

香港中文大學 The Chinese University of Hong Kong

# **CENG3420**

Lab 1-1: RISC-V Assembly Language Programing I

Chen BAI Department of Computer Science & Engineering Chinese University of Hong Kong cbai@cse.cuhk.edu.hk

Spring 2022

1 Introduction to Basic RISC-V Assembly Programing

### **2** RARS

3 System Service in RARS

4 Lab 1-1 Assignment

# Introduction to Basic RISC-V Assembly Programing

- We can manipulate 32 general purpose registers in assembly programming directly.
- We prefer using aliases to indicate registers.
- Instructions category
  - Load and store instructions
  - Bitwise instructions
  - Arithmetic instructions
  - Control transfer instructions
  - Pseudo instructions

Table:	Register	names	and	descriptions	

Register Names	ABI Names	Description			
x0	zero	Hard-wired zero			
x1	ra	Return address			
x2	sp	Stack pointer			
x3	gp	Global pointer			
x4	tp	Thread pointer			
x5	t0	Temporary / Alternate link register			
x6-7	t1 - t2	Temporary register			
x8	s0 / fp	Saved register / Frame pointer			
x9	s1	Saved register			
x10-11	a0-a1	Function argument / Return value registers			
x12-17	a2-a7	Function argument registers			
x18-27	s2-s11	Saved registers			
x28-31	t3-t6	Temporary registers			

#### Data types:

- All instructions are encoding in 32 bits
- Alias: byte (8 bits), halfword (2 bytes), word (4 bytes), double word (8 bytes)

#### Literals:

- numbers entered as is. e.g., 12 in decimal, and 0xC in hexadecimal
- characters enclosed in single quotes. e.g., 'b'
- strings enclosed in double quotes. e.g., "A string"

# Program Structure I

- Plain text file with data declarations, program code (name of file can be suffixed with *.asm*)
- Data declaration section is followed by program code section

### **Data Declarations**

- Identified with assembler directive .data
- Declares variable names used in program
- Storage allocated in main memory (e.g., RAM)
- <name>: .<datatype> <value>

# Program Structure II

## Code

- placed in section of text identified with assembler directive .text
- contains program code (instructions)
- starting point for code e.g. execution given label start:

### Comments

Anything following # on a line

#### The structure of an assembly program looks like this:

### Program outline

# Comment giving name of program and description

```
# Template.asm
```

# Bare-bones outline of RISC-V assembly language program

```
.globl _start
```

```
.data # variable declarations follow this line
    # ...
.text # instructions follow this line
```

```
_start: # indicates start of code
# ...
```

# End of program, leave a blank line afterwards is preferred

# An Example Program

1	.globl _start	We also and the second se
2		
3	.data	
4	welcome_msg: .asciz "Welcome to ENG3420!\n"	
5		
6	.text	
7	_start:	
8		
9	addi a0, x0, 1	
10		
11	la a1, welcome_msg	
12		
13	addi a2, x0, 21	
14		
15	addi a7, x0, 64	
16		
17		
18		

