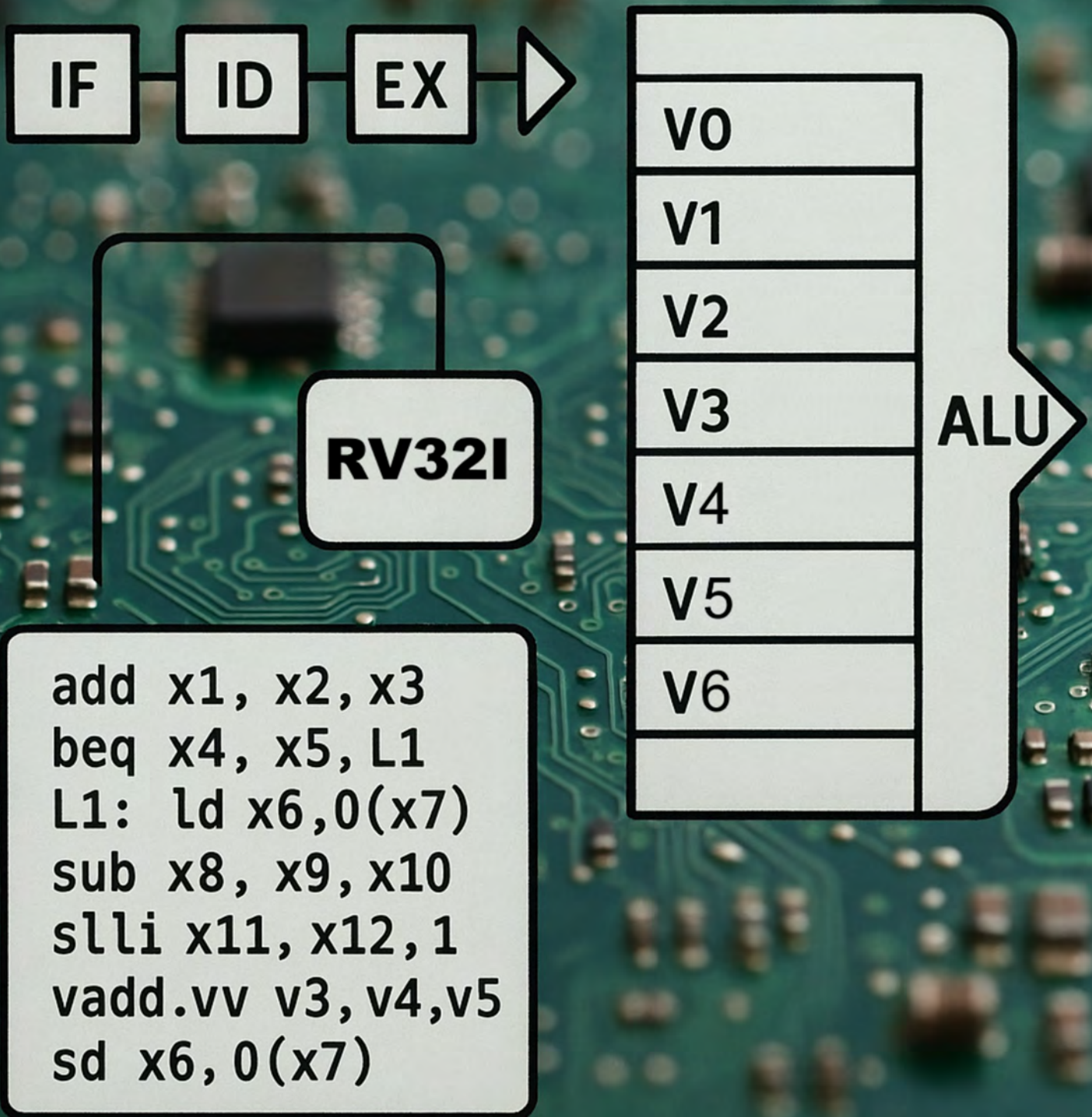


Programación en ensamblador RISC-V

Prácticas con Ripes y GNU/Linux



Francisco Charte Ojeda

Programación en ensamblador RISC-V

Prácticas con Ripes y GNU/Linux

Francisco Charte Ojeda
<francisco@fcharte.com>

Programación en ensamblador RISC-V

Prácticas con Ripes y GNU/Linux



Torre de Babel - fcharte.com

ISBN: 978-84-09-76297-2

Materia IBIC: UYFL : Lenguajes Assembly/ensambladores

Número de edición: 1

Fecha de edición: 01/09/2025

Copyright

Francisco Charte Ojeda, 2023-2025

Publicado en Jaén. Septiembre de 2025

Economía lingüística

A lo largo de este documento el autor ha procurado emplear lenguaje inclusivo siempre que ha sido posible, pero, por economía lingüística, ha de saberse que los sustantivos y determinantes masculinos no se emplean solo para referirse a personas de ese sexo, sino que han de entenderse como genéricos y, por tanto, que designan a todas las personas sin distinción de sexo. Así, para referirse a un individuo del colectivo del *profesorado* se usa la forma *el profesor* que hay que hacer extensiva a *el profesor o la profesora*.

This document was typeset with \LaTeX using the [kaobook](https://www.ctan.org/pkg/kaobook) class.

Índice general

Índice general	v
0. Introducción	1
0.1. Contenidos y estructura	1
0.2. La elección de RISC-V	2
0.3. RISC y CISC	3
0.4. Entorno de trabajo	3
0.5. Hardware RISC-V	5
0.6. Modos de operación del procesador	7
1. Introducción a RISC-V y Ripes	9
1.1. Planificación	9
1.2. Introducción	9
