajustar %eax a 6, y continuar ejecutando paso a paso
  - b) ejecutar hasta jle, ajustar %eax a 4, y continuar ejecutando paso a paso
  - c) cambiar jle por jmp usando ddd o un editor hex,
  - d) salvar el programa y reiniciar la depuración con el nuevo ejecutable
- Ninguna de las opciones anteriores es de interés
- X d) (bien porque no se pueda hacer eso o porque no sirva para evitar la bomba)

Puntuación: **-0,33**

[P4Tutorial]

[E13FebPra12]

**14** [P3.2]

La práctica “parity” debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la

Elección siguiente versión de parity4:  
única

```
int parity4(unsigned* array, int len){
 int val,i,res=0;
 unsigned x;
 for (i=0; i<len; i++){
 x=array[i];
 val=0;
```

```

asm("\n"
"ini3: \n\t"
"xor %[x],%[v] \n\t"
"shr %[x] \n\t"
"test %[x], %[x]\n\t"
"jne ini3 \n\t"
:[v]" +r" (val)
:[x] "r" (x)
);
val = val & 0x1;
res+=val;
}
return res;
}

```

La sentencia `asm()` del listado anterior tiene las siguientes restricciones

Usuario Profesores

- a) ninguna
- b) un registro y dos sobreescritos (clobber)
- c) arquitectura de 32 bits
- d) [v] cuenta como salida y entrada, [x] como entrada

Puntuación: **0,00**

[P3.2Parity]

[E16SepPra15]

**15** [P1]

¿De qué tipo son los procesadores Intel que usamos en los laboratorios?

Elección

única

Usuario Profesores



- a) little-endian

el concepto de endian no es aplicable a estas

- b) máquinas, ya que un registro del procesador no cabe en una posición de memoria

- c) big-endian

- d) puede ajustarse mediante un bit de control en el registro CR0 que funcionen como little- o big-endian

Puntuación: **1,00**

[P1]

[T1.1UniFun]

[E15FebPra01]

**16** [P1]

La etiqueta del punto de entrada a un programa ensamblador en el entorno de las prácticas 1 a 4 (GNU/as Linux x86) es:

Elección Usuario Profesores

única **X**

- a) \_\_init
- b) \_\_main
- \_\_start  
        P3 y P4 se redactan en C.  
        P1 y P2 sí son en ensamblador GNU/as Linux x86.
- c) Incluso en P2 se llega a usar main para ensamblar  
        con gcc ya que usamos printf.  
        En cualquier caso, las otras opciones son  
        descabelladas
- d) .LO

Puntuación: **-0,33**

[P1]

[E17FebPra03]

**17** [T2.1.2]

En X86-64, el registro contador de programa se denomina:

Elección Usuario Profesores

única **✓**

- a) RIP
- b) R15
- c) EIP
- d) IP

Puntuación: **1,00**

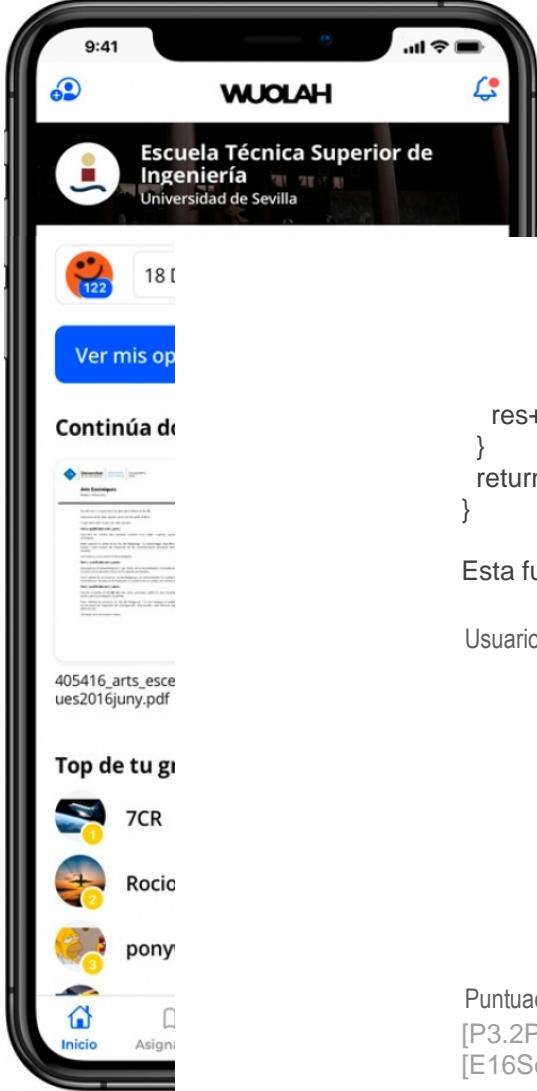
[T2.1.2Lngjes]

[E16FebTeo03]

**18** [P3.2]

La práctica “parity” debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la Elección siguiente versión de parity4:  
única

```
int parity4(unsigned* array, int len){
 int val,i,res=0;
 unsigned x;
 for (i=0; i<len; i++){
 x=array[i];
 val=0;
 asm("\n"
 "ini3: \n\t"
 "xor %[x],%[v] \n\t"
 "shr %[x] \n\t"
 "test %[x], %[x]\n\t"
 "jne ini3 \n\t"
 "[v]"+"r" (val)
 "[x]" "r" (x)
);
 val = val & 0x1;
```



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

```
res+=val;
}
return res;
}
```

Esta función parity4:

Usuario Profesores

produce siempre el resultado correcto  
solemos escribir res+=val&0x1, en lugar de ponerlo

- a) en 2 sentencias C  
adicionalmente, shr afecta al flag ZF de manera que sobra test %[x],%[x]
- b) fallaría con array={0,1,2,3}
- c) fallaría con array={1,2,4,8}
- d) no siempre produce el resultado correcto, pero el error no se manifiesta en los ejemplos propuestos, o se manifiesta en ambos

Puntuación: 0,00

[P3.2Parity]  
[E16SepPra14]

**19** [P2T]

Dada la siguiente definición de datos:

Elección lista: .int 0x10000000, 0x50000000,  
única 0x10000000, 0x20000000  
longlista: .int (. -lista)/4  
resultado: .quad 0x123456789ABCDEF  
formato: .ascii "suma=%llu=%llx hex\n\0"

la llamada correcta a printf será:

Usuario Profesores

```
push resultado
push resultado+4
push resultado
push resultado+4
```

- a) push \$formato  
call printf  
add \$20, %esp  
No. Debe ser little-endian, los push resultado están desordenados
- push resultado+4  
push resultado
- b) push \$formato  
call printf  
add \$12, %esp  
No. 2 veces resultado

- ✓ •
- c) push resultado  
push resultado+4  
push \$formato  
call printf  
add \$12, %esp  
No. 2 veces resultado  
push resultado+4  
push resultado  
push resultado+4
  - d) push resultado  
push \$formato  
call printf  
add \$20, %esp
- Sí.  $20 = 8 (\%llx) + 8 (\%llu) + 4 (\$formato)$

Puntuación: **1,00**

[P2Tutorial]

[E17JulPra08]

notar que formato lleva %llu y %llx, para imprimir (se adivina que) dos veces resultado

**20** [P2T]

¿Qué modificador (switch) de gcc hace falta para compilar una aplicación de 32 bits en un sistema de 64 bits?

Elección

única Usuario Profesores

- ✓ •
- a) -march64
  - b) -m32
  - c) -march32
  - d) -m64

Puntuación: **1,00**

[P2Tutorial]

[E17FebPra02]

---

1

¿En qué técnica para determinar la dirección de comienzo de la rutina de servicio de interrupción se fija dicha dirección en los circuitos de la CPU?

Elección

única Usuario Profesores

- ✓ •
- a) Direccionamiento relativo.
  - b) Direccionamiento absoluto.
  - c) Envío de instrucción de bifurcación completa.
  - d) Direcciones fijas.

Puntuación: **1,00**

**2** [T5.2]

La E/S programada:

Elección Usuario Profesores  
única



- Empeora las prestaciones globales del sistema respecto a la E/S por interrupciones porque una instrucción de transferencia individual de datos con la interfaz del periférico (por ej. IN, OUT) es más lenta en E/S programada que en E/S por interrupciones.
- Mejora las prestaciones globales del sistema respecto a la E/S por interrupciones porque la CPU es más rápida que el controlador de interrupciones y la interfaz del periférico.
- Empeora las prestaciones globales del sistema respecto a la E/S por interrupciones porque la CPU debe encargarse de la sincronización con la interfaz del periférico haciendo una espera activa.
- Mejora las prestaciones globales del sistema respecto a la E/S por interrupciones porque la CPU tiene el control de toda la operación.

Puntuación: **1,00**

[T5.2ESProg]

[T5.3ES IRQ]

[E17FebTeo21]

**3** ¿Cuál de las siguientes afirmaciones es falsa?

Elección Usuario Profesores  
única



- a) la anchura del bus de datos es siempre de 16 bytes
- b) el bus de datos es bidireccional
- c) el bus de direcciones es unidireccional
- d) el bus de control puede transportar señales de estado

Puntuación: **1,00**

**4** [T5.1]

Algunas de las ventajas de la E/S mapeada en memoria frente a la E/S aislada o independiente son:

Elección Usuario Profesores  
única



- a) Los puertos de E/S no ocupan direcciones de memoria.
- b) Facilita la protección de E/S.
- c) El diseño de la CPU es más sencillo.
- d) Todas las respuestas anteriores son ciertas.

Puntuación: **-0,33**

[T5.1FunE/S]

## 5 La "postescritura (write-back) marcada"

- Elección única Usuario Profesores
- a) es más eficiente que la "postescritura siempre"  
b) requiere más bits de modificación ("bits sucios")  
c) cuando aumenta el número de vías  
d) provoca una menor tasa de faltas que la "postescritura siempre"

Puntuación: 0,00

## 6 [T6.3]

Con 8 circuitos de memoria RAM de 1K x 8 se puede crear un memoria de:

- Elección única Usuario Profesores
- a) 1K x 64  
b) 8K x 8  
c) 2K x 32  
✓ • d) Todas las combinaciones anteriores son posibles

Puntuación: 1,00

[T6.3Diseño]

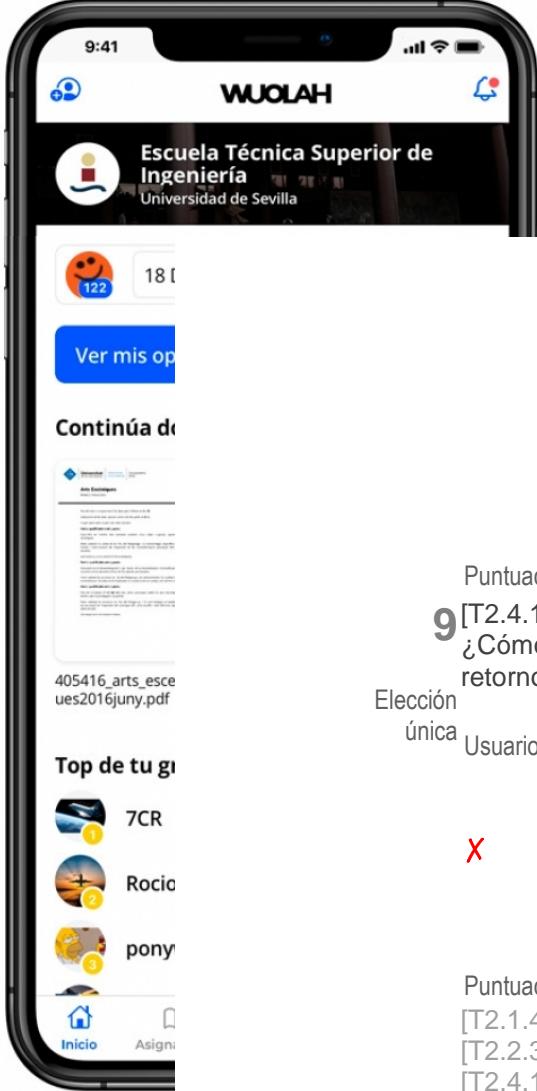
## 7 La conexión de las salidas de tres registros hacia un bus común en el camino de datos puede realizarse usando...

- Elección única Usuario Profesores
- ✓ • a) dos buffers triestado  
b) tres conexiones directas al bus común  
c) dos multiplexores de 2 a 1  
d) tres demultiplexores

Puntuación: 1,00

## 8 Respecto a las unidades de control nanoprogramadas:

- Elección única Usuario Profesores
- a) La realización nanoprogramada de una unidad de control es más rápida que la micropogramada.  
La anchura de la memoria de nanoprograma es la misma que la de memoria de micropograma en un diseño de la misma unidad de control que no usara nanoprogramación.  
c) El diseño de las unidades de control nanoprogramadas debe ser vertical.



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

Suponiendo una memoria de microprograma con n microinstrucciones de w bits cada una, de las cuales  $2^m$  son distintas, el ahorro en bits si se utiliza nanoprogramación es  $(n \cdot m + 2^m \cdot w) - n \cdot w$ .

Puntuación: 0,00

9 [T2.4.1]

¿Cómo se devuelve en ensamblador x86-64 Linux gcc el valor de retorno de una función long int al terminar ésta?

Elección  
única  
Usuario Profesores

X

- a) La instrucción RET lo almacena en un registro especial de retorno.
- b) Por convención se guarda en %eax.
- c) Se almacena en pila justo encima de los argumentos de la función.
- d) Ninguna de esas formas es la correcta.

Puntuación: -0,33

[T2.1.4x86-64]  
[T2.2.3CodCon]  
[T2.4.1x86-64]  
[P3Tutorial]  
[E12SepTeo02]

10 [T2.3.1]

¿Cuál de las siguientes afirmaciones es falsa respecto al lenguaje C?

Elección  
única  
Usuario Profesores

✓

•

- Antes de volver de la rutina llamada, el programa
- a) en C se encarga de quitar de la pila los parámetros de llamada realizando varios pop
  - b) Los parámetros se introducen en la pila en el orden inverso a como aparecen en la llamada de C, es decir, empezando por el último y acabando por el primero
  - c) Pasar a una función un puntero a una variable se traduce en introducir en la pila el valor de la dirección de memoria donde está almacenada la variable
  - d) En lenguaje C, al llamar a una subrutina o función se introducen los parámetros en la pila y después se realiza una llamada a la subrutina

Puntuación: 1,00

[T2.3.1MarcoP]  
[P3Tutorial]  
[E13FebPra06]

**11** [T6.2]

¿En qué tipo de ciclo de refresco se hace RAS# = 0?

Elección única Usuario Profesores

- a) Sólo RAS#
- b) CAS# antes de RAS#
- c) Refresco transparente
- d) En todos los anteriores

Puntuación: **0,00**

[T6.2RAMROM]

**12** [T6.3]

Una puerta AND con 16 entradas conectada a un bus de direcciones de 16 bits, con todos los bits negados excepto A10 y A6, permite seleccionar un dispositivo (con CS activa en alta) en la dirección:

Usuario Profesores

**X**

- a) 0x0220
- b) 0xFBFF
- c) 0x0440
- d) 0xFDDF

Puntuación: **-0,33**

[T1.3EstBus]

[T6.3Diseño]

[E17FebTeo22]

**13**

Respecto a registros salva-invocante y salva-invocado en GCC/Linux IA32, ¿cuál de éstos es de distinto tipo que el resto?

Elección única Usuario Profesores

- a) esi
- b) ebx
- c) edi
- d) eax

Puntuación: **1,00**

**14**

Señale cuál de las siguientes opciones no es un modo para llevar a cabo la transferencia de datos entre el computador y los dispositivos de E/S externos:

Elección única Usuario Profesores

- a) E/S programada
- b) E/S por flanco
- c) Acceso directo a memoria (DMA)
- d) E/S iniciada por interrupción

Puntuación: **1,00**

**15** [T3.3]

Un procesador con una unidad de control microprogramada tiene una memoria de control de 300 palabras de 100 bits, de las que 200 son diferentes. Si se rediseñara como unidad de control nanoprogramada, ¿qué tamaño ocuparía la nanomemoria que contiene las microinstrucciones completas sin repeticiones?

Usuario Profesores

- a) 22400 bits
  - b) 21600 bits
  - c) 30000 bits
  - d) 20000 bits
- ✓ • 200 uinstr. x 100 bits

Puntuación: **1,00**

[T3.3CtrlUp]

[E17FebTeo15]

**16**

Un controlador de E/S posee un buffer para el almacenamiento temporal de los datos con una capacidad de 64 KB. En un instante determinado inicia una operación de E/S con una impresora a una velocidad de transferencia de 128 KB/s. Si el controlador de E/S recibe la información que debe enviar a la impresora a una velocidad de 1 MB/s, ¿cuánto tiempo tardará en llenarse por primera vez el buffer suponiendo que inicialmente está vacío, y que recibe y envía información simultáneamente de forma continua?

Usuario Profesores

- a) 71 ms
- b) 143 ms
- c) No se puede calcular
- d) Ninguna de las anteriores

Puntuación: **1,00**

**17**

¿Es posible utilizar 4 GB de memoria en un sistema cuya CPU emplea E/S mapeada en memoria, cuyo bus de direcciones es de 32 bits y que tiene al menos un puerto de E/S? Supondremos que no se puede emplear ninguna técnica de extensión del bus de direcciones.

Usuario Profesores

- a) Sí
- b) No
- c) Depende de si el número puertos de E/S es muy elevado
- d) Ninguna de las respuestas anteriores es correcta

Puntuación: **-0,33**

**18** [T2.2.1]

¿Cuál de las siguientes instrucciones máquina copia en EAX el entero

Elección almacenado en la posición de memoria cuya dirección efectiva es el único resultado de la operación EDX\*4 + EBX?

Elección única Usuario Profesores

- X a) movl 4(%edx, %edx), %eax  
• b) leal (%ebx, %edx, 4), %eax  
c) movl (%ebx, %edx, 4), %eax  
d) leal 4(%edx, %edx), %eax

Puntuación: -0,33

[T2.2.1ModDir]

[P2Tutorial]

[E17JulPra02]

**19** ¿Cuál de los siguientes elementos no forma parte de un canal de un controlador de acceso directo a memoria?

Elección única Usuario Profesores

- ✓ • a) Registro contador  
b) Registro de prioridades  
c) Registro de órdenes  
d) Registro de dirección

Puntuación: 1,00

**20** [T2.2.3]

La instrucción IA32 test sirve para...