# An Example Program

it Execute									Control and Sta	atus	
									Registers	Floati	ng Point
Text Segment		,						ت <u>م</u>	Name	Number	Value
pt Address	Code	Basic			Sourc	e			zero	0	0x00000
	0x00100513addi x1		∋: addi aO, xO					*	ra	1	0×00000
	0x0fc10597auipc x		l: la al, welc	one_nsg					SD	2	0x7fffe
	0xffc58593addi x1								gp	3	0x10008
	0x01500613 addi x1								tp	4	0×00000
		7,x0,0x00000040 15		. 64					t0	5	0×00000
UXUU4UUU14 C	0×00000073ecall	1	7: ecall						t1	6	0x00000
									t2	7	0×00000
									s0	8	0x00000
									sl	9	0x00000
									a0	10	0x00000
								-	al	11	0×10010
									a2	12	0x00000
									a3	13	
Data Segment								ت <u>م</u> ا	a3 a4	14	0x00000 0x00000
Data Segment	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)		a3 a4 a5	14	0x00000
9	Value (+0) 0x00100513	Value (+4) 0x0fc10597	Value (+8) 0xffc56593	Value (+c) 0x01500613	Value (+10) 0x04000893	Value (+14) 0x00000073	Value (+18) 0x00000000	r 🛛	a3 a4 a5 a6	14 15 16	0x00000 0x00000 0x00000
Address								value (+1c)	a3 a4 a5 a6 a7	14 15 16 17	Gx 00000 Gx 00000 Gx 00000 Gx 00000
Address 0x00400000	0x00100513	0x0fc10597	0xffc58593	0x01500613	0x04000893	0x00000073	Gx 00000000	ut using (+1c) 0x00000000 ▲	a3 a4 a5 a6 a7 s2	14 15 16 17 18	0×00000 0×00000 0×00000 0×00000 0×00000
Address 0x00400000 0x00400020	0x00100513 0x00000000	0×0fc10597 0×00000000	0xffc58593 0x00000000	0x01500613 0x00000000	0x04000893 0x00000000	0x00000073 0x00000000	0x00000000 0x00000000	© <sup>×</sup> 2 <sup>3</sup> Value (+1c) 0x00000000 ▲ 0x00000000	a3 a4 a5 a6 a7 \$2 \$3	14 15 16 17 18 19	0x 00000 0x 00000 0x 00000 0x 00000 0x 00000 0x 00000 0x 00000
Address 0x00400000 0x00400020 0x00400020	0x00100513 0x00000000 0x00000000	0×0fc10597 0×00000000 0×00000000	0xffc58593 0x00000000 0x00000000	0x01500613 0x00000000 0x00000000	0x04000893 0x00000000 0x00000000	0×00000073 0×00000000 0×00000000	0×00000000 0×00000000 0×00000000		a3 a4 a5 a6 a7 s2 s3 s4	14 15 16 17 18 19 20	0x 00000 0x 00000 0x 00000 0x 00000 0x 00000 0x 00000 0x 00000
Address 0x00400000 0x00400020 0x00400040 0x00400060	0x00100513 0x00000000 0x00000000 0x00000000	0×0fc10597 0×00000000 0×00000000 0×00000000	0xffc58593 0x0000000 0x0000000 0x00000000	0x01500613 0x00000000 0x00000000 0x00000000	0x04000893 0x00000000 0x00000000 0x00000000 0x000000	0x00000073 0x00000000 0x00000000 0x00000000	0x0000000 0x0000000 0x0000000 0x0000000		a3 a4 a5 a6 a7 s2 s3 s4 s5	14 15 16 17 18 19 20 21	0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000
Address 0x00400000 0x00400020 0x00400040 0x00400060 0x00400060	0x00100513 0x00000000 0x00000000 0x00000000 0x000000	0x0fc10597 0x00000000 0x00000000 0x00000000 0x000000	0xffc58593 0x0000000 0x0000000 0x0000000 0x0000000	0x01500613 0x0000000 0x0000000 0x0000000 0x0000000	0x04000893 0x0000000 0x0000000 0x0000000 0x0000000	0x 00000073 0x 00000000 0x 00000000 0x 00000000 0x 00000000	0x00000000 0x00000000 0x00000000 0x000000	■ " □" Value (+1c) 0x00000000 0x00000000 0x00000000 0x00000000	a3 a4 a5 a6 a7 s2 s3 s3 s4 s5 s6	14 15 16 17 18 19 20 21 22	0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000
Address 0x00400000 0x00400020 0x00400040 0x00400060 0x00400080 0x00400080 0x00400080 0x00400080	0x00100513 0x00000000 0x00000000 0x000000000 0x000000	0x0fc10597 0x0000000 0x0000000 0x00000000 0x000000	0xffc58593 0x0000000 0x0000000 0x00000000 0x000000	0x01500613 0x00000000 0x00000000 0x00000000 0x000000	0x04000993 0x00000000 0x00000000 0x00000000 0x000000	0x6000073 0x0000000 0x0000000 0x0000000 0x0000000	0x 00000000 0x 00000000 0x 00000000 0x 00000000		a3 a4 a5 a6 a7 s2 s3 s4 s5 s6 s6 s7	14 15 16 17 18 19 20 21 22 22 23	0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000