1.2.1. Banco de registros	9
1.2.2. Conjunto de instrucciones	10
1.2.3. Seudoinstrucciones	10
1.3. La herramienta Ripes	11
1.3.1. Descarga e instalación	11
1.3.2. La interfaz de Ripes	12
Editor de registros	13
Editor de código	13
Controles de ejecución	14
La consola	14
Guardar y cargar programas	15
1.4. Instrucciones aritméticas	15
1.5. Carga y almacenamiento de datos	15
1.5.1. Espacio de direccionamiento y mapa de memoria	15
1.5.2. Leer y escribir datos en memoria	16
1.5.3. Examinar la memoria en Ripes	17
1.6. Modos de direccionamiento	17
1.6.1. Formato general de las instrucciones RV32I	17
1.6.2. Instrucciones tipo R	18
1.6.3. Instrucciones tipo I y U	18
1.6.4. Instrucciones tipo S	19
1.7. Aspectos avanzados	19
1.7.1. Asignación de una dirección a un registro	19
1.7.2. Uso del puntero global	20
1.8. Ejercicios propuestos	21
2. Bucles y condicionales en RISC-V	23
2.1. Planificación	23
2.2. Introducción	23
2.3. Instrucciones de salto	23
2.3.1. Saltos condicionales	24
2.3.2. Saltos incondicionales	25
2.3.3. Llamadas a subrutinas	25
2.4. Otras instrucciones de la arquitectura RV32IM	26
2.4.1. Instrucciones lógicas	26
2.4.2. Instrucciones de comparación	26
2.4.3. Otras seudoinstrucciones	27
2.5. Aspectos avanzados	27
2.5.1. Almacenamiento temporal de datos en la pila	27
2.5.2. La pila para transferir parámetros a funciones	28
2.5.3. Funciones recursivas y la pila	29
2.6. Ejercicios propuestos	30
3. Segmentación del cauce y riesgos de datos	33
3.1. Planificación	33
3.2. Introducción	33