Elección única Usuario Profesores

- ✓ • a) Realizar la operación resta (a-b) pero no guardar el resultado, sino simplemente ajustar los flags  
b) Mover el operando fuente al destino, pero sólo si se cumple la condición indicada  
Realizar la operación and lógico bit-a-bit (a&b) pero  
c) no guardar el resultado, sino simplemente ajustar los flags  
d) Testear el código de condición indicado, y poner un byte a 1 si se cumple

Puntuación: 1,00

[T2.2.3CodCon]

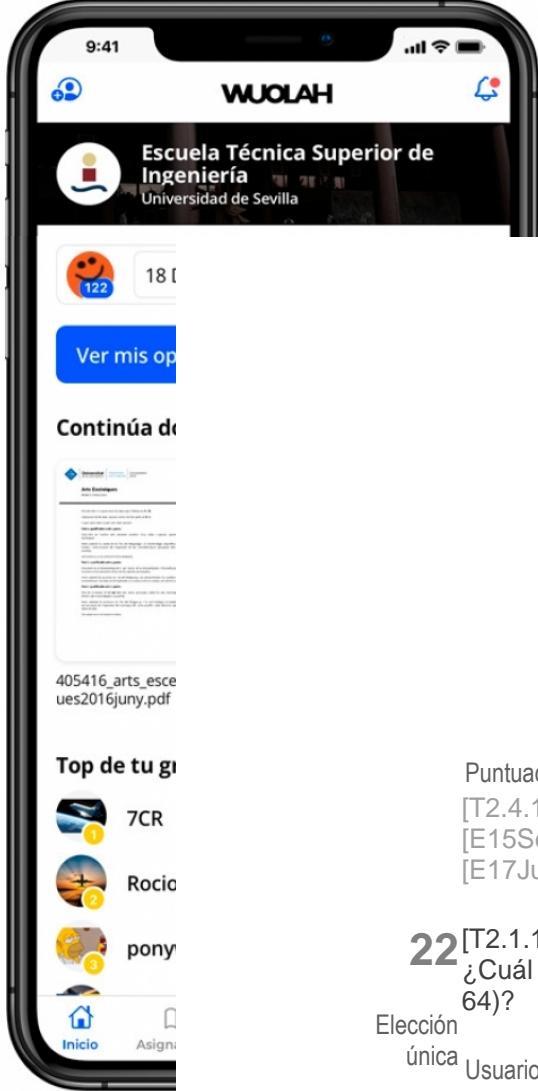
[E13SepTeo04]

**21** [T2.4.1]

Comparando las convenciones de llamada de gcc Linux IA32 con x86-64 respecto a registros

Elección única Usuario Profesores

- X a) En IA32 %esi es salva-invocado, y en x86-64 %rsi es salva-invocado también



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

• %rsi es 2º argumento - salva-invocante (amarillo en tr.2.4.5)

En IA32 %ecx es salva-invocante, y en x86-64

b) %rcx es salva-invocante también

%rcx es 4º argumento - salva-invocante, ok

En IA32 %ebp es especial (marco de pila), y en

c) x86-64 %rbp también

%rbp es salva-invocado (verde en tr.2.4.5)

En IA32 %ebx es salva-invocante, pero en x86-64

d) %rbx es salva-invocado

%ebx es salva-invocado (verde en tr.2.3.36)

Puntuación: **-0,33**

[T2.4.1x86-64]

[E15SepPra11]

[E17JulTeo06]

**22** [T2.1.1]

¿Cuál fue el primer procesador de Intel de 64-bits en la familia x86(-64)?

Elección única  
Usuario Profesores

- a) Pentium 4F
- b) 386
- c) Core i7
- d) 8086

Puntuación: **1,00**

[T2.1.1Histor]

[E14SepTeo01]

**23** [T6.1]

Sea un computador de 32 bits que dispone de una memoria cache de 512 KB y líneas de 64 bytes. ¿Cuántas líneas tiene la cache?

Elección única  
Usuario Profesores

- a) 8192
- b) 1024
- c) 65536
- d) 64

Puntuación: **-0,33**

[T6.1ConLoc]

[E14FebTeo20]

**24** [T2.3.1]

En la secuencia de programa siguiente:

804854e:e8 3d 06 00 00 call 8048b90 <main>

8048553:50 pushl %eax

Elección

única ¿cuál es el valor que introduce en la pila la instrucción call?

Usuario Profesores

- a) 0x8048b90
- b) 0x804854f
- c) 0x8048553
- d) 0x804854e

Puntuación: **0,00**

[T2.3.1MarcoP]

[E16FebTeo14]

**25** [T5.2]

¿Cuántos puertos de E/S permite manejar la interfaz de periféricos programable 8255?

Elección

única

Usuario Profesores

- a) 2 puertos de 16 bits
- b) 3 puertos de 8 bits
- c) 4 puertos de 32 bits
- d) Todas las respuestas anteriores son falsas

Puntuación: **0,00**

[T5.2ESProg]

**26**

Si un procesador no segmentado necesita 5 ns para leer una instrucción de memoria, 2 ns para decodificar la instrucción, 3 ns para leer del banco de registros, 3 ns para realizar el cálculo requerido por la instrucción, y 2 ns para escribir el resultado en el banco de registros, ¿cuál es la frecuencia de reloj máxima del procesador?

Usuario Profesores

- a) 200 MHz
- b) 66,67 MHz
- c) 500 MHz
- d) 40 MHz

Puntuación: **0,00**

**27** [T5.1]

Respecto a si un computador dispone de E/S independiente (separada) o usa E/S mapeada a memoria:

Elección

única

Usuario Profesores

- Si el encapsulado del procesador no dispone de
- a) patilla IO/M# (ni equivalentes), el computador sólo dispone de E/S separada
  - ✓      • Si el repertorio del procesador tiene instrucciones
  - b) del tipo IN y OUT, es que el computador dispone de E/S separada

- Si el repertorio del procesador tiene instrucciones  
c) del tipo LOAD y STORE, el computador sólo  
dispone de E/S mapeada a memoria  
Si el encapsulado (chip) del procesador tiene patilla  
d) (pin) IO/M# (o patillas equivalentes), eso evidencia  
que el computador usa E/S mapeada a memoria

Puntuación: **1,00**

[T5.1FunE/S]

[E15FebTeo16]

**28** [T2.2.4]

Las instrucciones JB y JNAE del Pentium provocan un salto si...

Elección única Usuario Profesores

- a) ZF == 0
- b) CF == 1
- c) ZF != SF
- d) SF == 1

X Puntuación: **-0,33**

[T2.2.4SalCon]

**29** [T5.2]

¿Cuántos puertos puede gestionar la interfaz de periféricos  
programable 8255?

Elección única Usuario Profesores

- a) Uno de 24 bits
- b) Dos de 12 bits
- c) Tres de 8 bits
- d) Todas las combinaciones anteriores son válidas

Puntuación: **0,00**

[T5.2ESProg]

**30**

Utilizando E/S programada y como modo de direccionamiento  
selección lineal, ¿cuántos periféricos podrían conectarse a un 8086?

Elección única Usuario Profesores

- a) 20 periféricos
- b) 16 periféricos
- c) Cualquier número de periféricos
- d) 8 periféricos

Puntuación: **0,00**

### **1** [P3T]

¿En qué registro se pasa el primer argumento a una función según el estándar \_cdecl en una arquitectura IA32?

Elección

única

Usuario Profesores

**X**

- a) edi
- b) esi
- c) eax
- d) Las anteriores respuestas son erróneas

Puntuación: **-0,33**

[P3Tutorial]

[T2.4.1x86-64]

[E15FebPra02]

### **2** [P2.2]

Elección En la práctica “media” se pide sumar una lista de enteros \*con\* signo de 32 bits en una plataforma de 32 bits sin perder precisión, esto es, evitando overflow. ¿Cuál es el menor valor positivo que repetido en los únicos dos primeros elementos de la lista causaría overflow con 32 bits al realizar la suma de esos dos primeros elementos de la lista?

Usuario Profesores

- ✓**
- 

- a) 0x0400 0000  
suma 0x0800 0000 sin problema
- b) 0x8000 0000  
es negativo, el enunciado pide menor valor positivo
- c) 0x0800 0000  
suma 0x1000 0000 sin problema
- d) 0x4000 0000  
0x8000 0000 sería negativo en 32bit

Puntuación: **1,00**

[P2.2SumSgn]

[E17JulPra09]

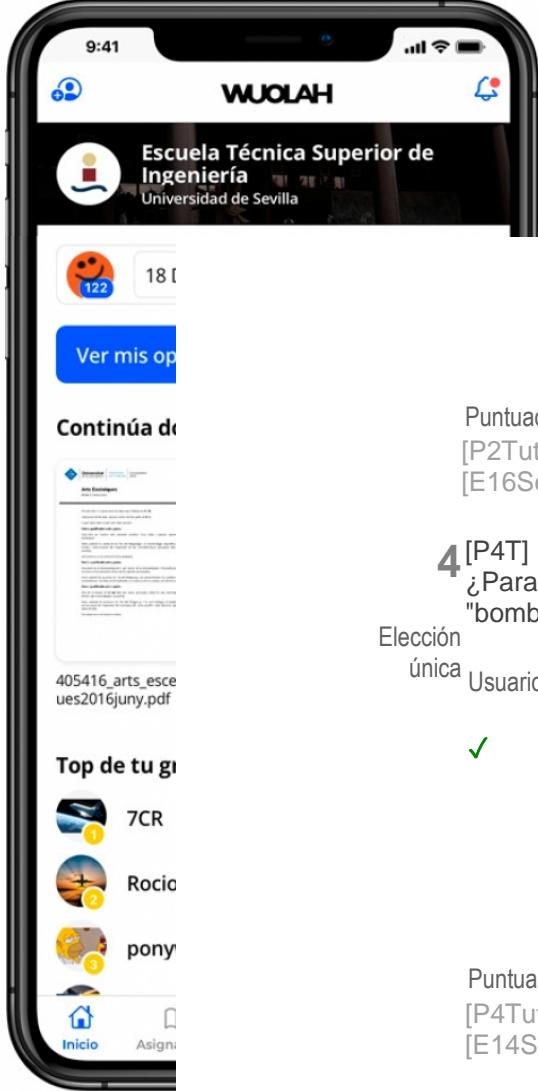
### **3** [P2T]

Elección El switch de gcc para que únicamente compile de lenguaje C a ensamblador, y no realice ningún paso adicional (ensamblar, enlazar, etc), es...  
única

Usuario Profesores

- ✓**
- 

- a) -g  
para incorporar info depuración
- b) -c  
de C/asm a objeto, .c/.s→.o
- c) -o  
para nombrar el ejecutable
- d) -S



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

Puntuación: 1,00

[P2Tutorial]  
[E16SepPra01]

4 [P4T]

¿Para qué se utiliza la función gettimeofday en la práctica de la "bomba digital"?

Elección  
única Usuario Profesores

- Para lanzar un error cuando el usuario tarde
- a) demasiado tiempo en introducir la contraseña o el PIN
- b) Para cronometrar y poder comparar las duraciones de las distintas soluciones del programa
- c) Para cifrar la contraseña en función de la hora actual
- d) Para imprimir la hora en la pantalla

Puntuación: 1,00

[P4Tutorial]  
[E14SepPra20]

5 [P2T]

La siguiente línea en la sección de datos de un programa en ensamblador de IA32

Elección  
única result: .int 0,0

Usuario Profesores

- a) Reserva espacio para un único entero, inicializado a 0,0
- b) Reserva espacio para un único entero, inicializado a 0, en la posición de memoria 0
- c) Reserva espacio para un entero, inicializado a 0, seguido de un dato de tamaño indefinido, también inicializado a 0
- d) Reserva espacio para dos enteros, inicializados ambos a 0

Puntuación: 1,00

[P2Tutorial]  
[E17FebPra04]

6 [P2A2]

¿Cuál de los siguientes grupos de instrucciones IA32 sólo modifican los indicadores de estado sin almacenar el resultado de la operación?

Elección  
única Usuario Profesores

- a) IMUL, IDIV
- b) AND, OR, XOR

- ✓ • c) CMP, TEST  
d) ADC, SBB

Puntuación: **1,00**

[P2Apéndice2]

[T2.2.2OpArit]

[T2.2.3CodCon]

[E12FebTeo16]

## 7 [P4T]

En la práctica de la bomba necesitamos estudiar el código máquina de la bomba del compañero. A veces dicho código no se visualiza directamente en el depurador ddd, y algunas de las técnicas que se única pueden probar para conseguir visualizarlo son... (marcar la opción \*falsa\*)

Usuario Profesores

- X
- a) comprobar que está activado el panel View → Machine Code Window
  - b) asegurarse de que se ha escrito correctamente el nombre del ejecutable
  - c) escribir info line main en el panel de línea de comandos gdb  
recompilar con información de depuración, por si se nos había olvidado, ya que sin -g el ejecutable no
  - d) contiene información de depuración  
sí, claro, como que nos van a pasar el código fuente... :-)

Puntuación: **-0,33**

[P4Tutorial]

[E17JulPra14]

## 8 [P1]

El punto de entrada de un programa ensamblador en GNU/as Linux x86 se llama

Elección

única Usuario Profesores

- ✓ • a) \_init  
b) main  
c) \_start  
d) begin

Puntuación: **1,00**

[P1]

[E14FebPra06]

## 9 [P2.2]

En la práctica “media” se pide sumar una lista de 32 enteros \*con\* signo de 32bits en una plataforma de 32bits sin perder precisión, esto

Elección es, evitando overflow. ¿Cuál es el mayor valor negativo (menor en única valor absoluto) que repetido en toda la lista causaría overflow con 32bits?

Usuario Profesores

- a) 0xf000 0000  
se pierde el signo con <<4, mucho más con <<5
- b) 0xffff ffff  
eso es -1, y  $-1 \times 32 == -32$  sin problemas  
0xfbff ffff
- c) en cuanto sea algo más grande que 0xfc00 0000 sale overflow  
0xfc00 0000
- d)  $n \times 32 == n \ll 5$  y quedaría 0x8000 0000 !!! justo para que no haya overflow

Puntuación: 1,00

[P2.2SumSgn]

[E15SepPra16]

## 10 [P3T]

En la convención cdecl estándar para arquitecturas x86 de 32 bits, cuál de las siguientes afirmaciones es cierta:

Elección

única

Usuario Profesores

- a) Los parámetros se pasan en pila, de derecha a izquierda; es decir, primero se pasa el último parámetro, después el penúltimo... y por fin el primero
- b) Solamente es necesario salvar el registro EAX
- c) Los registros EBX, ESI y EDI son salva-invocante
- d) Ninguna de las anteriores es cierta

Puntuación: 1,00

[P3Tutorial]

[E14FebPra01]

## 11 [P2.2]

En la práctica "media" se pide sumar una lista de 32 enteros CON signo de 32bits en una plataforma de 32bits sin perder precisión, esto es, evitando desbordamiento. ¿Cuál es el valor negativo más pequeño (en valor absoluto) que repetido en toda la lista causaría desbordamiento con 32bits (en complemento a 2)?

Elección

única

Usuario Profesores

- a) 0xf800 0001
- b) 0xfc00 0000
- c) 0xf800 0000
- d) 0xfbff ffff

Puntuación: **1,00**

[P2.2SumSgn]  
[E14SepPra04]  
[E16FebPra07]

## 12 [P2T]

Los switches --32 y --64 para trabajar en 32bit/64bit corresponden a la herramienta...

Elección  
única Usuario Profesores

- a) nm  
no tiene 32/64 bits
- b) gcc  
sería -m32 y -m64
- c) ld  
sería -melf\_i386 y -melf\_xx86-64
- ✓ d) as

Puntuación: **1,00**

[P2Tutorial]  
[E16SepPra02]

## 13 [P4T]

Una de las “bombas” utiliza el siguiente código para cifrar la clave numérica introducida por el usuario y ahora almacenada en eax:

Elección  
única  
804870d: xor \$0xffff,%eax  
8048712: mov \$0x2,%ecx  
8048717: cltd  
8048718: idiv %ecx  
804871a: cmp %eax,0x804a034

Si el entero almacenado a partir de 0x804a034 es 0x7ff, la clave numérica puede ser:

Usuario Profesores

- a) 0x1009 4F97 (269 045 655)
- b) 0xffff (4095)
- X c) 0x7ff (2047)

1

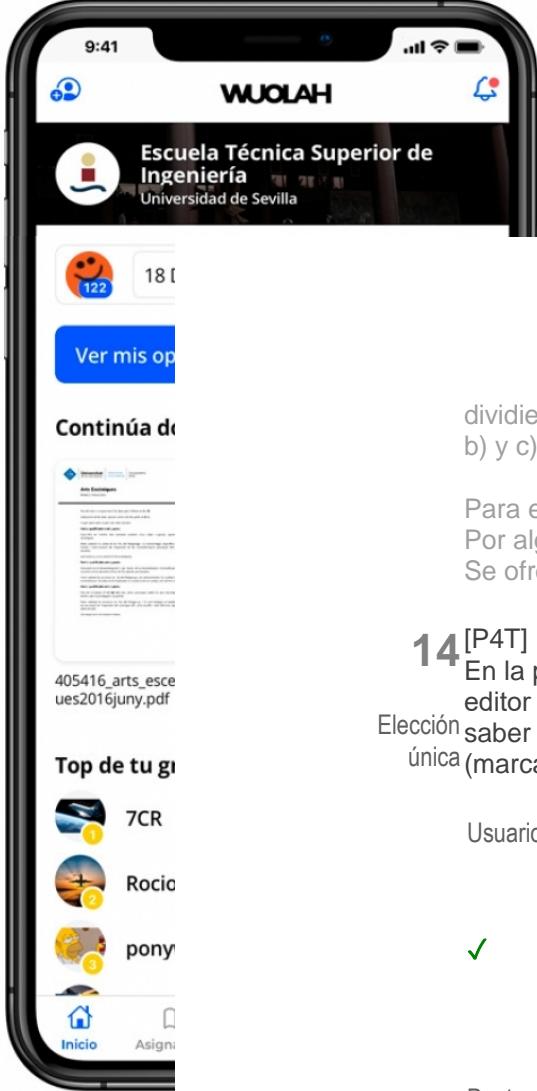
Tanto 0 como 1 servirían de contraseña.

- d) El enunciado de la bomba prohibía explícitamente casos como éste.  
La bomba debe ser un valor fijo. Uno, no dos.

Puntuación: **-0,33**

[P4Tutorial]  
[E17FebPra18]

El código invierte los últimos 12 bits de la clave numérica introducida por el usuario, pierde entonces el LSB (bit menos significativo)



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

dividiendo por 2, y lo que quede debe coincidir con 0x7ff (las opciones b) y c) quedan automáticamente descartadas).

Para ello el número original debería haber sido 0 ó 1.

Por algún motivo, es fácil confundirse razonando.

Se ofrecen opciones que eviten confundirse con 0xffff f000 y similares.

