



Universidad Autónoma de Baja California
Facultad de Ingeniería Mexicali
Ingeniería en Computación

Reporte de Práctica: #XX
Título de la Práctica

Organización y Arquitectura de Computadoras
Ing. Felix Arredondo Santamaria

Moya Monreal Erick Anselmo — 1110604

Fecha de entrega:
DD/FEBRERO/2026

Mexicali, Baja California
31 de enero de 2026

Índice

1. Introducción	2
2. Problemática	2
3. Justificación	2
4. Objetivos de la práctica	2
5. Marco teórico	2
6. Procedimiento	2
6.1. Ejemplo de Formato de Instrucción	3
6.2. Tabla de Estado de Registros	3
6.3. Tabla de Formato de Instrucción	3
6.4. Código en Ensamblador	3
6.5. Código en C	4
6.6. Código Mixto C con Assembly	4
7. Resultados con pruebas	5
7.1. Configuración de Visual Studio para Capturas	5
8. Conclusiones	5
A. Repositorio de código	7

1. Introducción

Escribe aquí el contexto general de la práctica y un resumen de lo que se abordará.

2. Problemática

Describe el reto técnico o el problema específico que se busca resolver en esta sesión de laboratorio.

3. Justificación

Explica por qué es importante resolver esta problemática y cómo ayuda a entender la arquitectura de computadoras.

4. Objetivos de la práctica

- **General:** Diseñar e implementar...
- **Específicos:**
 - Manipular registros del procesador...
 - Implementar lógica en ensamblador inline...

5. Marco teórico

Conceptos clave como: Pila de registros, FPU, direccionamiento indirecto, etc.

6. Procedimiento

Describe paso a paso lo realizado. Puedes incluir diagramas de flujo o bloques.

La instrucción MOV mueve datos del registro **EAX** al registro **EBX**. El valor **0x1A2B** en binario es **0001101000101011**.

6.1. Ejemplo de Formato de Instrucción

La instrucción completa sería: MOV EAX, 0x1A2B

6.2. Tabla de Estado de Registros

A continuación se muestra el estado de los registros antes y después de la ejecución:

Tabla 1: Estado de registros antes y después de la ejecución.

Antes		Después	
Registro	Valor	Registro	Valor
EAX	0x00000000	EAX	0x0000001A
EBX	0x00000000	EBX	0x0000002B
ECX	0x00000000	ECX	0x00000000
EDX	0x00000000	EDX	0x00000000

6.3. Tabla de Formato de Instrucción

Tabla 2: Formato de instrucción MOV en x86.

Bits	31-24	23-16	15-8	7-0
Opcode	1000 1001	ModR/M	Immediate	
Hex	89	C3	00	

6.4. Código en Ensamblador

```

1 section .data
2     msg db 'Hola Mundo', 0xA
3     len equ $ - msg
4
5 section .text
6     global _start
7
8 _start:
9     ; Escribir mensaje
10    mov eax, 4           ; sys_write
11    mov ebx, 1           ; stdout
12    mov ecx, msg         ; mensaje
13    mov edx, len         ; longitud
14    int 0x80             ; llamada al sistema

```

```
15
16     ; Salir
17     mov eax, 1           ; sys_exit
18     xor ebx, ebx        ; codigo 0
19     int 0x80
```

Listing 1: Ejemplo de código Assembly x86

6.5. Código en C

```
1 #include <stdio.h>
2
3 int main() {
4     int resultado;
5     int a = 5, b = 3;
6
7     // Ensamblador inline
8     __asm {
9         mov eax, a
10        add eax, b
11        mov resultado, eax
12    }
13
14    printf("Resultado: %d\n", resultado);
15    return 0;
16 }
```

Listing 2: Integración de ensamblador inline en C

6.6. Código Mixto C con Assembly

```
1 int suma_asm(int x, int y) {
2     int res;
3     __asm {
4         mov eax, x
5         add eax, y
6         mov res, eax
7     }
8     return res;
9 }
```

Listing 3: Ejemplo de función con assembly embebido

7. Resultados con pruebas

Inserta aquí capturas de pantalla de la ejecución, tablas de datos o fotos del hardware si aplica.

7.1. Configuración de Visual Studio para Capturas

Para obtener capturas óptimas del debugger en Visual Studio:

- Activar vista de registros: `Debug` → `Windows` → `Registers`
- Mostrar desensamblado: `Debug` → `Windows` → `Disassembly`
- Vista de memoria: `Debug` → `Windows` → `Memory`
- Asegurar que los puntos de interrupción estén visibles en la captura

8. Conclusiones

Reflexiones finales sobre el aprendizaje obtenido y el cumplimiento de los objetivos.

Referencias

- [1] Irvine, K. (2019). *Assembly Language for x86 Processors*. Pearson.
- [2] Patterson, D. A., & Hennessy, J. L. (2017). *Computer Organization and Design*. Morgan Kaufmann.

A. Repositorio de código

El código completo de las prácticas y programas utilizados se encuentra disponible en el siguiente repositorio de GitHub:

<https://github.com/usuario/arquitectura-computadoras-practica-XX>