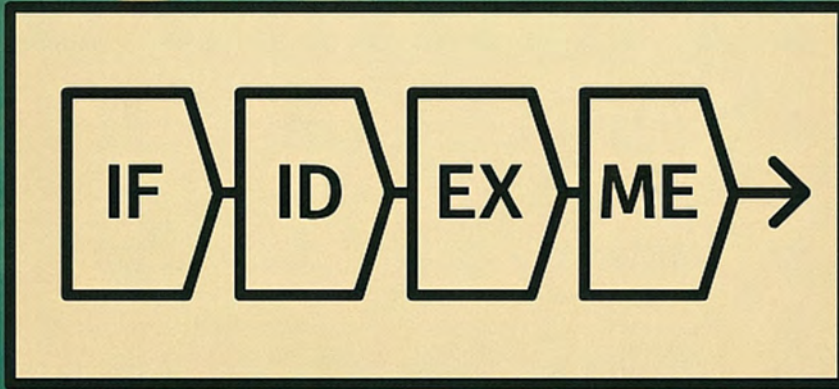
3.3.	Configuración de procesador	33
3.3.1.	Selección de la configuración	33
3.3.2.	Vista del cauce de ejecución	34
3.4.	Determinar mejoras de rendimiento	35
3.4.1.	Ciclos de un programa en una configuración sin segmentación	35
3.4.2.	Ciclos del programa en otras configuraciones	36
3.5.	Riesgos de datos	36
3.5.1.	Problemas de la segmentación	36
3.5.2.	Etapas de ejecución y cronograma	37
3.5.3.	Análisis de la ejecución en el cronograma	38
3.5.4.	Introducción manual de paradas	39
3.5.5.	Detección automática de riesgos	40
3.5.6.	Adelantamiento de datos	41
3.5.7.	Reordenación de código	42
3.6.	Comparativa de rendimientos	43
3.7.	Aspectos avanzados	44
3.7.1.	Registros de control y estado	45
3.7.2.	Lectura de contadores de rendimiento	45
3.8.	Ejercicios propuestos	45
4.	Segmentación del cauce y riesgos de control	47
4.1.	Planificación	47
4.2.	Introducción	47
4.3.	Instrucciones de salto	47
4.3.1.	Cómputo de la dirección de salto	48
4.3.2.	Evaluación de la condición de salto	49
4.4.	Riesgos de control	49
4.5.	¿Cómo reducir los riesgos de control?	51
4.5.1.	Desenrollado de bucles	52
4.5.2.	Desenrollado y reordenado	52
4.5.3.	Comparativa de rendimientos	53
4.6.	Aspectos avanzados	53
4.7.	Ejercicios propuestos	54
5.	Arquitectura superescalar	55
5.1.	Planificación	55
5.2.	Introducción	55
5.3.	Configuración del procesador	55
5.4.	Arquitectura superescalar	56
5.4.1.	Planificación estática	56
5.5.	La arquitectura superescalar de Ripes	57
5.5.1.	Elementos en el cauce de ejecución	57
5.5.2.	Tipos de riesgos en el cauce superescalar	58
5.5.3.	Estructura del cronograma	59
5.6.	Preparación del código	59
5.6.1.	Programa inicial	60
5.6.2.	Desenrollado del código	60
5.6.3.	Identificar riesgos de planificación	61
5.6.4.	Reordenar las instrucciones	61
5.6.5.	Precauciones a considerar	62
5.7.	Aspectos avanzados	63
5.8.	Ejercicios propuestos	64
6.	Memoria caché	67
6.1.	Planificación	67
6.2.	Introducción	67
6.3.	Cómo estimar el rendimiento de la memoria	67
6.4.	La vista caché de Ripes	68
6.4.1.	Terminología	68
6.4.2.	Configuración de la caché de datos	69

6.4.3.	Interpretación del diagrama de memoria	69
6.4.4.	Gráfica de rendimiento y estadísticas	71
6.5.	La caché en accesos de lectura desde memoria	71
6.5.1.	Sumar los elementos de una matriz	71
6.5.2.	Rendimiento según la configuración de la caché	72
6.5.3.	Patrón de accesos a memoria e impacto en la caché	73
6.5.4.	Estrategia de reemplazo para caché asociativa	75
6.5.5.	Comparativa de rendimientos	75
6.6.	La caché en accesos de escritura en memoria	76
6.6.1.	Cambios en el código	76
6.6.2.	Rendimiento según la configuración de escritura	77
6.7.	Ejercicios propuestos	77
7.	GNU/Linux en RISC-V	81
7.1.	Planificación	81
7.2.	Simuladores, emuladores y hardware RISC-V	81
7.3.	Linux en un SBC RISC-V	82
7.3.1.	Características del SBC	82
7.3.2.	Preparación de la imagen del sistema	83
7.3.3.	Primer inicio	85
7.3.4.	Configuración de red y acceso por SSH	86
7.4.	Virtualización con QEMU	87
7.4.1.	Instalación de QEMU	87
7.4.2.	Descarga del gestor de arranque y de Ubuntu	88
7.4.3.	Configuración del inicio de QEMU	88
7.4.4.	Acceso a la máquina virtual por SSH	89
7.5.	Programar en ensamblador RISC-V en Linux	90
7.5.1.	Instalación de las herramientas de desarrollo	90
7.5.2.	Comprobar las extensiones RISC-V que soporta la CPU	91
7.5.3.	Primer programa RISC-V en Linux	92
7.5.4.	Ensamblar y enlazar el ejecutable	93
7.5.5.	Examinar el contenido del ejecutable	93
7.5.6.	Depuración del programa	94
7.6.	Convenciones a seguir	96
7.6.1.	Inicialización del puntero global	96
	Direcciones de los símbolos	97
	Uso del puntero global en el código	98
	Depuración del programa	98
	Cómo inicializar el puntero global	99
7.6.2.	Acceso a los servicios de GNU/Linux	100
	Códigos de los servicios	100
	Parámetros de los servicios	101
7.7.	Medir el tiempo de ejecución	102
7.7.1.	Código del programa	102
	Segmento de datos	103
	Cuerpo principal del programa	104
	La función principal	104
	Toma de tiempos	104
	Cálculo del tiempo transcurrido	104
	Conversión de número a cadena	104
	Enviar una cadena a la consola	105
	Finalización	105
7.7.2.	Ejecución y depuración	105
7.8.	Ejercicios propuestos	107
8.	Operaciones con números en punto flotante	109
8.1.	Planificación	109
8.2.	Introducción	109
8.3.	Obtener valores en punto flotante	110
8.3.1.	Carga y almacenamiento desde memoria	111