## 14 [P4T]

En la práctica de la bomba, el tercer ejercicio consistía en usar un editor hexadecimal para crear un ejecutable sin "explosiones". Para saber qué contenidos del fichero hay que modificar, se puede utilizar... única (marcar la opción \*falsa\*)

Usuario Profesores

- a) objdump  
hexedit
- b) el enunciado implícitamente indica que no basta con hexedit. Aún así hubo quien la dejó en blanco (pocos) y quien falló (50%)
- c) gdb
- d) ddd

Puntuación: 1,00

[P4Tutorial]

[E17JulPra17]

## 15 [P3.2]

La práctica "parity" debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la siguiente versión de parity6:

```
int parity6(unsigned * array, int len)
{
 int i, result = 0;
 unsigned x;
 for (i=0; i<len; i++){
 x = array[i];
 asm("mov %[x], %%edx \n\t"
 "shr $16, %%edx \n\t"
 "shr $8, %%edx \n\t"
 "xor %%edx,%%edx \n\t"
 "setp %%dl \n\t"
 "movzx %%dl,%[x] \n\t"
 : [x] "+r" (x)
 :
 : "edx"
);
 result += x;
 }
 return result;
}
```

}

Esta función parity6:

Usuario Profesores

- a) produce el resultado correcto  
no es correcta; fallaría por ejemplo con  
array={0,1,2,3}  
Caso real, entregado en prácticas. Las tres primeras instrucciones asm se pierden al poner edx
- b) a 0 usando xor. Consecuentemente, se activa PF para ajustar a impar, y termina siendo x=1. Es decir, todos los elementos del array contabilizan paridad=1. El array {1,2,4,8} pasa desapercibido, pero {0,1,2,3} debería producir resultado=2<>4.
- c) no es correcta; fallaría por ejemplo con  
array={1,2,4,8}
- d) no es correcta pero el error no se manifiesta en los ejemplos propuestos, o se manifiesta en ambos

Puntuación: **0,00**

[P3.2Parity]  
[E13SepPra08]  
[E14FebPra14]  
[E15FebPra15]  
[E16FebPra10]

## 16 [P2T]

Como parte del proceso de compilación de una aplicación en lenguaje C, enlazar .o → .exe (de objeto proveniente de fuente C a ejecutable)  
Elección usando sólo as y ld, sin gcc...  
única

Usuario Profesores

- a) Ninguna de las anteriores respuestas es correcta
- b) Se puede, repartiendo entre as y ld los modificadores (switches) que corresponda
- c) Basta usar ld, con los modificadores de gcc que corresponda, y añadiéndole el runtime de C
- X d) Se puede, repartiendo modificadores entre as y ld, y añadiendo al comando ld el runtime de C

Puntuación: **-0,33**

[P2Tutorial]  
[E13SepPra06]

## 17 [P3T]

Suponga la siguiente sentencia asm en un programa:

```
asm(" add (%[a],%[i],4),%r"
 :[r] "+r" (result)
 :[i] "r" (i),
```

Elección [a] "r" (array) );  
única ¿Cuál de las siguientes afirmaciones es correcta?

Usuario Profesores

- a) la salida de la función se fuerza a que esté en la variable result
- b) r es una posición de memoria de entrada/salida
- c) i es un registro de entrada
- d) a es una posición de memoria de entrada

X

Puntuación: -0,33

[P3Tutorial]  
[E13FebPra10]

18 [P5.1]

En la práctica de la cache, el código de line.cc incluye la sentencia

Elección for (unsigned long long line=1;  
única       line<=LINE; line<<=1) { ... }

¿Qué objetivo tiene la expresión line<<=1?

Usuario Profesores

- a) sacar un uno (1) por el stream line
- b) salir del bucle si el tamaño de línea se volviera menor o igual que 1 para algún elemento del vector
- c) volver al principio del vector cuando el índice exceda la longitud del vector
- d) duplicar el tamaño del salto en los accesos al vector respecto a la iteración anterior

✓       • Puntuación: 1,00

[P5.1Line]  
[E14FebPra19]

19 [P3T]

¿Cuál de las siguientes afirmaciones es cierta respecto al lenguaje C?

Elección      Usuario Profesores  
única

- Pasar a una función un puntero a una variable se
- a) traduce en introducir en la pila el valor de la variable
  - b) En lenguaje C, al llamar a una subrutina o función se introducen los parámetros en la pila y después se realiza una llamada a la subrutina
  - c) Antes de volver de la rutina llamada, el programa en C se encarga de quitar de la pila los parámetros de llamada realizando varios pop

✓       •

- Los parámetros se introducen en la pila en el orden  
d) en el que aparecen en la llamada de C, es decir,  
empezando por el primero y acabando por el último

Puntuación: **1,00**

[P2Tutorial]

[P3Tutorial]

[E13SepPra16]

**20** [P2.1]

En la práctica "media" se pide sumar una lista de 32 enteros SIN signo de 32 bits en una plataforma de 32 bits sin perder precisión, esto es, evitando perder acarreos. De entre los siguientes, ¿cuál es el mínimo único valor entero que repetido en toda la lista causaría acarreo con 32 bits (sin signo)? Se usa notación decimal y espacios como separadores de millares/millones/etc.

Usuario Profesores

- a) 10 000 000
- b) 100 000 000  
No llega, 100 millones < 128M
- ✓ c) 1 000 000 000  
Se pasa, 1000 millones >> 128M
- d) 10 000 000 000

Puntuación: **1,00**

[P2.1SumUns]

[E17FebPra08]

32bit sin signo llega a  $2^{32}-1 = 4G-1 \approx 4\ 000\ 000\ 000$ .

Habría acarreo si llegáramos a 4G. Con 32 elementos iguales, tendrían que valer  $4G/32 = 1G/8 = 128M$ .

Puntuación: **12,33 (6,17 sobre 10)**

---

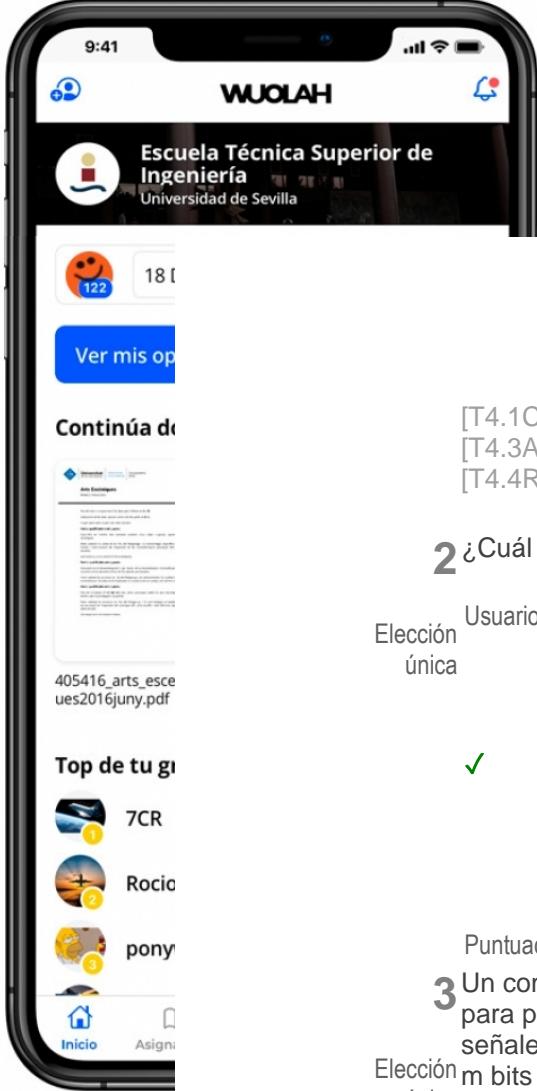
**1** [T4.4]

La segmentación de cauce...

Elección única  
Usuario Profesores

- a) permite ejecutar varias instrucciones concurrentemente
- b) acelera la ejecución de un programa
- c) provoca riesgos debido a datos
- ✓ d) todas las respuestas son ciertas

Puntuación: **1,00**



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

Continúa d

[T4.1ConSeg]  
[T4.3Aceler]  
[T4.4Riesgs]

2 ¿Cuál de las siguientes afirmaciones es incorrecta?

Elección única  
Usuario Profesores

✓ •

- El repertorio de instrucciones es el conjunto de
- a) operaciones que es capaz de interpretar la unidad de control.
  - b) Los operandos siempre están almacenados en memoria.
  - c) El repertorio de instrucciones debe ser capaz de realizar una tarea en un tiempo finito.
  - d) El modo de direccionamiento permite determinar un operando o la ubicación del operando.

Puntuación: 1,00

3 Un computador usa el formato vertical de codificación de instrucciones para parte de las señales de control y el formato horizontal para k señales de control. El formato vertical posee n campos codificados de m bits cada uno. ¿Cuál es el máximo número de señales de control únicas que pueden usarse en este computador?

Elección única  
Usuario Profesores

X

- a)  $k + n \cdot 2^m$
- b)  $k + n^m$
- c)  $k + n \cdot (2^m - 1)$
- d) Ninguno de los anteriores

Puntuación: -0,33

4 [T2.1.2]

La instrucción leave hace exactamente lo mismo que

Elección única  
Usuario Profesores

✓ •

- a) mov %esp, %ebp; pop %esp
- b) pop %eip
- c) ret
- d) mov %ebp, %esp; pop %ebp

Puntuación: 1,00

[T2.1.2Lngjes]  
[T2.3.1Marcop]  
[E15FebPra11]

5 [T2.4.3]

Al traducir la sentencia C r->i = val; gcc genera el código ASM movl %edx, 12(%eax). Se deduce que

Elección Usuario Profesores  
única



- i es un entero que vale 12
- a) después de la asignación, i vale lo que había en %edx (que es val)
  - b) el desplazamiento de i en \*r es 12 y en concreto r está en %eax
  - c) val es un entero que vale 12 val está en %edx
  - r es un puntero que apunta a la posición de memoria 12
  - d) r es puntero a struct y seguramente valdrá 0x0804XXXX en ejecutables x86

Puntuación: **1,00**

[T2.4.3Struct]

[E15SepPra13]

**6** [T6.3]

Un sistema basado en un microprocesador con un bus de datos de n bits y un bus de direcciones de 16 bits direcciona la memoria por palabras de n bits y dispone de una memoria SRAM formada por dos módulos de 16 K x n cada uno. ¿Qué porcentaje del mapa de memoria está ocupado por la SRAM?

Usuario Profesores



- 50%
- a) 16bits => 64Kpal  
 $16+16 \text{ Kpal} = 32\text{Kpal} = 50\% \text{ 64Kpal}$
- b) 25%
- c) 100%
- d) 12,5%

Puntuación: **-0,33**

[T1.3EstBus]

[T6.3Diseño]

[E17FebTeo27]

**7** [T5.1]

La conexión entre un dispositivo de E/S y el procesador mediante bus:

Elección Usuario Profesores  
única



- a) Requiere mucha circuitería
- b) Permite conectar en paralelo varios dispositivos
- c) Requiere multiplexores y demultiplexores para las señales de datos
- d) Es difícil de expandir

Puntuación: **-0,33**

[T1.3EstBus]  
[T5.1FunE/S]  
[E17JulTeo19]

**8** Una memoria que está estructurada en palabras de 8 bits tiene una capacidad de 32 Kbits. ¿Cuántas líneas de dirección tiene dicha memoria?

Elección  
única Usuario Profesores

- ✓ • a) 12  
b) 8  
c) 4  
d) 32

Puntuación: **1,00**

**9** El espacio direccionable de memoria de un computador depende del diseño del:

Elección  
única Usuario Profesores

- a) Bus de direcciones  
b) Bus de datos  
c) Ninguna de las otras es correcta  
X d) a) y b) son correctas

Puntuación: **-0,33**

**10** [T6.2]

¿Cuál de las siguientes afirmaciones es falsa?

Elección  
única Usuario Profesores

- ✓ • La lectura de un bit de la matriz de almacenamiento de una memoria DRAM proporciona una señal mucho más débil que la suministrada por los inversores de una celda de memoria SRAM.  
a) Una celda DRAM sólo necesita un transistor y un condensador.  
c) Las memorias DRAM son en general mucho más rápidas que las SRAM  
Las memorias DRAM presentan generalmente una d) capacidad de almacenamiento mucho mayor que las SRAM.

Puntuación: **1,00**

[T6.2RAMROM]

**11** La operación aritmética calculada por el programa

Elección  
única

```
mov r1,#5
 mov r2,#3
 mov r3,#7
 mov r4,#8
 mul r5,r2,r3
```

add r6,r1,r5  
sub r7,r6,r4

es:

Usuario Profesores

- ✓ • a)  $8 - 5 + (3 \times 7)$   
b)  $5 + (3 \times 7) - 8$   
c)  $8 - (3 \times 7) + 5$   
d)  $(3 \times 7) + 8 - 5$

Puntuación: **1,00**

**12** ¿Cuál es la diferencia entre las instrucciones subl y cmpl?

Elección única Usuario Profesores

- ✓ • a) subl tiene en cuenta el acarreo de entrada y cmpl no  
b) subl no afecta a los indicadores de estado y cmpl sí  
c) subl almacena el resultado sobre escribiendo uno de los operandos y cmpl no  
d) subl realiza una resta y cmpl realiza una suma

Puntuación: **1,00**

**13** [T1.1]

Son funciones de la unidad de control:

Elección única Usuario Profesores

- X • a) la codificación de las instrucciones máquina  
b) la lectura de memoria principal de la instrucción apuntada por el µPC  
c) el secuenciamiento de las instrucciones máquina  
d) todas las respuestas son ciertas

Puntuación: **-0,33**

[T1.1UniFun]

[T3.1CamDat]

[E17JulTeo07]

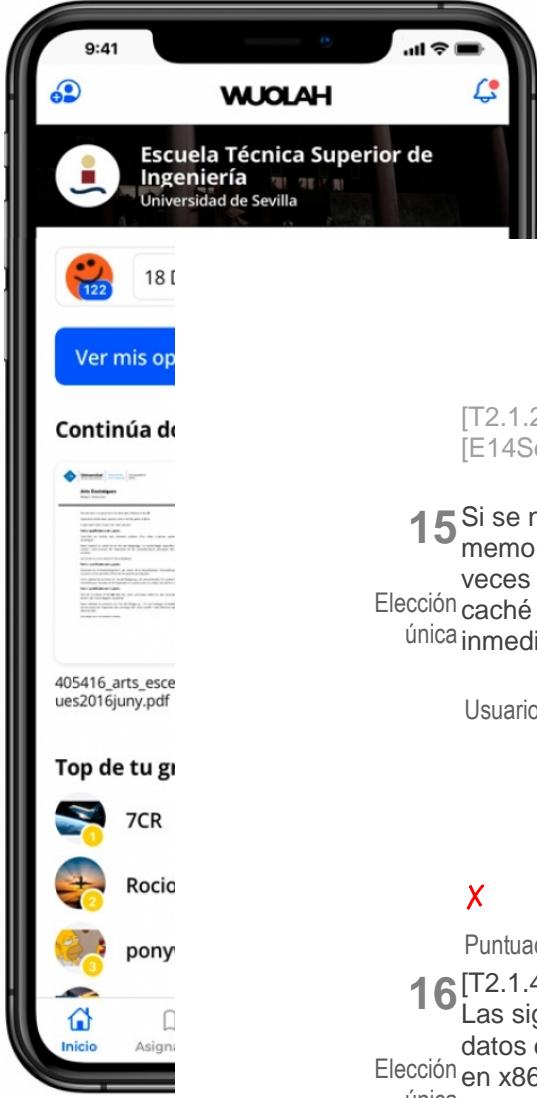
**14** [T2.1.2]

Un programa de ordenador que convierte un programa fuente de alto nivel completo en lenguaje máquina se llama un:

Elección única Usuario Profesores

- ✓ • a) compilador  
b) simulador  
c) ensamblador  
d) intérprete

Puntuación: **1,00**



**Descarga la APP de Wuolah.**  
Ya disponible para el móvil y la tablet.

Ya disponible para el móvil y la tablet.



A circular orange notification icon with a smiling face, containing the number 122, indicating 181 new messages.

[Ver mis op](#)

Continúa de

405416\_arts\_esce  
ues2016juniy.pdf

-  Top de tu grupo
-  7CR
-  Rocio
-  pony

---

-  Inicio
-  Asign.

[T2.1.2Lngjes]  
[E14SepTeo14]

**15** Si se necesitan 60 ns para escribir una palabra de datos de caché en memoria principal y cada bloque de caché tiene 8 palabras, ¿cuántas veces seguidas se tiene que escribir en un mismo bloque para que una Elección caché de postescritura sea más eficiente que una de escritura única inmediata?

## Usuario Profesores

- - a) Más de 8 veces.
    - b) Depende de la tasa de aciertos.
    - c) La caché de postescritura no puede ser más eficiente que la de escritura inmediata.
    - d) La caché de postescritura siempre será más eficiente que la de escritura inmediata.

Puntuación: -0,33

**16** [T2.1.4] Las siguientes afirmaciones sugieren que el tamaño de varios tipos de datos en C (usando el compilador gcc) son iguales tanto en IA32 como en x86-64. Sólo una de ellas es FALSA. ¿Cuál?

## Usuario Profesores

- ✓ • a) El tamaño de un double es 8 bytes  
b) El tamaño de un puntero es 4 bytes  
c) El tamaño de un short es 2 bytes  
d) El tamaño de un int es 4 bytes

Puntuación: 1,00

[T2.1.4x86-64]  
[E12SepTeo08]

**17** ¿En qué tipo de memoria virtual es un problema la fragmentación externa?

## Elección única Usuario Profesores

- - a) Memoria paginada
    - b) Memoria segmentada
    - c) Memoria con segmentación paginada
    - d) En ninguno de ellos

Puntuación: 0,00

18 [T4.4]  
B

- Respecto al salto retardado y al salto anulante, ¿cuál permite que se ejecute la siguiente instrucción, y cuál no?

Ejecute la siguiente  
Elección  
única Usuario Profesores

- ✓ •
- a) el retardado la ejecuta sólo si no se cumple la condición de salto, el anulante no la ejecuta nunca
  - b) el retardado la ejecuta sólo si se cumple la condición de salto, el anulante sólo si no se cumple
  - c) el retardado la ejecuta siempre, el anulante la ejecuta sólo si se cumple la condición de salto
  - d) el retardado ejecuta la siguiente instrucción (con el correspondiente retraso), el anulante no la ejecuta (de hecho la anula)

Puntuación: 1,00

[T4.4Riesgs]  
[E15SepTeo19]

## 19 ¿Cuál de las siguientes funciones no corresponde a un módulo de E/S?

Elección única Usuario Profesores

✓ •

- a) Almacenamiento de programas
- b) Comunicación con el dispositivo
- c) Almacenamiento temporal de datos
- d) Comunicación con el microprocesador

Puntuación: 1,00

## 20 [T2.3.1] Sobre el direccionamiento relativo a contador de programa:

Elección única Usuario Profesores

✓ •

- a) Favorece la implementación de código reubicable.
- b) Su uso en los saltos y llamadas a subrutinas reduce el tamaño de la instrucción.
- c) Es adecuado para alcanzar instrucciones próximas a la que se está ejecutando.
- d) Todas las respuestas son ciertas.

