

Introducción ARM

Arquitectura de Computadores y Ensambladores 1

Escuela de Ingenieria de Ciencias Y Sistemas

Facultad de Ingenieria

Universidad de San Carlos de Guatemala

Agenda



RECORDATORIOS



¿Qué Es Ensamblador?



Vocabulario Básico y Tipos De Datos



Registros Básicos Del Procesador



Instalación De Herramientas En Linux



COMPETENCIA(S) Que desarrollaremos



Competencia Principal:

- Analizar y comprender la arquitectura ARM y su lenguaje ensamblador para desarrollar programas eficientes a nivel de hardware.

Competencias Específicas:

- Identificar los componentes fundamentales de la arquitectura ARM
- Utilizar el vocabulario técnico apropiado del lenguaje ensamblador
- Manipular registros del procesador para operaciones básicas
- Configurar entornos de desarrollo para programación ARM en Linux

¿QUÉ ES ENSAMBLADOR?

```
0x00000000 90      nop
0x00000001 90      nop
0x00000002 6800009c00 push 0x9c0000
0x00000007 e8c7ace37b call 0x7be3acd3
0x7be3acd3(unk)
0x0000000c bb04009c00 mov ebx, 0x9c0004
0x00000011 8903      mov [ebx], eax
0x00000013 e81903f47b call 0x7bf40331
0x7bf40331()
0x00000018 bb08009c00 mov ebx, 0x9c0008
0x0000001d 8903      mov [ebx], eax
0x0000001f bb00009c00 mov ebx, 0x9c0000
0x00000024 c60300      mov byte [ebx], 0x0
-> 0x00000027 68e8030000 push 0x3e8 ; 0x000000
0x0000002c e81124e37b call 0x7be32442
0x7be32442(unk)
0x00000031 ebf4      jmp 0x100000027
0x00000033 90      nop
0x00000034 ff      invalid
0x00000035 ff      invalid
0x00000036 ff      invalid
0x00000037 ff      invalid
```

- Lenguaje de programación de bajo nivel que utiliza instrucciones fáciles de recordar
- Relación con el código máquina: Correspondencia directa entre instrucciones y código binario
- Ventajas:
 - Control total del hardware
 - eficiencia en recursos
 - velocidad de ejecución
- Desventajas:
 - Complejidad
 - dependencia de arquitectura
 - tiempo de desarrollo

VOCABULARIO BÁSICO Y TIPOS DE DATOS

Instrucciones:

ADD, SUB, MOV, LDR, STR

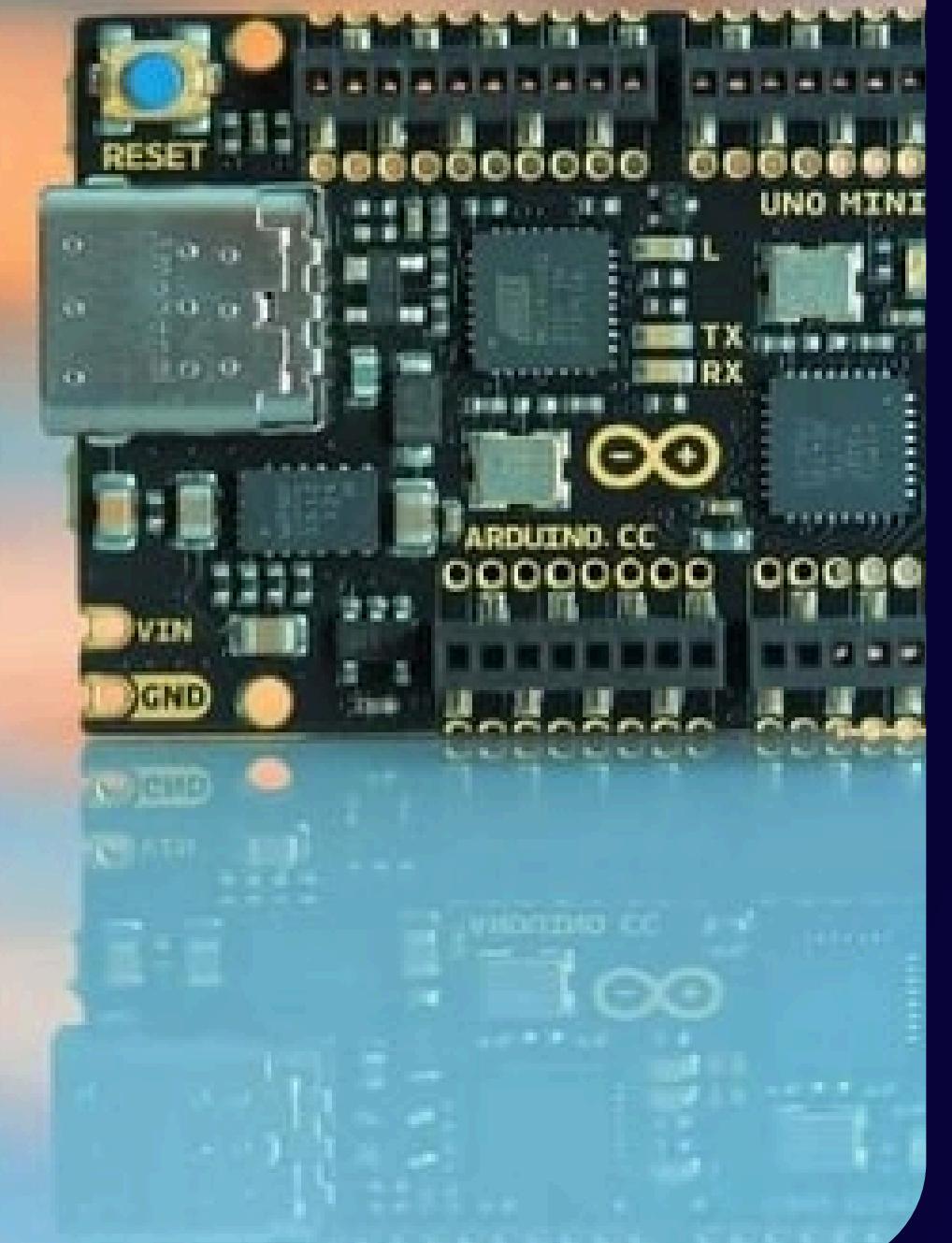
Operандos: Registros, constantes, direcciones de memoria