8.3.2.	Directivas para datos en punto flotante	111
8.3.3.	Conversiones entero-flotante	111
8.4.	Operaciones básicas	112
8.4.1.	Instrucciones aritméticas	112
8.4.2.	Instrucciones de comparación	113
8.5.	Operaciones adicionales	115
8.5.1.	Instrucciones aritméticas	115
8.5.2.	Instrucciones de clasificación	116
8.6.	Aspectos avanzados	118
8.6.1.	Indicadores de excepciones	118
8.6.2.	Estrategias de redondeo	120
8.7.	Ejercicios propuestos	121
9.	Operaciones con datos vectoriales	123
9.1.	Planificación	123
9.2.	Introducción	123
9.3.	Registros vectoriales	124
9.3.1.	Denominación y tamaño de los registros	124
9.3.2.	Registros de estado y control vectorial	125
9.3.3.	Visualización de los registros en el depurador	126
9.4.	Procedimiento de trabajo	126
9.4.1.	Parámetros de configuración	126
9.4.2.	Carga y almacenamiento de datos	128
9.4.3.	Operaciones básicas sobre vectores	128
9.4.4.	Ciclo de operación sobre un vector completo	128
9.4.5.	Opciones de compilación específicas	129
9.4.6.	Alineación de datos en memoria	130
9.5.	Tipos de operaciones	131
9.5.1.	Operaciones entre vectores	132
9.5.2.	Operaciones con escalares y valores inmediatos	132
9.5.3.	Operaciones en punto flotante	132
9.5.4.	Operaciones de reducción	134
9.6.	Trabajo con vectores grandes	134
9.6.1.	Ajuste del parámetro LMUL	135
9.6.2.	Cómo evaluar la diferencia de rendimiento	135
9.7.	Aspectos avanzados	137
9.7.1.	Enmascaramiento	137
9.7.2.	Acceso a elementos no contiguos en memoria	139
9.7.3.	Aritmética de punto fijo	139
9.7.4.	Permutaciones de elementos	140
9.8.	Ejercicios propuestos	140
10.	Operaciones de entrada y salida	143
10.1.	Planificación	143
10.2.	Introducción	143
10.3.	Dispositivos de E/S en Ripes	143
10.3.1.	Configuración de un dispositivo	144
10.3.2.	Eliminación de dispositivos	144
10.4.	Entrada/salida mapeada	145
10.4.1.	Direcciones de los puertos de E/S	145
10.4.2.	Uso de la matriz de ledes	146
10.4.3.	Panel de microinterruptores	147
10.4.4.	Conjunto de pulsadores	147
10.5.	E/S en hardware físico	148
10.5.1.	E/S en microcontroladores	148
10.5.2.	E/S en ordenadores	150
10.6.	Ejercicios propuestos	151
11.	Interfaz entre ensamblador y C	153
11.1.	Planificación	153