Puntuación: 1,00

[T2.2.4SalCon]  
[T2.3.1MarcoP]

## 21 [T2.1.2] En IA32 la pila es:

Elección única Usuario Profesores

X

•

- a) una dirección de memoria de 32 bits almacenada en el contador de programa
- b) un registro de 32 bits en el microprocesador
- c) un conjunto de posiciones de memoria usadas para almacenar información temporal durante la ejecución del programa

d) un registro de 16 bits en el microprocesador

Puntuación: **-0,33**

[T2.1.2Lngjes]

[T2.3.1MarcoP]

[E14SepTeo15]

[E16FebTeo19]

**22** [T2.4.2]

En el siguiente código, ¿qué reordenamiento de los bucles muestra mejor localidad?

Elección

única // X, Y, Z ctes #define previo  
int a[X][Y][Z]  
int i, j, k, sum = 0;  
for (i = 0; i < Y; i++)  
 for (j = 0; j < Z; j++)  
 for (k = 0; k < X; k++)  
 sum += a[k][i][j];

Usuario Profesores

- ✓ • a) k externo, i central, j interno  
b) j externo, k central, i interno  
c) El orden de los bucles no afecta a la localidad  
d) i externo, j central, k interno (el orden en que están ahora)

Puntuación: **1,00**

[T2.4.2Arrays]

[T6.1ConLoc]

[E14FebTeo04]

**23** [T5.1]

Respecto a si un computador dispone de E/S independiente (separada) o usa E/S mapeada a memoria:

Elección

única Usuario Profesores

- Si el encapsulado del procesador no dispone de  
a) patilla IO/M# (ni equivalentes), el computador sólo dispone de E/S separada  
Si el repertorio del procesador tiene instrucciones  
b) del tipo IN y OUT, es que el computador dispone de E/S separada  
Si el repertorio del procesador tiene instrucciones  
c) del tipo LOAD y STORE, el computador sólo dispone de E/S mapeada a memoria  
Si el encapsulado (chip) del procesador tiene patilla  
d) (pin) IO/M# (o patillas equivalentes), eso evidencia que el computador usa E/S mapeada a memoria

Puntuación: **1,00**

[T5.1FunE/S]  
[E15FebTeo16]

## 24 ¿Cuántos puertos de E/S permite manejar la interfaz de periféricos programable 8255?

Elección  
única

X

•

- a) 3 puertos de 12 bits
- b) 2 puertos de 16 bits y 1 puerto de 8 bits
- c) 2 puertos de 8 bits y 2 puertos de 4 bits
- d) Todas las respuestas anteriores son falsas

Puntuación: -0,33

## 25 [T2.2.1]

¿Cuál de las siguientes instrucciones x86 se puede usar para sumar dos registros y guardar el resultado sin sobrescribir ninguno de los registros originales?

Elección  
única

Usuario Profesores

✓

•

- a) mov
- b) lea
- c) add
- d) Ninguna de ellas

Puntuación: 1,00

[T2.1.3ConASM]

[T2.2.1ModDir]

[T2.2.2OpArit]

[E12FebTeo06]

## 26 Al diseñar el formato de instrucción:

Elección  
única

Usuario Profesores

•

- a) el número de formatos de instrucción diferentes no afecta a la complejidad de la UC.
- b) se suele omitir el campo que indica la siguiente instrucción (la siguiente a ejecutar es la siguiente en memoria, salvo en caso de salto).
- c) sólo hay que saber el tipo de operación a realizar y los operandos necesarios.
- d) hay que indicar explícitamente todos los operandos y destinos.

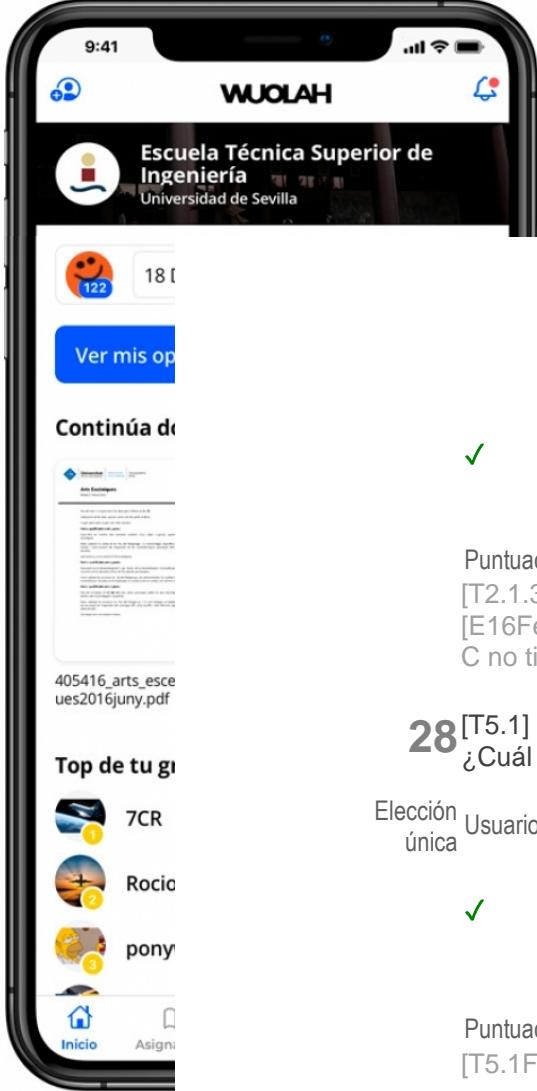
Puntuación: 0,00

## 27 [T2.1.3]

¿Cuál de los siguientes lenguajes no permite el paso de parámetros por referencia?

Elección  
única

Usuario Profesores



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the  
App Store

GET IT ON  
Google Play

- a) C++
- b) C
- c) FORTRAN
- d) Pascal

Puntuación: 1,00

[T2.1.3ConASM]  
[E16FebTeo13]

C no tiene "referencias", por eso se pasan punteros a la función swap()

**28** [T5.1]

¿Cuál de las siguientes afirmaciones es cierta?

Elección  
única

- a) La E/S en memoria emplea la patilla IO/M#
- b) En E/S independiente, las instrucciones de acceso a memoria suelen ser más largas que las de E/S
- c) La E/S en memoria facilita la protección
- d) Ninguna de las anteriores es cierta

Puntuación: 1,00

[T5.1FunE/S]

**29** Cada celda de un chip de memoria DRAM de 1M x 1, organizada en una matriz de 512 filas x 2048 columnas, necesita ser refrescada cada 16 ms. ¿Cada cuánto tiempo ha de realizarse una operación de refresco en el chip?

Elección  
única

Usuario Profesores

X

- a) 8192 milisegundos
- b) 32,768 segundos
- c) 7,8125 microsegundos
- d) 31,25 microsegundos

Puntuación: -0,33

**30** El resultado de desplazar aritméticamente dos posiciones hacia la derecha el número de 8 bits en complemento a dos –32 es:

Elección  
única

Usuario Profesores

.

- a) 56
- b) -128
- c) -8
- d) Ninguno de los resultados anteriores es correcto

Puntuación: 0,00

## LAS PREGUNTAS MÁS INTERASANTES

**11** [P2.1]

En la práctica "media" se pide sumar una lista de 32 enteros SIN signo de 32 bits en una plataforma de 32 bits sin perder precisión, esto es, evitando perder acarreos. De entre los siguientes, ¿cuál es el mínimo único valor entero que repetido en toda la lista causaría acarreo con 32 bits (sin signo)? Se usa notación decimal y espacios como separadores de millares/millones/etc.

Usuario Profesores

X

- a) 10 000 000
- b) 100 000 000  
No llega, 100 millones < 128M
- c) 1 000 000 000  
Se pasa, 1000 millones >> 128M
- d) 10 000 000 000

Puntuación: **-0,33**

[P2.1SumUns]

[E17FebPra08]

32bit sin signo llega a  $2^{32}-1 = 4G-1 \approx 4\ 000\ 000\ 000$ .

Habría acarreo si llegáramos a 4G. Con 32 elementos iguales, tendrían que valer  $4G/32 = 1G/8 = 128M$ .

**24** [T2.3.1]

En la secuencia de programa siguiente:

Elección única  
804854e:e8 3d 06 00 00 call 8048b90 <main>  
8048553:50 pushl %eax

¿cuál es el valor que introduce en la pila la instrucción call?

Usuario Profesores

- a) 0x8048b90
- b) 0x804854f
- c) 0x8048553
- d) 0x804854e

Puntuación: **0,00**

[T2.3.1MarcoP]

[E16FebTeo14]

**2** [P2.2]

En la práctica "media" se pide sumar una lista de enteros \*con\* signo de 32 bits en una plataforma de 32 bits sin perder precisión, esto es, evitando overflow. ¿Cuál es el menor valor positivo que repetido en los \*dos\* primeros elementos de la

## Elección lista causaría overflow con 32 bits al realizar la suma de \*esos dos\* primeros elementos de la lista?

Usuario Profesores

- a) 0x0400 0000  
suma 0x0800 0000 sin problema  
0x8000 0000
- b) es negativo, el enunciado pide menor valor positivo
- c) 0x0800 0000  
suma 0x1000 0000 sin problema
- d) 0x4000 0000  
0x8000 0000 sería negativo en 32bit

✓ • Puntuación: 1,00

[P2.2SumSgn]  
[E17JulPra09]

3 [P2T]

El switch de gcc para que únicamente compile de lenguaje C a ensamblador, y no realice ningún paso adicional (ensamblar, enlazar, etc), es...

## Elección

### única

Usuario Profesores

- a) -g  
para incorporar info depuración
- b) -c  
de C/asm a objeto, .c/.s→.o
- c) -o  
para nombrar el ejecutable
- d) -S

✓ • Puntuación: 1,00

[P2Tutorial]  
[E16SepPra01]

5 [P2T]

La siguiente línea en la sección de datos de un programa en ensamblador de IA32  
result: .int 0,0

Usuario Profesores

- a) Reserva espacio para un único entero, inicializado a 0,0  
Reserva espacio para un único entero, b) inicializado a 0, en la posición de memoria 0

✓ •

- Reserva espacio para un entero,  
c) inicializado a 0, seguido de un dato de  
tamaño indefinido, también inicializado a 0  
d) Reserva espacio para dos enteros,  
inicializados ambos a 0

Puntuación: **1,00**

[P2Tutorial]  
[E17FebPra04]

**7** [P4T]

## Elección única

En la práctica de la bomba necesitamos estudiar el código máquina de la bomba del compañero. A veces dicho código no se visualiza directamente en el depurador ddd, y algunas de las técnicas que se pueden probar para conseguir visualizarlo son... (marcar la opción \*falsa\*)

Usuario Profesores

X

•

- a) comprobar que está activado el panel View  
→ Machine Code Window  
b) asegurarse de que se ha escrito  
correctamente el nombre del ejecutable  
c) escribir info line main en el panel de línea  
de comandos gdb  
recompilar con información de depuración,  
por si se nos había olvidado, ya que sin -g  
el ejecutable no contiene información de  
depuración  
sí, claro, como que nos van a pasar el  
código fuente... :-)

Puntuación: **-0,33**

[P4Tutorial]  
[E17JulPra14]

**9** [P2.2]

## Elección única

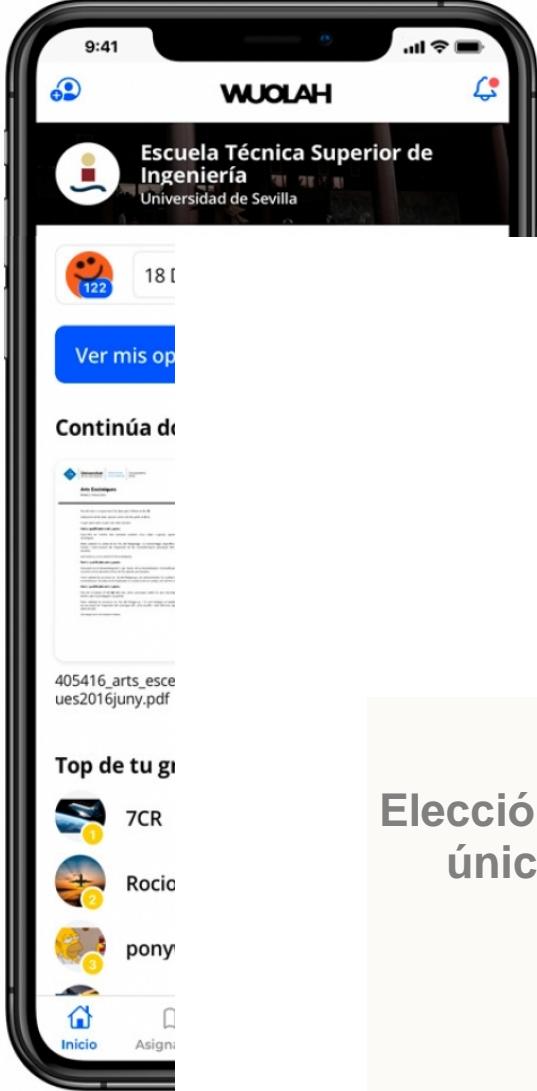
En la práctica “media” se pide sumar una lista de 32 enteros \*con\* signo de 32bits en una plataforma de 32bits sin perder precisión, esto es, evitando overflow. ¿Cuál es el mayor valor negativo (menor en valor absoluto) que repetido en toda la lista causaría overflow con 32bits?

Usuario Profesores

✓ •

0xf000 0000

- a) se pierde el signo con <<4, mucho más  
con <<5  
b) 0xffff ffff  
eso es -1, y -1x32 == -32 sin problemas  
c) 0xfbff ffff



# Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the App Store GET IT ON Google Play

en cuanto sea algo más grande que  
0xfc00 0000 sale overflow  
0xfc00 0000

- d)  $nx32 == n << 5$  y quedaría 0x8000 0000 !!!  
justo para que no haya overflow

Puntuación: 1,00

[P2.2SumSgn]  
[E15SepPra16]

**6** [T6.3]

## Elección única

Un sistema basado en un microprocesador con un bus de datos de  $n$  bits y un bus de direcciones de 16 bits direcciona la memoria por palabras de  $n$  bits y dispone de una memoria SRAM formada por dos módulos de 16 K x  $n$  cada uno. ¿Qué porcentaje del mapa de memoria está ocupado por la SRAM?

Usuario Profesores

- a) 50%  
 $16\text{bits} \Rightarrow 64\text{Kpal}$   
 $16+16\text{ Kpal} = 32\text{Kpal} = 50\% 64\text{Kpal}$
- b) 25%
- c) 100%
- d) 12,5%

X

Puntuación: -0,33

[T1.3EstBus]  
[T6.3Diseño]  
[E17FebTeo27]

**9** El espacio direccionable de memoria de un computador depende del diseño del:

## Elección única

Usuario Profesores

- a) Bus de direcciones
- b) Bus de datos
- c) Ninguna de las otras es correcta
- d) a) y b) son correctas

X

Puntuación: -0,33

**15** Si se necesitan 60 ns para escribir una palabra de datos de caché en memoria principal y cada bloque de caché tiene 8 palabras, ¿cuántas veces seguidas se tiene que escribir en un mismo bloque para que una caché de postescritura sea más eficiente que una de escritura inmediata?

## Elección única

Usuario Profesores

|                       |                                                                                                                              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                       |                                                                                                                              | <ul style="list-style-type: none"> <li>• a) Más de 8 veces.</li> <li>• b) Depende de la tasa de aciertos.</li> <li>• c) La caché de postescritura no puede ser más eficiente que la de escritura inmediata.</li> <li>• d) La caché de postescritura siempre será más eficiente que la de escritura inmediata.</li> </ul> <p><span style="color: red;">X</span></p>                                                                                                                                                                                                               |
|                       |                                                                                                                              | Puntuación: <b>-0,33</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>18</b>             | [T4.4]<br>Respecto al salto retardado y al salto anulante, ¿cuál permite que se ejecute la siguiente instrucción, y cuál no? |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Elección única</b> | Usuario Profesores                                                                                                           | <p><span style="color: green;">✓</span></p> <ul style="list-style-type: none"> <li>• a) el retardado la ejecuta sólo si no se cumple la condición de salto, el anulante no la ejecuta nunca</li> <li>• b) el retardado la ejecuta sólo si se cumple la condición de salto, el anulante sólo si no se cumple</li> <li>• c) el retardado la ejecuta siempre, el anulante la ejecuta sólo si se cumple la condición de salto</li> <li>• d) el retardado ejecuta la siguiente instrucción (con el correspondiente retraso), el anulante no la ejecuta (de hecho la anula)</li> </ul> |
|                       |                                                                                                                              | Puntuación: <b>1,00</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|                       | [T4.4Riesgs]<br>[E15SepTeo19]                                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>21</b>             | [T2.1.2]<br>En IA32 la pila es:                                                                                              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Elección única</b> | Usuario Profesores                                                                                                           | <p><span style="color: red;">X</span></p> <ul style="list-style-type: none"> <li>• a) una dirección de memoria de 32 bits almacenada en el contador de programa</li> <li>• b) un registro de 32 bits en el microprocesador</li> <li>• c) un conjunto de posiciones de memoria usadas para almacenar información temporal durante la ejecución del programa</li> </ul>                                                                                                                                                                                                            |

- d) un registro de 16 bits en el microprocesador

Puntuación: **-0,33**

[T2.1.2Lngjes]  
[T2.3.1MarcoP]  
[E14SepTeo15]  
[E16FebTeo19]

**22** [T2.4.2]

En el siguiente código, ¿qué reordenamiento de los bucles muestra mejor localidad?

## Elección única

```
// X, Y, Z ctes #define previo
int a[X][Y][Z]
int i, j, k, sum = 0;
for (i = 0; i < Y; i++)
 for (j = 0; j < Z; j++)
 for (k = 0; k < X; k++)
 sum += a[k][i][j];
```

Usuario Profesores

- ✓ • a) k externo, i central, j interno  
b) j externo, k central, i interno  
c) El orden de los bucles no afecta a la localidad  
d) i externo, j central, k interno (el orden en que están ahora)

Puntuación: **1,00**

[T2.4.2Arrays]  
[T6.1ConLoc]  
[E14FebTeo04]

**26**

Al diseñar el formato de instrucción:

## Elección única

Usuario Profesores

- a) el número de formatos de instrucción diferentes no afecta a la complejidad de la UC.  
• b) se suele omitir el campo que indica la siguiente instrucción (la siguiente a ejecutar es la siguiente en memoria, salvo en caso de salto).  
c) sólo hay que saber el tipo de operación a realizar y los operandos necesarios.