Tipos de datos ARM:

- Byte (8 bits)
- Halfword (16 bits)
- Word (32 bits)
- Doubleword (64 bits)

REGISTROS BÁSICOS DEL PROCESADOR

- Registros de propósito general:
R0-R12
- Registros especiales:
 - R13 (SP): Stack Pointer
 - R14 (LR): Link Register
 - R15 (PC): Program Counter
- Registro de estado:
CPSR (Current Program Status Register)



INSTALACIÓN DE HERRAMIENTAS EN LINUX

- Compilador cruzado: arm-linux-gnueabi-gcc
- Ensamblador: arm-linux-gnueabi-as
- Enlazador: arm-linux-gnueabi-ld
- Emulador: QEMU para ARM



EJEMPLO 1

SUMA SIMPLE

```
.text  
.global _start  
  
_start:  
    mov r0, #5      @ Cargar 5 en r0  
    mov r1, #3      @ Cargar 3 en r1  
    add r2, r0, r1  @ r2 = r0 + r1
```

EJEMPLO 2

USO DE MEMORIA

```
ldr r0, =numero      @ Cargar dirección  
ldr r1, [r0]          @ Cargar valor desde memoria  
str r1, [r0, #4]     @ Guardar en memoria
```

CONCEPTOS CLAVE APRENDIDOS

Conceptos Fundamentales:

- ARM:

Arquitectura RISC de 32 bits ampliamente utilizada

- Registros:

Memoria de alta velocidad dentro del procesador

- Direccionamiento:

Métodos para acceder a datos en memoria

Puntos Clave:

- ARM utiliza arquitectura load/store
- Todas las instrucciones son de 32 bits
- Ejecución condicional disponible en todas las instrucciones
- Barrel shifter integrado para operaciones eficientes

VALORES Y ACTITUDES

VALOR: EXCELENCIA

Aplicación Práctica:

- Escribir código ensamblador limpio y bien documentado
- Optimizar algoritmos para máximo rendimiento
- Buscar siempre la solución más eficiente

VALOR: RESPONSABILIDAD

Aplicación Práctica:

- Gestionar correctamente los recursos del sistema
- Validar entradas para evitar errores críticos
- Documentar el código para futuros desarrolladores

VALOR: COMPROMISO

Aplicación Práctica:

- Dedicar tiempo suficiente para dominar conceptos complejos
- Practicar regularmente con ejercicios de programación
- Mantenerse actualizado con nuevas versiones de ARM

REFERENCIAS

1. Yiu, Joseph. "The Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors." Newnes, 2013.
2. Sloss, Andrew N. "ARM System Developer's Guide." Morgan Kaufmann, 2004.
3. Knaggs, Peter. "ARM Assembly Language Programming." CreateSpace, 2014.

Recursos Web:

ARM Developer Documentation: <https://developer.arm.com/>

ARM Assembly Language Guide:

<https://www.ic.unicamp.br/~pannain/mc404/>

Linux ARM Cross-compilation: <https://wiki.ubuntu.com/ARM/>



¡GRACIAS POR SU ATENCION!

DUDAS



Recuerda que tenemos nuestro foro semanal donde
puedes consultar cualquier duda que te surja en la
semana