11.2.	Introducción	153
11.3.	La interfaz binaria para aplicaciones	154
11.3.1.	Detectar y especificar la ABI	154
11.3.2.	Responsabilidad del código que hace la llamada	155
11.3.3.	Responsabilidad del código al que se llama	155
11.3.4.	Responsabilidades comunes	156
11.3.5.	Transferencia de parámetros en las llamadas	156
11.4.	Ensamblador embebido en código C	157
11.4.1.	Sintaxis extendida del comando asm	157
11.4.2.	Examen del código producido por el compilador	159
11.4.3.	Optimización de la compilación	160
11.5.	Invocar a funciones ensamblador desde C	161
11.5.1.	Funciones ensamblador públicas	162
11.5.2.	Prototipo en C de la función externa	162
11.5.3.	Ensamblado, compilación y enlace	163
11.6.	Invocar a funciones C desde ensamblador	163
11.6.1.	Módulo con funciones C	164
11.6.2.	Estructura del programa ensamblador	164
11.6.3.	Listas de parámetros variables	165
11.7.	Aspectos avanzados	166
11.7.1.	Creación de una librería estática	166
11.7.2.	Preparar el archivo de cabecera	167
11.7.3.	Uso de la librería	168
11.8.	Ejercicios propuestos	168
A.	Soluciones a ejercicios	171
A.1.	Introducción a RISC-V y Ripes	171
A.1.1.	Ejercicio 1.2	171
A.1.2.	Ejercicio 1.3	171
A.1.3.	Ejercicio 1.4	171
A.1.4.	Ejercicio 1.5	171
A.1.5.	Ejercicio 1.6	171
A.1.6.	Ejercicio 1.7	172
A.1.7.	Ejercicio 1.8	172
A.1.8.	Ejercicio 1.9	172
A.1.9.	Ejercicio 1.10	172
A.2.	Bucles y condicionales en RISC-V	172
A.2.1.	Ejercicio 2.1	172
A.2.2.	Ejercicio 2.2	173
A.2.3.	Ejercicio 2.3	173
A.2.4.	Ejercicio 2.4	173
A.2.5.	Ejercicio 2.5	173
A.2.6.	Ejercicio 2.6	174
A.2.7.	Ejercicio 2.7	174
A.2.8.	Ejercicio 2.8	174
A.2.9.	Ejercicio 2.9	175
A.2.10.	Ejercicio 2.10	176
A.2.11.	Ejercicio 2.11	176
A.2.12.	Ejercicio 2.12	176
A.2.13.	Ejercicio 2.13	177
A.2.14.	Ejercicio 2.14	177
A.2.15.	Ejercicio 2.15	178
A.2.16.	Ejercicio 2.16	178
A.2.17.	Ejercicio 2.17 - Recursivo	179
A.2.18.	Ejercicio 2.17 - Iterativo	180
A.2.19.	Ejercicio 2.18	180
A.2.20.	Ejercicio 2.19	181
A.3.	Segmentación del cauce y riesgos de datos	181
A.3.1.	Ejercicio 3.1	181
A.3.2.	Ejercicio 3.4	181

A.4. Segmentación del cauce y riesgos de control	181
A.4.1. Ejercicio 4.1	181
A.4.2. Ejercicio 4.2	182
A.4.3. Ejercicio 4.3	182
A.4.4. Ejercicio 4.4	182
A.5. Arquitectura superescalar	183
A.5.1. Ejercicio 5.1	183
A.6. Memoria caché	183
A.6.1. Ejercicio 6.1	183
A.6.2. Ejercicio 6.2	184
A.6.3. Ejercicio 6.3	184
A.6.4. Ejercicio 6.4	184
A.7. RISC-V en GNU/Linux	185
A.7.1. Ejercicio 7.2	185
A.8. Operaciones con números en punto flotante	186
A.8.1. Ejercicio 8.1	186
A.8.2. Ejercicio 8.5	188
A.9. Operaciones con datos vectoriales	189
A.9.1. Ejercicio 9.1	189
A.9.2. Ejercicio 9.2	190
A.9.3. Ejercicio 9.3	191
A.9.4. Ejercicio 9.4	191
A.9.5. Ejercicio 9.5	192
A.10. Operaciones de entrada y salida	193
A.10.1. Ejercicio 10.1	193
A.10.2. Ejercicio 10.2	194
A.10.3. Ejercicio 10.3	195
A.10.4. Ejercicio 10.5	196
A.11. Interfaz entre ensamblador y C	197
A.11.1. Ejercicio 11.1	197
A.11.2. Ejercicio 11.4	197
A.11.3. Ejercicio 11.5	198
Bibliografía	199
Índice alfabético	201



```
add s0, s1, s2
lw s3, 0(s0)
sub s4, s0, s3
bge s5, s6, label
slli s4, s4, 1
mul s7, s4, s6
jalr ra, s7, -24
andi s8, s8, 255
label:
vadd.vv v1, v2, v3
csrr s9, ustatus
blt s8, s9, -16
sw s10, 4(s3)
```