- d) hay que indicar explícitamente todos los operandos y destinos.

Puntuación: **0,00**

**27** [T2.1.3]

¿Cuál de los siguientes lenguajes no permite el paso de parámetros por referencia?

## Elección única

Usuario Profesores

- ✓      •      a) C++
- b) C
- c) FORTRAN
- d) Pascal

Puntuación: **1,00**

[T2.1.3ConASM]

[E16FebTeo13]

C no tiene "referencias", por eso se pasan punteros a la función swap()

# EDEN

El direccionamiento relativo a registro base utiliza...

Usuar Profesor

io es

- a) dos registros.
- b) un registro.
- c) dos desplazamientos contenidos en la propia instrucción.
- ✓ • d) un registro y un desplazamiento.

Puntuación: **1,00**

Una instrucción de salto si menor, para números positivos sin signo, tiene que comprobar el valor de:

Usuar Profesor

io es

- a) los bits de signo y desbordamiento
- b) el bit de signo
- ✗ • c) el bit de cero
- d) el bit de acarreo

Puntuación: **-0,33**

La instrucción setg %al:

U Pro  
su fes  
ari ore  
o s

- a Pone siempre AL a 1.  
)
- ✓ • b Pone AL a 1 en algunos casos.  
)
- c No cambia el contenido de AL  
)
- d Complementa AL si el resultado  
) de la comparación anterior es A >  
B.

Puntuación: **1,00**

¿Cuál de las siguientes instrucciones no modifica necesariamente la secuencia de ejecución del programa?

Usuar Profesor

io es

- a) RET
- b) CALL dir
- c) JNE dir
- d) JMP dir

Puntuación: **1,00**

El lenguaje máquina es...

Usuar Profesor

io es

- a) fácilmente legible por el programador.
- b) portable entre arquitecturas.
- c) difícil de codificar manualmente.
- d) una alternativa razonable al uso del lenguaje ensamblador.

Puntuación: **1,00**

¿En qué generación, dentro de la historia de los computadores digitales, aparece la segmentación de cauce?

Usuar Profesor

io es

- a) primera
- b) segund  
a
- c) tercera
- d) cuarta

Puntuación: **1,00**

[T1.5]

¿En qué generación, dentro de la historia de los computadores digitales, se alcanzaron tiempos de conmutación del orden de nanosegundos?

Usuar Profesor

io es

- a) primera

- b) segunda
- c) tercera
- d) cuarta

Puntuación: **1,00**

En IA32, el registro contador de programa se denomina:

Usuar Profesor

io es

- a) PC
- b) RIP
- c) PC  
R
- d) EIP

Puntuación: **-0,33**

En cdecl/x86, ¿cuál de los siguientes registros tiene que ser guardado por la función llamada si es alterado por ésta?

- a) eax
- b) ecx
- c) ebx
- d) edx

¿Cuál de las siguientes listas está correctamente ordenada temporalmente?

- a) 486, 8086, Pentium MMX, Core 2, Pentium III, Pentium 4
- b) 8086, 486, Pentium MMX, Pentium III, Pentium 4, Core 2
- c) 8086, 486, Pentium III, Pentium MMX, Core 2, Pentium 4
- d) 486, 8086, Core 2, Pentium III, Pentium 4, Pentium MMX

Si el contenido de EBP es 0x13000, ¿a qué dirección se hace referencia con la instrucción INC –0x80(%EBP)?

- ✓ a) 0x13080
- b) 0x13000
- c) 0x80
- d) 0x12F80

¿Cuál es la dirección física del operando fuente de la instrucción ADD AX, ETIQUETA siendo ETIQUETA = 7000h, CS = 1500h, DS = 4500h, e IP = 25h (la instrucción ocupa 3 bytes)?

- ✗ a) 53000h
- b) **4C000h**
- c) 15025h
- d) 5C000h

### 7 [T1.2]

El modo de direccionamiento en el que se especifica un registro y una dirección de memoria cuyo contenido se suma al contenido del registro base para obtener la dirección efectiva, se conoce como:

ón Us Prof  
ún ua eso  
ic río res

- a a base con
  - ) desplazamiento
- b directo o absoluto
  - )
- ✗ c indirecto a registro
  - ) través de memoria
- d ninguno de los
  - ) anteriores

Puntuación: **-0,33**

**8**

[T1.5]

Elección única

Usuar Profesor

io es

- a) primera
- b) segund
- a
- ✓ • c) tercera
- d) cuarta

Puntuación: **1,00**

Sobre el ensamblador:

Usuar Profesor

io es

- a) La calidad de un programa ensamblador afectará menos al tiempo de ejecución de los programas generados por él que la calidad de un compilador.
- b) Las etiquetas permiten que el programador especifique el destino de un salto de forma que éste no tenga que modificarse manualmente cuando el programa varíe de tamaño.
- c) El lenguaje ensamblador elimina la posibilidad de errores en la generación de la representación en lenguaje máquina de cada instrucción.
- ✓ • d) Todas las respuestas son ciertas.

Suponiendo que todos los registros inicialmente contienen el valor 1 y que el destino es el primer registro, ¿cuál es el valor de r1 tras la ejecución de la siguiente secuencia de instrucciones?

```
mov r1, #4
mov r2, #3
add r3, r1, r1
```

sub r1, r3, r2

mul r3, r1, r1

Usuar Profesor

io es

- a) 3
- 6
- b) 6
- ✓ • c) 5
- d) 2
- 0

Puntuación: **1,00**

[T2.2.3]

En los casos concretos indicados para las siguientes instrucciones IA32, ¿cuál no funciona como instrucción de transferencia?

Usuar Profesor

io es

- ✓ • a) cmpl 0x08048040, %eax
- b) movl \$0x15, %eax
- c) pushl %eax
- d) leal variable, %eax

Puntuación: **1,00**

¿Qué medida de prestaciones es la más fiable de todas las posibles?

Usuar Profesor

io es

- a) MIPS
- ✗ b) MIPS equivalentes
- c) ninguna de las otras respuestas es correcta
- d) MFLOPS

Puntuación: **-0,33**

En un microprocesador de 4 bits, una operación en la que el bit 0 de un registro se copia en el acarreo, después el bit 1 se copia en el bit 0, después el bit 2 se copia en el bit 1, y por último el bit 3 se copia en el bit 2, es:

Usuar Profesor

io es

- a) Un desplazamiento lógico a la derecha.
- b) Un desplazamiento aritmético a la derecha.
- c) Una rotación a la derecha.
- d) Una rotación a la derecha a través de acarreo.

Puntuación: **-0,33**

El ancho de palabra de una memoria corresponde a:

Usuar Profesor

io es

- a) La longitud del registro de datos de la memoria.
- b) El número de posiciones que la componen.
- c) El número que identifica únicamente cada posición de la memoria.
- d) La longitud del registro de direcciones de la memoria.

Puntuación: **-0,33**

Una CPU con bus de direcciones de 16 bits y bus de datos de 8 bits tiene un registro de 8 bits conectado al bus de datos y a la unidad de control. Puede tratarse del registro

Usuar Profesor

io es

- a) De direcciones
- b) Puntero de pila
- c) Contador de programa
- d) De instrucción

Puntuación: **1,00**

El objetivo de un diseño CISC es...

Usuar Profesor

io es

- a) disminuir el tamaño medio de instrucción.
- b) disminuir la frecuencia de reloj.
- ✓ • c) disminuir el número de instrucciones a ejecutar por un programa.
- d) disminuir el número medio de ciclos por instrucción.

Puntuación: **1,00**

En la captación de la instrucción:

Usuar Profesor

io es

- a) en MBR indicamos la dirección donde está la instrucción y en la ALU recogemos la instrucción.
- b) en MAR indicamos la dirección donde está la instrucción y en la ALU recogemos la instrucción.
- ✓ • c) en MAR indicamos la dirección donde está la instrucción y en MBR recogemos la instrucción.
- d) en MBR indicamos la dirección donde está la instrucción y en MAR recogemos la instrucción.

Puntuación: **1,00**

¿En qué generación, dentro de la historia de los computadores digitales, aparecen las memorias de semiconductores?

Usuar Profesor

io es

- a) Tercera generación.
- b) Cuarta generación.
- ✗ c) Segunda generación.
- d) Quinta generación.

Puntuación: **-0,33**

En el 8086, la dirección efectiva de la cabecera de pila vendrá dada por:

Usuar Profesor

io es

- a) BP

- b) SS \* 10h +  
SP
- X** c) SP
- d) SS \* 10h +  
BP

Puntuación: **-0,33**

Cuando un operando se encuentra almacenado en un registro, se trata de un direccionamiento...

Usuar Profesor

io es

- ✓**
- a) directo a registro
  - b) a registro inmediato
  - c) a registro literal
  - d) ninguno de los anteriores

Puntuación: **1,00**

Cuando un operando se encuentra almacenado en un registro, se trata de un direccionamiento...

Usuar Profesor

io es

- ✓**
- a) directo a registro
  - b) a registro inmediato
  - c) a registro literal
  - d) ninguno de los anteriores

Puntuación: **1,00**

Si AX = 0xFA50 y ejecutamos AND \$0xFF, %AX

Usuar Profesor

io es

- X**
- a) El registro AH se pone a FF
  - b) El registro AH se pone a 0
  - c) El registro AL se pone a FF
  - d) El registro AL se pone a 0

Puntuación: **-0,33**

En una máquina con 32 registros direccionables e instrucciones de 16 bits:

Usuar Profesor

io es

**X**

- a) no se pueden codificar a la vez 64 instrucciones de 0 direcciones, 63 instrucciones de dos registros y 16 instrucciones de 1 registro.
- b) no se pueden codificar a la vez 63 instrucciones de dos registros, una instrucción de 0 direcciones y 32 instrucciones de 1 registro.
- c) no se pueden codificar a la vez 64 instrucciones de 1 registro, 32 instrucciones de dos registros y 32 instrucciones de 0 direcciones.
- d) no se pueden codificar a la vez 62 instrucciones de dos registros, 32 instrucciones de 1 registro y 64 instrucciones de 0 direcciones.

Puntuación: **-0,33**

**7**

¿Qué tipos de instrucciones se emplean más en una arquitectura de registros de propósito general?

Elecció  
n única

Usua  
rio Profes  
ores

**X**

- a) De desplazamiento y rotación.
- b) Aritmético-lógicas.
- c) De transferencia de datos.
- d) De transferencia de control.

Puntuación: **-0,33**

8

Elecció n únic a Si d es un desplazamiento, r un registro índice e i una constante apropiada, el modo de direccionamiento indexado con postautodecremento realiza...

Usua rio  
n  
únic a  
Profes ores



- a) dirección efectiva =  $r + d$ ;  $r = r - i$
- b) dirección efectiva =  $r - i$ ;  $r = r - d$
- c)  $r = r - i$ ; dirección efectiva =  $r - d$
- d)  $r = r + i$ ; dirección efectiva =  $r - d$

Puntuación: 1,00

En el direccionamiento inmediato:

Usuar Profesor

io es



- a) el tamaño (rango de valores) de la constante está limitado.
- b) el operando es una constante de tamaño arbitrario contenida en la misma instrucción.
- c) el código de operación indica el operando afectado.
- d) una vez captada y decodificada la instrucción, se accede a memoria para obtener el valor del operando.

Puntuación: 1,00

En el RISC-I, una ventana de registros contiene:

Usuar Profesor

io es



- a) registros para recibir parámetros del procedimiento llamador.
- b) registros para almacenar variables locales.
- c) registros para enviar parámetros a procedimientos.
- d) todas las respuestas son ciertas.

Puntuación: -0,33

Si queremos almacenar la palabra de 16 bits 0x9660 en una memoria de bytes según "little-endian", quedará almacenada a partir de la posición 0x1000 como:

Usuar Profesor

io es

- a) M[0x1000]=0x06 y  
M[0x1001]=0x69
- b) M[0x1000]=0x69 y  
M[0x1001]=0x06
- ✓ • c) M[0x1000]=0x60 y  
M[0x1001]=0x96
- d) M[0x1000]=0x96 y  
M[0x1001]=0x60

Puntuación: **1,00**

¿En qué generación, dentro de la historia de los computadores digitales, aparece la memoria cache?

Usuar Profesor

io es

- a) primera
- b) segund  
a
- ✓ • c) tercera
- d) cuarta

Puntuación: **1,00**

GCC/Linux IA32 resuelve el ajuste de marco de pila al comenzar un procedimiento mediante las instrucciones:

Usuar Profesor

io es

- a) pushl %ebp movl %esp,  
%ebp
- b) movl %esp, %ebp popl %esp
- ✗ c) pushl %esp movl %ebp,  
%esp
- d) movl %ebp, %esp popl %ebp

Puntuación: **-0,33**

¿En qué orden debería ejecutarse en una máquina de tipo pila la operación aritmética ( $a+b/c-d$ )?

Usuar Profesor

io es

a)  $a + b / c -$

d

b)  $a b + c d -$

/



• c)  $a b c / + d$

-

d)  $a b + / c d$

-

Puntuación: **1,00**

¿Qué arquitectura se caracteriza por presentar una gran variación en la longitud de las instrucciones?

Usuar Profesor

io es

a) registro-registro

b) registro-memoria



• c) memoria-memoria

d) ninguna de las anteriores es  
cierta

Puntuación: **1,00**

¿Cuál de las siguientes parejas de microprocesadores representa mejor los conceptos RISC?

Usuar Profesor

io es



• a) MIPS, SPARC

b) PA-RISC,  
PowerPC

c) Itanium, Alpha

d) Pentium, Athlon

Puntuación: **1,00**

En el 8086, la dirección efectiva de la cabecera de pila vendrá dada por:

Usuar Profesor

io es

- a) SS \* 10h +  
    BP
- b) BP
- c) SS \* 10h +  
    SP
- X d) SP

Puntuación: **-0,33**

**9** [T1.5]

Elecció  
n única

¿En qué generación, dentro de la historia de los computadores digitales, aparece la memoria cache?

Usuar Profesor

io es

- a) primera
- b) segunda
- ✓ c) tercera
- d) cuarta

Puntuación: **1,00**

**2**

[T1.1]

Elección  
única

Usu Profe  
ario sores

- ✓ • a de transferencia de datos con  
    ) memoria
- b aritmético-lógicas  
    )
- c de transferencia de datos  
    ) entre registros

d de desplazamiento y rotación  
)

Puntuación: **1,00**

Una máquina superescalar es aquella que:

Usuar Profesor

io es

- a) basa su funcionamiento en la segmentación software como forma de incrementar el paralelismo.
- b) las instrucciones tienen un campo por cada unidad funcional al realizarse varias operaciones por instrucción.
- ✓ • c) emite simultáneamente múltiples instrucciones por ciclo de reloj, por ejemplo, una entera y otra de coma flotante.
- d) ninguna respuesta de las anteriores es correcta.

Puntuación: **1,00**

¿En qué generación, dentro de la historia de los computadores digitales, aparece la segmentación de cauce?

Usuar Profesor

io es

- a) primera
- b) segund
- a
- ✓ • c) tercera
- d) cuarta

Puntuación: **1,00**

Si el registro EAX contiene X, La secuencia de instrucciones siguiente:

cmpl \$6, %eax

jae Destino

salta a la etiqueta Destino sólo si:

Usuar Profesor

io es

- ✓ • a)  $X < 0 \text{ || } X \geq 6$
- b)  $X \leq 6$
- c)  $X \geq 0 \text{ && } X \leq 6$
- d)  $X > 6$

Puntuación: **1,00**

El lenguaje máquina es...

Usuar Profesor

io es

- ✓ • a) difícil de codificar manualmente.  
b) portable entre arquitecturas.  
c) fácilmente legible por el programador.  
d) una alternativa razonable al uso del lenguaje ensamblador.

Puntuación: **1,00**

Si usamos una estructura de bus con DMA:

Usuar Profesor

io es

- a) al bus del sistema sólo se conecta la CPU y la MP, ya que el DMA se conecta directamente a MP para realizar las transferencias de datos.
- ✗ b) podemos prescindir de controladores de E/S ya que el controlador de DMA se ocupa de controlar las transferencias hacia/desde los periféricos.
- c) la velocidad de este controlador establece la velocidad del bus del sistema.
- d) la CPU puede dejar las transferencias entre MP y periféricos en manos de este controlador (DMA) y seguir ejecutando otras instrucciones.

Puntuación: **-0,33**

Sobre el direccionamiento relativo al contador de programa:

Usuar Profesor

io es

- a) Favorece la implementación de código reubicable.
- b) Su uso en los saltos reduce el tamaño de la instrucción.
- c) Es adecuado para alcanzar instrucciones próximas a la que se está ejecutando.
- ✓ • d) Todas las respuestas son ciertas.

Puntuación: **1,00**

Si %eax contiene x, ¿cuál de las siguientes instrucciones calcula x\*5?

Usuar Profesor

io es

- a) leal 3(%eax,2),%edx
- b) leal 3(%eax,%eax,2),%edx
- c) leal 4(%eax,%eax),%edx
- d) leal (%eax,%eax,4),%edx

Puntuación: **1,00**

Si queremos almacenar la palabra de 16 bits 8965h en memoria según "big-endian", quedará almacenada a partir de la posición 1000h como:

Usuar Profesor

io es

- a) en el byte 1000h se guarda A6h y en el 1001h 91h
- b) en el byte 1000h se guarda 91h y en el 1001h A6h
- c) en el byte 1000h se guarda 89h y en el 1001h 65h
- d) en el byte 1000h se guarda 65h y en el 1001h 89h

Puntuación: **1,00**

¿Cuál es la dirección física del operando fuente de la instrucción ADD AX, ETIQUETA siendo ETIQUETA = 7000h, CS = 1500h, DS = 4500h, e IP = 25h (la instrucción ocupa 3 bytes)?

Usuar Profesor

io es

- a) 15025h
- b) 4C000 h
- c) 5C000 h
- d) 53000h

Puntuación: **1,00**

Sobre el ensamblador:

Usuar Profesor

io es

- a) La calidad de un programa ensamblador afectará menos al tiempo de ejecución de los programas generados por él que la calidad de un compilador.
- b) Las etiquetas permiten que el programador especifique el destino de un salto de forma que éste no tenga que modificarse manualmente cuando el programa varíe de tamaño.
- c) El lenguaje ensamblador elimina la posibilidad de errores en la generación de la representación en lenguaje máquina de cada instrucción.
- ✓ • d) Todas las respuestas son ciertas.

Puntuación: **1,00**

Un procesador emplea codificación en bloque del código de operación. Existen 123 instrucciones que tienen una longitud de 16 bits. ¿A cuántas direcciones de memoria pueden acceder como máximo si todas emplean una estructura “código de operación + dirección de memoria”?

Usuar Profesor

io es

- a) 128.
- ✗ b) 256.
- c) 1024
- .
- d) 512.

Puntuación: **-0,33**

¿Qué arquitectura se caracteriza por presentar una gran variación en la longitud de las instrucciones?

Usuar Profesor

io es

- a) registro-registro
- b) registro-memoria
- ✓ • c) memoria-memoria
- d) ninguna de las anteriores es cierta

Puntuación: **1,00**

Para direccionar una memoria de 1 G palabras de 32 bits cada una, que se direcciona byte a byte, se necesitarán:

Usuar Profesor

io es

**X**

- a) 33  
bits
- b) 21  
bits
- c) 32  
bits
- d) 31  
bits

Puntuación: **-0,33**

[T1.5]

¿En qué generación, dentro de la historia de los computadores digitales, aparece la memoria virtual?

Usuar Profesor

io es

**✓**

- a) primera
- b) segund  
a
- c) tercera
- d) cuarta

Puntuación: **1,00**

En el RISC-I, una ventana de registros contiene, entre otros registros,...

Usuar Profesor

io es

**✓**

- a) registros para recibir parámetros del procedimiento llamador
- b) registros para almacenar matrices de enteros
- c) registros para enviar parámetros a otros procesos
- d) todas las respuestas son falsas

Puntuación: **1,00**

En el arbitraje de un bus...

Usuar Profesor

io es

- a) los dispositivos pasivos pueden requerir el uso del bus para iniciar una transferencia
- b) si hay un único dispositivo pasivo, siempre funciona como esclavo
- c) si hay varios dispositivos activos, siempre funcionan como maestros
- X d) todas las respuestas anteriores son ciertas

Puntuación: **-0,33**

¿Cómo se almacenaría como palabra de 32 bits el número -128 en un sistema que utilice el criterio del extremo menor ("little endian")?

Usuar Profesor

io es

- a) posición 0: FF pos.1:FF pos.2: FF pos.3:  
80
- ✓ • b) 0:80 1:FF 2:FF 3:FF
- c) 0:00 1:01 2: 00 3:80
- d) Ninguna de las anteriores

Puntuación: **1,00**

Si A=FF0Fh y B=0004h, el resultado de desplazar A a la derecha aritméticamente B veces es:

Usuar Profesor

io es

- a FFF0h  
)
- X b 0FF0h  
)
- c F0FFh  
)
- d F0F0h  
)

Puntuación: **-0,33**

Suponiendo que todos los registros inicialmente contienen el valor 0, ¿cuál es el valor de r1 tras la ejecución de la siguiente secuencia de instrucciones?

```
mov r1, #4
mov r2, #3
add r3, r1, r1
sub r1, r3, r2
mul r3, r1, r1
```

Usuar Profesor

io es

- a) 4
- b) 5
- c) 2
- 5
- d) 0

Puntuación: **1,00**

Un modo de vídeo de 1680 x 1050 píxeles y 4 canales de color por píxel (R, G, B, alpha) y 256 niveles de intensidad por canal (0 a 255), ocupa aproximadamente una memoria de:

Usuar Profesor

io es

- a) 1,7 MB
- b) 7 MB
- c) 54 MB
- d) 1,7 GB

Puntuación: **1,00**

Suponiendo que todos los registros inicialmente contienen el valor 0, ¿cuál es el valor de r1 tras la ejecución de la siguiente secuencia de instrucciones?

```
mov r1, #4
mov r2, #3
add r3, r1, r1
sub r1, r3, r2
mul r3, r1, r1
```

Usuar Profesor

io es

- a) 4
- b) 5
- c) 25
- d) 0

Puntuación: **1,00**

Una máquina superescalar es aquella que:

Usuar Profesor

io es

- a) basa su funcionamiento en la segmentación software como forma de incrementar el paralelismo.
- b) las instrucciones tienen un campo por cada unidad funcional al realizarse varias operaciones por instrucción.
- c) emite simultáneamente múltiples instrucciones por ciclo de reloj, por ejemplo, una entera y otra de coma flotante.
- X d) ninguna respuesta de las anteriores es correcta.

Puntuación: **-0,33**

¿Cuál de las siguientes afirmaciones es incorrecta?

Usuar Profesor

io es

- a) En el direccionamiento inmediato el dato se encuentra en la propia instrucción
- b) En el direccionamiento implícito no se indica la ubicación del operando
- c) El direccionamiento indirecto indica la dirección del operando.
- ✓ d) El direccionamiento indexado es útil para manejo de vectores.



PruebaAlien

[www.wuolah.com/student/PruebaAlien](http://www.wuolah.com/student/PruebaAlien)

13803

## Ex-Ene-19-Test-Teo.pdf

examen 2019 enero



2º Estructura de Computadores



Grado en Ingeniería Informática



Escuela Técnica Superior de Ingenierías Informática y de  
Telecomunicación  
Universidad de Granada



**Descarga la APP de Wuolah.**  
Ya disponible para el móvil y la tablet.





**KEEP  
CALM  
AND  
ESTUDIA  
UN POQUITO**

**Nombre:**
**DNI:**
**Grupo:**

### Test de Teoría (3.0p)

Todas las preguntas son de elección simple sobre 4 alternativas.

Cada respuesta vale 0.1p si es correcta, 0p si está en blanco o claramente tachada, -0.03p si es errónea.  
 Anotar las respuestas (a, b, c ó d) en la siguiente tabla.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
|   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |

1. ¿Cuál es el complemento a 2 del número binario 1110 1101 1000?

- a. 0001 0010 0110
- b. 0001 0010 0101
- c. 0001 0010 0111
- d. **0001 0010 1000**

d. 0xfffffffffffffde1

---

5. Para comprobar si el contenido del registro RDX es 0 (y posiblemente saltar a continuación usando la instrucción je), el compilador gcc genera:

- a. cmpq %rdx, %rdx
- b. testq %rdx
- c. **testq %rdx, %rdx**
- d. cmpq %rdx

6. Sabiendo que las instrucciones de salto condicional codifican la dirección de salto con direccionamiento relativo a contador de programa (de 8 o 32 bits con signo), indicar cuál es la dirección de salto de la instrucción je en el siguiente desensamblado, donde se ha tachado precisamente dicha dirección.

```
40042f: 74 f4 je xxxxxxxx
400431: 5d pop %rbp
```

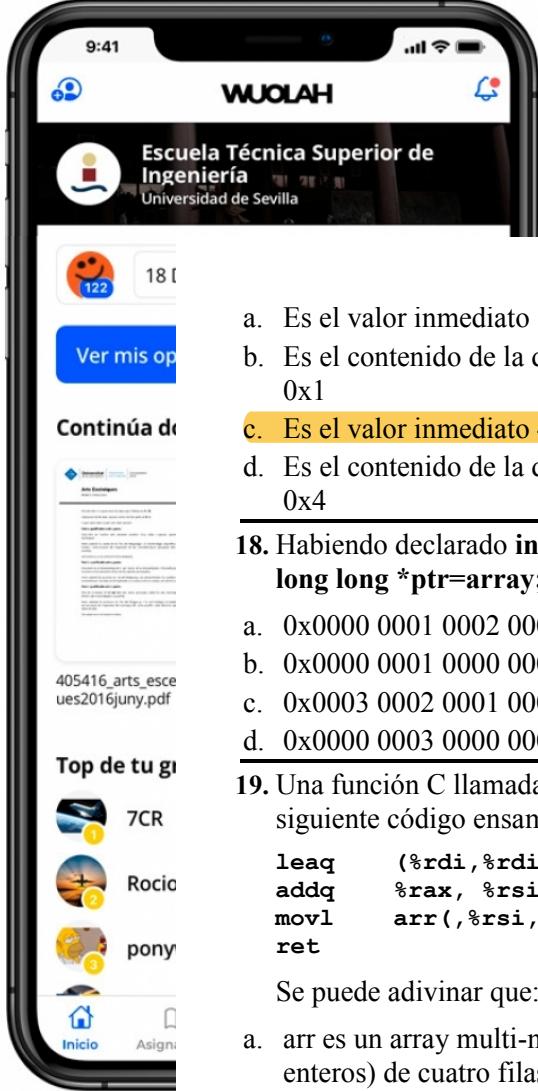
- a. 400431
- b. 400525
- c. **400425**
- d. 40043d

7. Para traducir una asignación condicional (**a = b ? c : d ;**) de lenguaje C a lenguaje ensamblador, gcc puede que utilice...

- a. Un salto incondicional, según la condición expresada en el código C, y otro salto incondicional
- b. Un salto condicional, según la condición opuesta a la del código C, y otro salto condicional



- c. Una instrucción de movimiento condicional, pero sólo si el procesador es Pentium Pro/II o superior
- d. Una instrucción de movimiento incondicional, pero sólo si el S.O. es de 64bits
- 
8. La instrucción **cmove %rdx,%rax**
- copia el byte bajo de rdx en el byte bajo de rax
  - copia en rax el byte de memoria apuntado por la dirección contenida en rdx
  - copia en rax el contenido de rdx si rax es menor que rdx
  - copia en rax el contenido de rdx si CF= 1**
- 
9. Uno de los puntos clave de la traducción que gcc hace de una construcción switch-case de lenguaje C a lenguaje ensamblador es...
- el salto condicional hacia atrás
  - el salto relativo a contador de programa
  - el salto directo
  - el salto indirecto**
- 
10. El procesador utiliza el puntero de pila...
- En las instrucciones de llamadas y retornos de subrutinas**
  - En todo tipo de instrucciones de saltos, incluyendo llamadas y retornos a subrutinas
  - En todas las instrucciones que tengan al menos dos accesos a memoria
  - En todas las instrucciones
- 
11. ¿Cuál de las siguientes instrucciones situada al principio de una función se utilizará probablemente para crear espacio en la pila para variables locales sin inicializar?
- sub \$0x30, %rsp**
  - add \$0x30, %rsp
  - sub \$0x30, %rbp
  - add \$0x30, %rbp
- 
12. En la convención de llamada SystemV AMD64 seguida por gcc Linux/x86-64...
- RAX es un registro salva-invocante, por eso en cualquier función hay que salvarlo antes de modificarlo
  - R10 es un registro salva-invocante, por eso si es necesario hay que salvarlo antes de llamar a función**
- caller saved - salva invocante  
callee saved - salva invocado*
- 
- c. R11 es un registro salva-invocado, por eso en cualquier función hay que salvarlo antes de modificarlo
- d. RBP es un registro salva-invocado, por eso si es necesario hay que salvarlo antes de llamar a función
- 
13. Un procedimiento llamado por una instrucción call debe guardar y restaurar los registros siguientes siempre que los altere:
- %rsi, %ordi
  - %rax, %rbx, %rcx, %rdx
  - %rax, %rdx, %rcx
  - %rbx, %rbp**
- 
14. Dada una función que devuelve la suma de 8 enteros en x86-64, ¿cuál de las siguientes instrucciones suma el 7º argumento?
- add -0x8(%rsp), %eax
  - add 0x8(%rsp), %eax**
  - add -0x4(%rsp), %eax
  - add 0x4(%rsp), %eax
- 
15. En el fragmento de programa siguiente:
- ```
66b: e8 8a ff ff ff  callq 5fa <f>
670: 48 83 c4 10      add $0x10,%rsp
```
- ¿Cuál es el valor que introduce en la pila la instrucción callq?
- 0x670
 - 0xffffffff8a
 - 0x66b
 - 0x5fa
-
16. En el fragmento de programa siguiente:
- ```
66b: e8 8a ff ff ff callq 5fa <f>
670: 48 83 c4 10 add $0x10,%rsp
```
- la instrucción callq suma al contador de programa la cantidad:
- 0x76
  - 0x5fa
  - 0xffffffff8a
  - 0x76
- 
17. Suponga la siguiente llamada a una función f de 4 argumentos:
- ```
mov    $0x1, %ecx
mov    $0x2, %edx
mov    $0x3, %esi
mov    $0x4, %edi
callq 5fa <f>
```
- El primer parámetro de llamada a la función:



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

- a. Es el valor inmediato 1
b. Es el contenido de la dirección de memoria 0x1
c. Es el valor inmediato 4
d. Es el contenido de la dirección de memoria 0x4

- 18.** Habiendo declarado **int array={0,1,2,3};** y **long long *ptr=array;** ¿cuánto vale **ptr[1]**?
- a. 0x0000 0001 0002 0003
b. 0x0000 0001 0000 0000
c. 0x0003 0002 0001 0000
d. 0x0000 0003 0000 0002

- 19.** Una función C llamada **get_el(...)** genera el siguiente código ensamblador.

```
leaq    (%rdi,%rdi,4), %rax
addq    %rax, %rsi
movl    arr(%rsi,4), %eax
ret
```

Se puede adivinar que:

- a. arr es un array multi-nivel (punteros a enteros) de cuatro filas
b. arr es un array multi-nivel pero no se pueden adivinar las dimensiones
c. arr es un array bidimensional de enteros, no se pueden adivinar dimensiones
d. arr es un array bidimensional de enteros, con cinco columnas

- 20.** Las microoperaciones de la fase de captación de una instrucción:

- a. Son comunes para todas las instrucciones
b. Dependen del código de operación de la instrucción que se encuentra en el registro de instrucción
c. Dependen de los indicadores de estado y del código de operación de la instrucción que se encuentra en el registro de instrucción
d. Dependen del valor del contador de programa

- 21.** Para el procesador con unidad de control microprogramada estudiado en clase, Tanenbaum propone codificar los 16 registros y añadir una señal "PERC" para habilitar la carga desde el bus C (recordar que era un diseño típico con 3 buses) y así no perder expresividad/paralelismo. El ahorro de bits en cada microinstrucción debido a esta técnica es de

- a. 40 bits
b. 39 bits
c. 35 bits
d. 29 bits

- 22.** Un procesador está segmentado en k etapas. Cada una de ellas consume un tiempo t. La aceleración ideal (si no hay riesgos) al ejecutar 5 instrucciones respecto a un procesador no segmentado será:

- a. 5k / (4+k)
b. (4+k) / 5t
c. 4k / (5+k)
d. (5+k) / 4t

- 23.** ¿Cuál de las siguientes afirmaciones sobre la E/S programada con consulta de estado es cierta?

- a. Si se emplea E/S programada puede hacerse con consulta de estado o sin consulta de estado
b. Un programa que realice salida programada con consulta de estado no ejecutará ninguna instrucción de entrada o carga
c. Sólo la E/S por DMA libera a la CPU de realizar la consulta de estado del dispositivo de E/S
d. La escritura de un led requiere consulta de estado

- 24.** En un sistema de interrupciones vectorizado y en daisy-chain, ¿cuál de las siguientes afirmaciones es cierta?

- a. El procesador informa de un ciclo de reconocimiento de interrupción con la señal de reconocimiento de interrupción (INTA) y la identificación de los dispositivos se realiza por consulta de estado
b. La gestión de prioridades queda establecida por el orden en que los dispositivos reciben la señal INTA y el dispositivo se identifica por un dato que deposita en el bus
c. La gestión de prioridades queda establecida por el orden en que los dispositivos reciben la señal INTA y la identificación de los dispositivos se realiza leyendo sus registros de estado
d. El daisy-chain asigna a todos los dispositivos la misma prioridad y la identificación de los dispositivos se realiza leyendo sus registros de estado

25. ¿Cuál de las siguientes características es menos probable que pueda programarse en un canal DMA?

- a. dos direcciones (origen y destino)
 - b. dos tamaños (copia origen y copia destino)
 - c. cuál de las dos direcciones es de E/S (si alguna lo es) en lugar de Memoria
 - d. si se desea producir una IRQ al terminar
-

26. En un computador con una jerarquía de memoria de dos niveles se observa experimentalmente que el tiempo medio de acceso a la memoria es de 300 ns cuando en realidad el tiempo medio de acceso al primer nivel es de 6 ns. Sabiendo que el tiempo de acceso al segundo nivel es de 3 microsegundos, ¿cuál sería aproximadamente el porcentaje de fallos en los accesos al primer nivel?

- a. 90%
 - b. 1%
 - c. 10%
 - d. 99%
-

27. El orden de magnitud del tiempo de acceso a la memoria DRAM de un computador es de:

- a. Picosegundos
 - b. Nanosegundos
 - c. Microsegundos
 - d. Milisegundos
-

28. Una memoria estática tiene un bus de datos de 32 bits y su bus de direcciones es de 20 bits, ¿cuál es su capacidad?

- a. 4 MBytes
 - b. 1 MByte
 - c. 32 MBytes
 - d. 80 GBytes
-

29. En una cache asociativa por conjuntos de 2^v vías con 2^b líneas (marcos de bloque) de 2^w palabras, el gestor de memoria **no** considera como campo (conjunto de bits contiguos con significado o relevancia) los siguientes bits:

- a. últimos w bits (0...w-1) (los menos significativos)
 - b. bits w...w+c-1 (con c=b-v)
 - c. bits w...w+c-1 (siendo $2^c=n$ conjuntos)
 - d. bits b...b+c-1 (siendo $2^c=n$ conjuntos)
-

30. Para obtener una única velocidad comparativa final, el benchmark SPEC CPU combina las ganancias en velocidad de ejecución de una serie de tests, respecto a un ordenador de referencia, usando...

- a. la mediana
 - b. la media aritmética
 - c. la media geométrica
 - d. la moda
-



PruebaAlien

www.wuolah.com/student/PruebaAlien



Ex-Feb-18-Test-Teo.pdf

examen 2018 febrero



2º Estructura de Computadores



Grado en Ingeniería Informática



Escuela Técnica Superior de Ingenierías Informática y de
Telecomunicación
Universidad de Granada



Descarga la APP de Wuolah.
Ya disponible para el móvil y la tablet.





**KEEP
CALM
AND
ESTUDIA
UN POQUITO**

Nombre:
DNI:
Grupo:

Test de Teoría (3.0p)

Todas las preguntas son de elección simple sobre 4 alternativas.

Cada respuesta vale 0.1p si es correcta, 0p si está en blanco o claramente tachada, -0.03p si es errónea.

Anotar las respuestas (a, b, c ó d) en la siguiente tabla.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

1. ¿Cuál es el valor mínimo (más negativo) que puede tomar un entero de 32 bits en complemento a dos? (el punto se usa como separador)

- a. -2.147.483.647
- b. **-2.147.483.648**
- c. -4.294.967.295
- d. -4.294.967.296

2. ¿Cómo se representa el valor -1 como entero con signo en 14 bits?

- a. 0xFFFF
- b. **0x3FFF**
- c. las respuestas anteriores no son válidas porque usan hexadecimal; habría que usar binario
- d. no se puede porque 14 no es múltiplo de 4

3. ¿Cuál de las siguientes no es una unidad de la arquitectura Von Neumann?

- a. Unidad central de proceso
- b. Memoria principal
- c. Sistema de entrada/salida
- d. **Núcleo del sistema operativo**

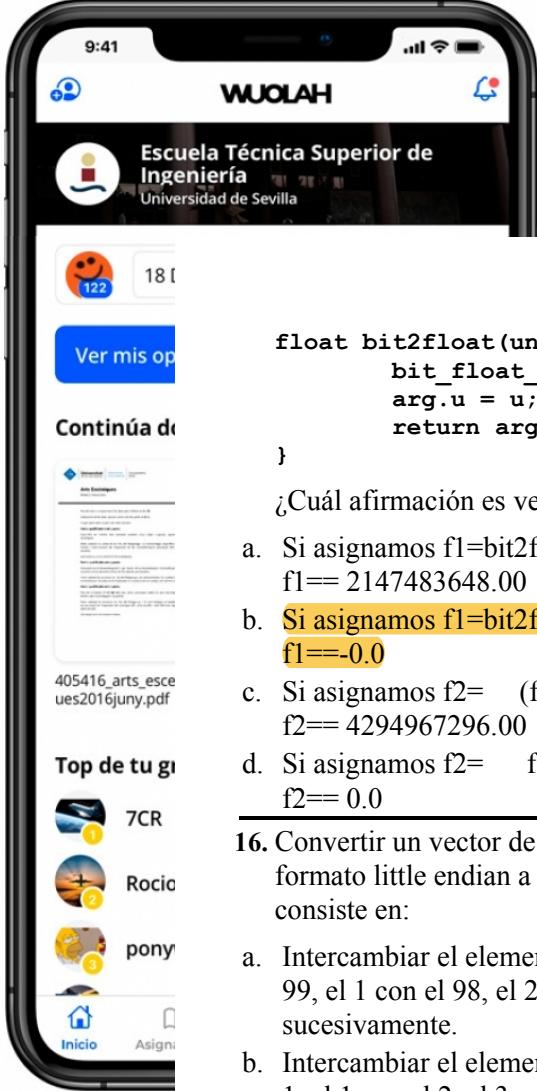
4. ¿Cuál de las siguientes afirmaciones es verdadera?

- a. La arquitectura Von Neumann en la que se basan los computadores tradicionales consiste en tener los datos separados de las instrucciones en memorias distintas.
- b. El registro de estado es un registro transparente al usuario, ya que éste no

puede utilizarlo en las instrucciones máquina.

- c. **El registro de instrucción es un registro transparente al usuario, ya que éste no puede utilizarlo en las instrucciones máquina.**
 - d. La unidad de control necesita como entrada el registro contador de programa, para saber cuál es la instrucción que debe ejecutar a continuación.
-
5. ¿Qué es el lenguaje máquina?
- a. Conjunto de datos binarios que representan señales eléctricas internas de la unidad de control de un microprocesador.
 - b. Conjunto de sentencias en un lenguaje escrito que se utilizan para generar programas codificados en lenguaje ensamblador.
 - c. Conjunto formado por las siglas asignadas a las instrucciones del repertorio de instrucciones más un conjunto de directivas que facilitan la generación del código binario.
 - d. **Conjunto de instrucciones en formato binario que entiende un determinado procesador.**
-
6. ¿Cuál de los siguientes elementos no forma parte de la Arquitectura del Repertorio de Instrucciones (ISA)?
- a. Descripción del espacio de direccionamiento de la memoria y de la E/S.

- b. Descripción de los campos de bits en los que están organizadas conceptualmente las microinstrucciones.
- c. Descripción de los registros de datos, registros de estado y control.
- d. Descripción de los tipos de datos sobre los que opera el lenguaje máquina.
-
7. ¿Cuál de las siguientes definiciones de modos de direccionamiento es ***incorrecta***?
- Inmediato: el dato está codificado dentro de la propia instrucción, en uno de los campos en los que se divide el formato de instrucción.
 - Registro: el dato se encuentra en un registro de propósito general.
 - Directo: la dirección se calcula como la suma de un dato codificado en la propia instrucción y el contenido de un registro de propósito general.**
 - Indirecto: el dato está contenido en una posición de memoria que es apuntada por un registro de propósito general.
-
8. Respecto a los registros enteros en arquitectura IA32 de 32bits (x86)
- Se puede acceder a 8, y en cada uno de esos 8 registros enteros, se puede acceder a todos los 32 bits (p.ej. EAX), a los 16 bits menos significativos (p.ej. AX) ó a los 8 LSBs (p.ej. AL)
 - Se puede acceder a 8, y en cada uno de esos 8 registros enteros, se puede acceder a todos los 32 bits (p.ej. EAX), a los 16 bits menos significativos (p.ej. AX), a los 8 LSBs (p.ej. AL) o a los bits 8-15 (p.ej. AH)
 - Se puede acceder a 8 de cada tamaño (32, 16, 8 bits), aunque no todos los registros tienen versión de 8 y 16 bits**
 - No hay distintos tamaños, son sólo registros de 32 bits, como corresponde a dicha arquitectura
-
9. ¿Cuál de las siguientes instrucciones es errónea? (sale mensaje de error al intentar ensamblar):
- movw %dx, (%eax)
 - movb \$0xFF, (%dl)**
 - movswl (%eax), %edx
 - movzbl %dl, %eax
-
10. ¿Qué modo de direccionamiento usa el operando fuente en la instrucción mov (%rcx), %al?
- Directo a memoria
 - Indirecto a memoria a través de registro**
 - Registro
 - Inmediato
-
11. Si el contenido del registro %rax es 0x10 antes de ejecutar la instrucción shl \$0xc,%rax, ¿cuánto es su contenido tras ejecutarla?
- 0x10000**
 - 0x1000
 - 0x4000
 - 0x800
-
12. En el fragmento de código
- ```
804854e: e8 3d 06 00 00 call 8048b90
8048553: 50 pushl %eax
```
- la instrucción call suma al contador de programa la cantidad:
- 0x0000063d**
  - 0x08048553
  - 0x0804854e
  - 0x50
- 
13. ¿Cuál de los siguientes registros x86-64 es distinto del resto en convenio de uso? (salva-invocante/invocado)
- RBX**
  - RCX
  - RSI
  - R8
- 
14. Respecto a requisitos de alineamiento de structs en gcc/IA32 x86 y x86-64, una de las siguientes afirmaciones es **\*FALSA\***
- en x86 Linux alinea double a 4x
  - en x86 Linux alinea long double a 4x
  - en x86-64 Linux alinea double a 8x
  - en x86-64 Linux alinea float a 8x**
- 
15. Se definen las variables, unión y función C siguientes:
- ```
float      f1;
unsigned   u1=0x80000000;
float      f2;
typedef union {
    float f;
    unsigned u;
} bit_float_t;
```



Descarga la APP de Wuolah.

Ya disponible para el móvil y la tablet.

Available on the
App Store

GET IT ON
Google Play

```
float bit2float(unsigned u) {
    bit_float_t arg;
    arg.u = u;
    return arg.f;
}
```

¿Cuál afirmación es verdadera?

- Si asignamos $f1=bit2float(u1)$; entonces $f1== 2147483648.00$
 - Si asignamos $f1=bit2float(u1)$; entonces $f1== -0.0$**
 - Si asignamos $f2= (float)u1$; entonces $f2== 4294967296.00$
 - Si asignamos $f2= float(u1)$; entonces $f2== 0.0$
-

16. Convertir un vector de 100 shorts de formato little endian a formato big endian consiste en:

- Intercambiar el elemento 0 del vector con el 99, el 1 con el 98, el 2 con el 97 y así sucesivamente.
 - Intercambiar el elemento 0 del vector con el 1, el 1 con el 2, el 3 con el 4 y así sucesivamente.
 - Cambiar el orden en memoria de los 4 bytes de cada elemento, es decir, en cada elemento intercambiar el byte 0 con el 3 y el 1 con el 2.
 - Cambiar el orden en memoria de los 2 bytes de cada elemento, es decir, en cada elemento intercambiar el byte 0 con el 1.
-

17. Motivos que impiden que la ganancia (aceleración) de un cauce segmentado sea ideal (señalar la respuesta *FALSA*)

- registros de acople (coste de segmentación)
 - fragmentación desigual (duración de etapas)
 - riesgos (hazards)
 - cola de instrucciones (precaptación)
-

18. Un procesador de 1GHz tarda 4ns en realizar 4 instrucciones sin realizar segmentación de cauce. ¿Cuanto tardaría en realizar 9 instrucciones con segmentación de cauce de 4 etapas si no existiera ningún retraso en ninguna de las instrucciones?

- 2 ns
 - 3 ns
 - 4.5 ns
 - 9 ns
-

19. ¿Qué es un controlador de E/S?

- Un circuito electrónico que implementa la memoria del computador.
 - Un circuito impreso del tipo DIMM.
 - Un circuito electrónico que puede guardar temporalmente datos enviados desde el procesador al periférico o viceversa.
 - Un bus que permite interconectar distintos periféricos entre sí.
-

20. Respecto a la interfaz de E/S, ¿cuál de las siguientes afirmaciones es *FALSA*?

- Involucra tareas que se pueden implementar parte en hardware y parte en software.
 - Permite configurar el funcionamiento del periférico en un momento determinado, y además conocer su estado.
 - Puede guardar temporalmente en registros internos tanto datos generados por el periférico para ser enviados al procesador, como datos que son enviados desde el procesador al periférico.
 - Una interfaz de entrada recibe los datos desde el procesador y los transforma y envía al periférico en formato digital.
-

21. ¿Cuál de las siguientes características corresponde a E/S mapeada en memoria?

- Determinadas zonas del espacio de direccionamiento del procesador se asignan por convenio a controladores de E/S.
 - Un ejemplo de mecanismo de E/S mapeada en memoria es la instrucción IN de los procesadores Intel.
 - Una misma dirección se usa alternativamente para E/S y para memoria en distintos momentos de ejecución de un programa.
 - Un pin IO/M# del procesador permite distinguir si accedemos a E/S o a memoria.
-

22. ¿Cuál de las siguientes afirmaciones es *FALSA*?

- La consulta del estado del dispositivo por parte de la CPU se suele hacer con E/S programada (salvo con dispositivos que siempre están listos para transferir) y con E/S por IRQ (cuando se usa polling para determinar el origen de la IRQ).
- Se suele avisar a la CPU (mediante una IRQ) de que debe realizar alguna tarea, tanto en E/S por IRQ (obligatoriamente, la tarea es la transferencia) como en E/S por DMA (optativamente, el controlador DMA puede avisar de que acabó).

- c. Sólo E/S por DMA libera a la CPU de realizar la consulta de estado del dispositivo de E/S.
- d. Sólo E/S por DMA libera a la CPU de realizar la transferencia de los datos de E/S.

23. La instrucción máquina DI (Disable Interrupts), conocida como CLI (Clear Interrupt Flag) en x86, se utiliza para desactivar:

- a. Todas las interrupciones enmascarables
- b. Las interrupciones de inferior o igual prioridad a una dada
- c. Determinados niveles de interrupción de forma selectiva
- d. Las interrupciones software

24. ¿Cuál de los siguientes es un registro de un controlador de DMA?

- a. IR (Instruction Register)
- b. PC (Program Counter)
- c. SP (Stack Pointer)
- d. WC (Word Count)

25. El ancho de banda de memoria es:

- a. el número de bits que se pueden transferir entre ésta y la CPU en paralelo en una sola operación de lectura o escritura
- b. el número de bytes que se pueden leer/escribir por unidad de tiempo
- c. el tiempo que se tarda en transferir una palabra entre memoria y CPU
- d. el intervalo de frecuencias de reloj permitidas entre memoria y CPU

26. ¿Cuál de las siguientes afirmaciones sobre la memoria DRAM es ***incorrecta***?

- a. El principio de funcionamiento de los circuitos electrónicos de la memoria DRAM consiste en cargar o descargar un transistor.
- b. Los bits de memoria se organizan dentro del circuito integrado en forma de matriz de celdas de bit, en la que se pueden diferenciar filas y columnas.
- c. Un transistor en cada celda permite o no permite circular la corriente eléctrica a través de él. Cuando el transistor no deja pasar la corriente, la información queda almacenada durante un tiempo en el condensador. Cuando el transistor deja pasar corriente, el condensador se carga o se descarga.
- d. Cada celda de memoria está compuesta por un transistor y un condensador y almacena un bit de información.

27. ¿Cuál de las siguientes afirmaciones sobre memorias es correcta?

- a. La memoria cache se construye con tecnología electrónica de tipo DRAM.
- b. La memoria principal se construye con tecnología electrónica de tipo SRAM.
- c. Los chips de memoria DRAM se conectan entre sí en un circuito impreso constituyendo lo que se denomina DIMM.
- d. Las memorias SRAM no son volátiles; es decir, cuando no están alimentadas eléctricamente siguen guardando toda la información.

28. ¿Cuál de los siguientes grupos de señales no se usa en un chip de memoria SRAM?

- a. Selección de filas RAS# y de columnas CAS#.
- b. Datos D_{n-1}-D₀.
- c. Direcciones A_{n-1}-A₀.
- d. Selección de chip CS# y habilitación de escritura WE#.

29. ¿Qué es el tiempo de refresco de memoria?

- a. La cantidad de datos transferidos por segundo entre dos niveles de la jerarquía de memoria.
- b. El tiempo que se tarda en recargar los condensadores que almacenan los bits de datos para que no se pierdan.
- c. El tiempo que transcurre entre la solicitud de una operación en un determinado nivel de la jerarquía de memoria (lectura o escritura) y la recepción de todos los datos solicitados.
- d. El tiempo que tiene que transcurrir entre sucesivas solicitudes de acceso a un determinado nivel de la jerarquía de memoria.

30. ¿Cuál de las siguientes políticas está ***menos*** relacionada con la jerarquía memoria?

- a. Política de escritura: determina cómo se actualiza el nivel de la memoria i+1 cuando se ejecutan instrucciones de almacenamiento en el nivel i.
- b. Política de reemplazo: qué bloque se tiene que sustituir (reemplazar) cuando se trae un bloque desde otro nivel.
- c. Política de planificación: en qué orden se ejecutarán los procesos pendientes.
- d. Política de colocación: dónde se almacena un bloque de datos dentro de la memoria.

EDEN

El direccionamiento relativo a registro base utiliza...

Usuar Profesor

io es

- a) dos registros.
- b) un registro.
- c) dos desplazamientos contenidos en la propia instrucción.
- ✓ • d) un registro y un desplazamiento.

Puntuación: **1,00**

Una instrucción de salto si menor, para números positivos sin signo, tiene que comprobar el valor de:

Usuar Profesor

io es

- a) los bits de signo y desbordamiento
- b) el bit de signo
- ✗ • c) el bit de cero
- d) el bit de acarreo

Puntuación: **-0,33**

La instrucción setg %al:

U Pro
su fes
ari ore
o s

- a Pone siempre AL a 1.
)
- ✓ • b Pone AL a 1 en algunos casos.
)
- c No cambia el contenido de AL
)
- d Complementa AL si el resultado
) de la comparación anterior es A >
B.

Puntuación: **1,00**

¿Cuál de las siguientes instrucciones no modifica necesariamente la secuencia de ejecución del programa?

Usuar Profesor

io es

- a) RET
- b) CALL dir
- c) JNE dir
- d) JMP dir

Puntuación: **1,00**

El lenguaje máquina es...

Usuar Profesor

io es

- a) fácilmente legible por el programador.
- b) portable entre arquitecturas.
- c) difícil de codificar manualmente.
- d) una alternativa razonable al uso del lenguaje ensamblador.

Puntuación: **1,00**

¿En qué generación, dentro de la historia de los computadores digitales, aparece la segmentación de cauce?

Usuar Profesor

io es

- a) primera
- b) segund
a
- c) tercera
- d) cuarta

Puntuación: **1,00**

[T1.5]

¿En qué generación, dentro de la historia de los computadores digitales, se alcanzaron tiempos de conmutación del orden de nanosegundos?

Usuar Profesor

io es

- a) primera

- b) segunda
- c) tercera
- d) cuarta

Puntuación: **1,00**

En IA32, el registro contador de programa se denomina:

Usuar Profesor

io es

- a) PC
- b) RIP
- c) PC
R
- d) EIP

Puntuación: **-0,33**

En cdecl/x86, ¿cuál de los siguientes registros tiene que ser guardado por la función llamada si es alterado por ésta?

- a) eax
- b) ecx
- c) ebx
- d) edx

¿Cuál de las siguientes listas está correctamente ordenada temporalmente?

- a) 486, 8086, Pentium MMX, Core 2, Pentium III, Pentium 4
- b) 8086, 486, Pentium MMX, Pentium III, Pentium 4, Core 2
- c) 8086, 486, Pentium III, Pentium MMX, Core 2, Pentium 4
- d) 486, 8086, Core 2, Pentium III, Pentium 4, Pentium MMX

Si el contenido de EBP es 0x13000, ¿a qué dirección se hace referencia con la instrucción INC –0x80(%EBP)?

- ✓ a) 0x13080
- b) 0x13000
- c) 0x80
- d) 0x12F80

¿Cuál es la dirección física del operando fuente de la instrucción ADD AX, ETIQUETA siendo ETIQUETA = 7000h, CS = 1500h, DS = 4500h, e IP = 25h (la instrucción ocupa 3 bytes)?

- ✗ a) 53000h
- b) **4C000h**
- c) 15025h
- d) 5C000h

7 [T1.2]

El modo de direccionamiento en el que se especifica un registro y una dirección de memoria cuyo contenido se suma al contenido del registro base para obtener la dirección efectiva, se conoce como:

ón Us Prof
ún ua eso
ic río res

- a a base con
 -) desplazamiento
- b directo o absoluto
 -)
- ✗ c indirecto a registro
 -) través de memoria
- d ninguno de los
 -) anteriores

Puntuación: **-0,33**

8 [T1.5]

Elección única

Usuar Profesor

io es

- a) primera
- b) segund
- a
- ✓ • c) tercera
- d) cuarta

Puntuación: **1,00**

Sobre el ensamblador:

Usuar Profesor

io es

- a) La calidad de un programa ensamblador afectará menos al tiempo de ejecución de los programas generados por él que la calidad de un compilador.
- b) Las etiquetas permiten que el programador especifique el destino de un salto de forma que éste no tenga que modificarse manualmente cuando el programa varíe de tamaño.
- c) El lenguaje ensamblador elimina la posibilidad de errores en la generación de la representación en lenguaje máquina de cada instrucción.
- ✓ • d) Todas las respuestas son ciertas.

Suponiendo que todos los registros inicialmente contienen el valor 1 y que el destino es el primer registro, ¿cuál es el valor de r1 tras la ejecución de la siguiente secuencia de instrucciones?

```
mov r1, #4
mov r2, #3
add r3, r1, r1
```

sub r1, r3, r2

mul r3, r1, r1

Usuar Profesor

io es

- a) 3
- 6
- b) 6
- ✓ • c) 5
- d) 2
- 0

Puntuación: **1,00**

[T2.2.3]

En los casos concretos indicados para las siguientes instrucciones IA32, ¿cuál no funciona como instrucción de transferencia?

Usuar Profesor

io es

- ✓ • a) cmpl 0x08048040, %eax
- b) movl \$0x15, %eax
- c) pushl %eax
- d) leal variable, %eax

Puntuación: **1,00**

¿Qué medida de prestaciones es la más fiable de todas las posibles?

Usuar Profesor

io es

- a) MIPS
- ✗ b) MIPS equivalentes
- c) ninguna de las otras respuestas es correcta
- d) MFLOPS

Puntuación: **-0,33**

En un microprocesador de 4 bits, una operación en la que el bit 0 de un registro se copia en el acarreo, después el bit 1 se copia en el bit 0, después el bit 2 se copia en el bit 1, y por último el bit 3 se copia en el bit 2, es:

Usuar Profesor

io es

- a) Un desplazamiento lógico a la derecha.
- b) Un desplazamiento aritmético a la derecha.
- c) Una rotación a la derecha.
- d) Una rotación a la derecha a través de acarreo.

Puntuación: **-0,33**

El ancho de palabra de una memoria corresponde a:

Usuar Profesor

io es

- a) La longitud del registro de datos de la memoria.
- b) El número de posiciones que la componen.
- c) El número que identifica únicamente cada posición de la memoria.
- d) La longitud del registro de direcciones de la memoria.

Puntuación: **-0,33**

Una CPU con bus de direcciones de 16 bits y bus de datos de 8 bits tiene un registro de 8 bits conectado al bus de datos y a la unidad de control. Puede tratarse del registro

Usuar Profesor

io es

- a) De direcciones
- b) Puntero de pila
- c) Contador de programa
- d) De instrucción

Puntuación: **1,00**

El objetivo de un diseño CISC es...

Usuar Profesor

io es

- a) disminuir el tamaño medio de instrucción.
- b) disminuir la frecuencia de reloj.
- c) disminuir el número de instrucciones a ejecutar por un programa.
- d) disminuir el número medio de ciclos por instrucción.

Puntuación: **1,00**

En la captación de la instrucción:

Usuar Profesor

io es

- a) en MBR indicamos la dirección donde está la instrucción y en la ALU recogemos la instrucción.
- b) en MAR indicamos la dirección donde está la instrucción y en la ALU recogemos la instrucción.
- c) en MAR indicamos la dirección donde está la instrucción y en MBR recogemos la instrucción.
- d) en MBR indicamos la dirección donde está la instrucción y en MAR recogemos la instrucción.

Puntuación: **1,00**

¿En qué generación, dentro de la historia de los computadores digitales, aparecen las memorias de semiconductores?

Usuar Profesor

io es

- a) Tercera generación.
- b) Cuarta generación.
- c) Segunda generación.
- d) Quinta generación.

Puntuación: **-0,33**

En el 8086, la dirección efectiva de la cabecera de pila vendrá dada por:

Usuar Profesor

io es

- a) BP

- b) SS * 10h +
SP
- X** c) SP
- d) SS * 10h +
BP

Puntuación: **-0,33**

Cuando un operando se encuentra almacenado en un registro, se trata de un direccionamiento...

Usuar Profesor

io es

- ✓**
- a) directo a registro
 - b) a registro inmediato
 - c) a registro literal
 - d) ninguno de los anteriores

Puntuación: **1,00**

Cuando un operando se encuentra almacenado en un registro, se trata de un direccionamiento...

Usuar Profesor

io es

- ✓**
- a) directo a registro
 - b) a registro inmediato
 - c) a registro literal
 - d) ninguno de los anteriores

Puntuación: **1,00**

Si AX = 0xFA50 y ejecutamos AND \$0xFF, %AX

Usuar Profesor

io es

- X**
- a) El registro AH se pone a FF
 - b) El registro AH se pone a 0
 - c) El registro AL se pone a FF
 - d) El registro AL se pone a 0

Puntuación: **-0,33**

En una máquina con 32 registros direccionables e instrucciones de 16 bits:

Usuar Profesor

io es

X

- a) no se pueden codificar a la vez 64 instrucciones de 0 direcciones, 63 instrucciones de dos registros y 16 instrucciones de 1 registro.
- b) no se pueden codificar a la vez 63 instrucciones de dos registros, una instrucción de 0 direcciones y 32 instrucciones de 1 registro.
- c) no se pueden codificar a la vez 64 instrucciones de 1 registro, 32 instrucciones de dos registros y 32 instrucciones de 0 direcciones.
- d) no se pueden codificar a la vez 62 instrucciones de dos registros, 32 instrucciones de 1 registro y 64 instrucciones de 0 direcciones.

Puntuación: **-0,33**

7

¿Qué tipos de instrucciones se emplean más en una arquitectura de registros de propósito general?

Elecció
n única

Usua
rio Profes
ores

X

- a) De desplazamiento y rotación.
- b) Aritmético-lógicas.
- c) De transferencia de datos.
- d) De transferencia de control.

Puntuación: **-0,33**

8

Elecció n únic a Si d es un desplazamiento, r un registro índice e i una constante apropiada, el modo de direccionamiento indexado con postautodecremento realiza...

Usua rio
n
únic a
Profes ores



- a) dirección efectiva = $r + d$; $r = r - i$
- b) dirección efectiva = $r - i$; $r = r - d$
- c) $r = r - i$; dirección efectiva = $r - d$
- d) $r = r + i$; dirección efectiva = $r - d$

Puntuación: 1,00

En el direccionamiento inmediato:

Usuar Profesor

io es



- a) el tamaño (rango de valores) de la constante está limitado.
- b) el operando es una constante de tamaño arbitrario contenida en la misma instrucción.
- c) el código de operación indica el operando afectado.
- d) una vez captada y decodificada la instrucción, se accede a memoria para obtener el valor del operando.

Puntuación: 1,00

En el RISC-I, una ventana de registros contiene:

Usuar Profesor

io es



- a) registros para recibir parámetros del procedimiento llamador.
- b) registros para almacenar variables locales.
- c) registros para enviar parámetros a procedimientos.
- d) todas las respuestas son ciertas.

Puntuación: -0,33

Si queremos almacenar la palabra de 16 bits 0x9660 en una memoria de bytes según "little-endian", quedará almacenada a partir de la posición 0x1000 como:

Usuar Profesor

io es

- a) M[0x1000]=0x06 y
M[0x1001]=0x69
- b) M[0x1000]=0x69 y
M[0x1001]=0x06
- ✓ • c) M[0x1000]=0x60 y
M[0x1001]=0x96
- d) M[0x1000]=0x96 y
M[0x1001]=0x60

Puntuación: **1,00**

¿En qué generación, dentro de la historia de los computadores digitales, aparece la memoria cache?

Usuar Profesor

io es

- a) primera
- b) segund
a
- ✓ • c) tercera
- d) cuarta

Puntuación: **1,00**

GCC/Linux IA32 resuelve el ajuste de marco de pila al comenzar un procedimiento mediante las instrucciones:

Usuar Profesor

io es

- a) pushl %ebp movl %esp,
%ebp
- b) movl %esp, %ebp popl %esp
- ✗ c) pushl %esp movl %ebp,
%esp
- d) movl %ebp, %esp popl %ebp

Puntuación: **-0,33**

¿En qué orden debería ejecutarse en una máquina de tipo pila la operación aritmética ($a+b/c-d$)?

Usuar Profesor

io es

a) $a + b / c -$

d

b) $a b + c d -$

/



• c) $a b c / + d$

-

d) $a b + / c d$

-

Puntuación: **1,00**

¿Qué arquitectura se caracteriza por presentar una gran variación en la longitud de las instrucciones?

Usuar Profesor

io es

a) registro-registro

b) registro-memoria



• c) memoria-memoria

d) ninguna de las anteriores es
cierta

Puntuación: **1,00**

¿Cuál de las siguientes parejas de microprocesadores representa mejor los conceptos RISC?

Usuar Profesor

io es



• a) MIPS, SPARC

b) PA-RISC,
PowerPC

c) Itanium, Alpha

d) Pentium, Athlon

Puntuación: **1,00**

En el 8086, la dirección efectiva de la cabecera de pila vendrá dada por:

Usuar Profesor

io es

- a) SS * 10h +
 BP
- b) BP
- c) SS * 10h +
 SP
- X d) SP

Puntuación: **-0,33**

9 [T1.5]

Elecció
n única

Usuar Profesor

io es

- a) primera
- b) segunda
- ✓ c) tercera
- d) cuarta

Puntuación: **1,00**

2

[T1.1]

Elección
única

Usu Profe
ario sores

- ✓ • a) de transferencia de datos con
) memoria
- b) aritmético-lógicas
)
- c) de transferencia de datos
) entre registros

d de desplazamiento y rotación
)

Puntuación: **1,00**

Una máquina superescalar es aquella que:

Usuar Profesor

io es

- a) basa su funcionamiento en la segmentación software como forma de incrementar el paralelismo.
- b) las instrucciones tienen un campo por cada unidad funcional al realizarse varias operaciones por instrucción.
- ✓ • c) emite simultáneamente múltiples instrucciones por ciclo de reloj, por ejemplo, una entera y otra de coma flotante.
- d) ninguna respuesta de las anteriores es correcta.

Puntuación: **1,00**

¿En qué generación, dentro de la historia de los computadores digitales, aparece la segmentación de cauce?

Usuar Profesor

io es

- a) primera
- b) segund
- a
- ✓ • c) tercera
- d) cuarta

Puntuación: **1,00**

Si el registro EAX contiene X, La secuencia de instrucciones siguiente:

cmpl \$6, %eax

jae Destino

salta a la etiqueta Destino sólo si:

Usuar Profesor

io es

- ✓ • a) $X < 0 \text{ || } X \geq 6$
- b) $X \leq 6$
- c) $X \geq 0 \text{ && } X \leq 6$
- d) $X > 6$

Puntuación: **1,00**

El lenguaje máquina es...

Usuar Profesor

io es

- ✓ • a) difícil de codificar manualmente.
b) portable entre arquitecturas.
c) fácilmente legible por el programador.
d) una alternativa razonable al uso del lenguaje ensamblador.

Puntuación: **1,00**

Si usamos una estructura de bus con DMA:

Usuar Profesor

io es

- a) al bus del sistema sólo se conecta la CPU y la MP, ya que el DMA se conecta directamente a MP para realizar las transferencias de datos.
- ✗ b) podemos prescindir de controladores de E/S ya que el controlador de DMA se ocupa de controlar las transferencias hacia/desde los periféricos.
- c) la velocidad de este controlador establece la velocidad del bus del sistema.
- d) la CPU puede dejar las transferencias entre MP y periféricos en manos de este controlador (DMA) y seguir ejecutando otras instrucciones.

Puntuación: **-0,33**

Sobre el direccionamiento relativo al contador de programa:

Usuar Profesor

io es

- a) Favorece la implementación de código reubicable.
- b) Su uso en los saltos reduce el tamaño de la instrucción.
- c) Es adecuado para alcanzar instrucciones próximas a la que se está ejecutando.
- ✓ • d) Todas las respuestas son ciertas.

Puntuación: **1,00**

Si %eax contiene x, ¿cuál de las siguientes instrucciones calcula x*5?

Usuar Profesor

io es

- a) leal 3(%eax,2),%edx
- b) leal 3(%eax,%eax,2),%edx
- c) leal 4(%eax,%eax),%edx
- d) leal (%eax,%eax,4),%edx

Puntuación: **1,00**

Si queremos almacenar la palabra de 16 bits 8965h en memoria según "big-endian", quedará almacenada a partir de la posición 1000h como:

Usuar Profesor

io es

- a) en el byte 1000h se guarda A6h y en el 1001h 91h
- b) en el byte 1000h se guarda 91h y en el 1001h A6h
- c) en el byte 1000h se guarda 89h y en el 1001h 65h
- d) en el byte 1000h se guarda 65h y en el 1001h 89h

Puntuación: **1,00**

¿Cuál es la dirección física del operando fuente de la instrucción ADD AX, ETIQUETA siendo ETIQUETA = 7000h, CS = 1500h, DS = 4500h, e IP = 25h (la instrucción ocupa 3 bytes)?

Usuar Profesor

io es

- a) 15025h
- b) 4C000 h
- c) 5C000 h
- d) 53000h

Puntuación: **1,00**

Sobre el ensamblador:

Usuar Profesor

io es

- a) La calidad de un programa ensamblador afectará menos al tiempo de ejecución de los programas generados por él que la calidad de un compilador.
- b) Las etiquetas permiten que el programador especifique el destino de un salto de forma que éste no tenga que modificarse manualmente cuando el programa varíe de tamaño.
- c) El lenguaje ensamblador elimina la posibilidad de errores en la generación de la representación en lenguaje máquina de cada instrucción.
- ✓ • d) Todas las respuestas son ciertas.

Puntuación: **1,00**

Un procesador emplea codificación en bloque del código de operación. Existen 123 instrucciones que tienen una longitud de 16 bits. ¿A cuántas direcciones de memoria pueden acceder como máximo si todas emplean una estructura “código de operación + dirección de memoria”?

Usuar Profesor

io es

- a) 128.
- ✗ b) 256.
- c) 1024
- .
- d) 512.

Puntuación: **-0,33**

¿Qué arquitectura se caracteriza por presentar una gran variación en la longitud de las instrucciones?

Usuar Profesor

io es

- a) registro-registro
- b) registro-memoria
- ✓ • c) memoria-memoria
- d) ninguna de las anteriores es cierta

Puntuación: **1,00**

Para direccionar una memoria de 1 G palabras de 32 bits cada una, que se direcciona byte a byte, se necesitarán:

Usuar Profesor

io es

X

- a) 33 bits
- b) 21 bits
- c) 32 bits
- d) 31 bits

Puntuación: **-0,33**

[T1.5]

¿En qué generación, dentro de la historia de los computadores digitales, aparece la memoria virtual?

Usuar Profesor

io es

✓

- a) primera
- b) segund a
- c) tercera
- d) cuarta

Puntuación: **1,00**

En el RISC-I, una ventana de registros contiene, entre otros registros,...

Usuar Profesor

io es

✓

- a) registros para recibir parámetros del procedimiento llamador
- b) registros para almacenar matrices de enteros
- c) registros para enviar parámetros a otros procesos
- d) todas las respuestas son falsas

Puntuación: **1,00**

En el arbitraje de un bus...

Usuar Profesor

io es

- a) los dispositivos pasivos pueden requerir el uso del bus para iniciar una transferencia
- b) si hay un único dispositivo pasivo, siempre funciona como esclavo
- c) si hay varios dispositivos activos, siempre funcionan como maestros
- X d) todas las respuestas anteriores son ciertas

Puntuación: **-0,33**

¿Cómo se almacenaría como palabra de 32 bits el número -128 en un sistema que utilice el criterio del extremo menor ("little endian")?

Usuar Profesor

io es

- a) posición 0: FF pos.1:FF pos.2: FF pos.3:
80
- ✓ • b) 0:80 1:FF 2:FF 3:FF
- c) 0:00 1:01 2: 00 3:80
- d) Ninguna de las anteriores

Puntuación: **1,00**

Si A=FF0Fh y B=0004h, el resultado de desplazar A a la derecha aritméticamente B veces es:

Usuar Profesor

io es

- a FFF0h
)
- X b 0FF0h
)
- c F0FFh
)
- d F0F0h
)

Puntuación: **-0,33**

Suponiendo que todos los registros inicialmente contienen el valor 0, ¿cuál es el valor de r1 tras la ejecución de la siguiente secuencia de instrucciones?

```
mov r1, #4  
mov r2, #3  
add r3, r1, r1  
sub r1, r3, r2  
mul r3, r1, r1
```

Usuar Profesor

io es

- a) 4
- b) 5
- c) 2
- 5
- d) 0

Puntuación: **1,00**

Un modo de vídeo de 1680 x 1050 píxeles y 4 canales de color por píxel (R, G, B, alpha) y 256 niveles de intensidad por canal (0 a 255), ocupa aproximadamente una memoria de:

Usuar Profesor

io es

- a) 1,7 MB
- b) 7 MB
- c) 54 MB
- d) 1,7 GB

Puntuación: **1,00**

Suponiendo que todos los registros inicialmente contienen el valor 0, ¿cuál es el valor de r1 tras la ejecución de la siguiente secuencia de instrucciones?

```
mov r1, #4  
mov r2, #3  
add r3, r1, r1  
sub r1, r3, r2  
mul r3, r1, r1
```

Usuar Profesor

io es

- a) 4
- b) 5
- c) 25
- d) 0

Puntuación: **1,00**

Una máquina superescalar es aquella que:

Usuar Profesor

io es

- a) basa su funcionamiento en la segmentación software como forma de incrementar el paralelismo.
- b) las instrucciones tienen un campo por cada unidad funcional al realizarse varias operaciones por instrucción.
- c) emite simultáneamente múltiples instrucciones por ciclo de reloj, por ejemplo, una entera y otra de coma flotante.
- X d) ninguna respuesta de las anteriores es correcta.

Puntuación: **-0,33**

¿Cuál de las siguientes afirmaciones es incorrecta?

Usuar Profesor

io es

- a) En el direccionamiento inmediato el dato se encuentra en la propia instrucción
- b) En el direccionamiento implícito no se indica la ubicación del operando
- ✓ • c) El direccionamiento indirecto indica la dirección del operando.
- d) El direccionamiento indexado es útil para manejo de vectores.

Las arquitecturas de acumulador se caracterizan por:

Usuar Profesor

io es

- X
- a) tráfico reducido entre procesador y memoria.
 - b) instrucciones muy largas.
 - c) diseño simple del procesador.
 - d) facilidad de generación de código eficiente para los compiladores.

Puntuación: **-0,33**

En el 8086, la dirección efectiva de la cabecera de pila vendrá dada por:

Usuar Profesor

io es

- ✓
- a) SS * 10h +
SP
 - b) BP

c) SS * 10h +

BP

d) SP

Puntuación: **1,00**

¿Qué modelo de programa se ejecuta en las arquitecturas de Von Neumann?

Usuar Profesor

io es

a) nanoprogramado

b) cableado

c) micropogramado

✓ • d) almacenado

Puntuación: **1,00**