Address 0x00400000 0x00400020 0x00400020 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060 0x00400000	0x00100513 0x00000000 0x00000000 0x00000000 0x000000	0x0fc10597 0x00000000 0x00000000 0x00000000 0x000000	0xffc58593 0x0000000 0x0000000 0x0000000 0x0000000	0x01500613 0x00000000 0x00000000 0x00000000 0x000000	0x04000893 0x00000000 0x000000000 0x00000000 0x000000	0x60060073 0x60060000 0x60060000 0x60060000 0x60060000 0x60060000 0x60060000 0x60060000 0x60060000 0x60060000	0x00000000 0x00000000 0x00000000 0x000000	a*         a*         a*           Value (+1)         6x00000000         a           0x00000000         0x00000000         a           0x00000000         0x00000000         a           0x00000000         0x00000000         a           0x00000000         0x00000000         a	a3 a4 a5 a6 a7 s2 s3 s4 s5 s6 s5 s6 s7 s8	14 15 16 17 18 19 20 21 22 23 23 24	0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000 0x00000
Address 0x00400000 0x00400020 0x00400040 0x00400080 0x00400080 0x00400080 0x00400080 0x00400080	0x00100513 0x00000000 0x00000000 0x000000000 0x000000	0x0fc10597 0x0000000 0x0000000 0x00000000 0x000000	0xffc58593 0x0000000 0x0000000 0x00000000 0x000000	0x01500613 0x00000000 0x00000000 0x00000000 0x000000	0x04000993 0x00000000 0x00000000 0x00000000 0x000000	0x6000073 0x0000000 0x0000000 0x0000000 0x0000000	0x 00000000 0x 00000000 0x 00000000 0x 00000000		03 04 05 07 02 03 04 05 05 05 07 08 09 00 00 00 00 00 00 00 00 00	14 15 16 17 18 19 20 21 22 23 24 25	0x 00000 0x 00000
Address 0x00400000 0x00400020 0x00400020 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060 0x00400000 0x004000000 0x00400100	0x00100513 0x00000000 0x00000000 0x00000000 0x000000	0x0fc10597 0x00000000 0x00000000 0x00000000 0x000000	0xffc58593 0x00000000 0x000000000 0x00000000 0x00000000	0x01500613 0x00000000 0x00000000 0x00000000 0x000000	0x0400093 0x00000000 0x00000000 0x00000000 0x000000	0x00000073 0x00000000 0x00000000 0x00000000 0x000000	0x 00000000 0x 00000000 0x 00000000 0x 00000000		a3 a4 a5 a6 a7 s2 s3 s4 s5 s5 s6 s7 s8 s9 s9 s10	14 15 16 17 18 19 20 21 22 23 24 23 24 25 26	0x 00000 0x 00000
Address 0x00400000 0x00400020 0x00400020 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060	0x00100513 0x00000000 0x00000000 0x000000000 0x000000	0x0fc10597 0x0000000 0x0000000 0x00000000 0x000000	0xffc58593 0x00000000 0x000000000 0x00000000 0x00000000	0x01500613 0x00000000 0x00000000 0x00000000 0x000000	0x04000893 0x00000000 0x000000000 0x00000000 0x000000	0x00000073 0x00000000 0x00000000 0x00000000 0x000000	0x 00000000 0x 00000000 0x 00000000 0x 00000000		a3       a4       a5       a6       a7       s2       s3       s6       s7       s6       s7       s8       s9       s10       s11	14 15 16 17 18 19 20 21 22 23 24 25 26 27	0x 00000 0x 00000
Address 0x00400000 0x00400020 0x00400020 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060 0x00400060	0x00100513 0x00000000 0x00000000 0x00000000 0x000000	0x0fc10597 0x00000000 0x00000000 0x00000000 0x000000	0xffc58593 0x00000000 0x000000000 0x00000000 0x00000000	0x01500613 0x00000000 0x00000000 0x00000000 0x000000	0x0400093 0x00000000 0x00000000 0x00000000 0x000000	0x00000073 0x00000000 0x00000000 0x00000000 0x000000	0x 00000000 0x 00000000 0x 00000000 0x 00000000		a3 a4 a5 a6 a7 s2 s3 s4 s5 s5 s6 s7 s8 s9 s9 s10	14 15 16 17 18 19 20 21 22 23 24 23 24 25 26 27 28	0x 00000 0x 00000
Address 0x00400000 0x00400020 0x00400060 0x00400060 0x00400060 0x00400080 0x00400080 0x00400080 0x00400080 0x00400080 0x00400080 0x00400080 0x00400800 0x00400080	0x00100513 0x0000000 0x00000000 0x00000000 0x000000	0x0fc10597 0x00000000 0x00000000 0x00000000 0x000000	0xffc58593 0x00000000 0x000000000 0x00000000 0x00000000	0x01500613 0x00000000 0x00000000 0x00000000 0x000000	0x0400093 0x00000000 0x00000000 0x00000000 0x000000	0x00000073 0x00000000 0x00000000 0x00000000 0x000000	0x 00000000 0x 00000000 0x 00000000 0x 00000000		a3       a4       a5       a6       a7       s2       s3       s4       s5       s6       s7       s8       s9       s10       s11       t3	14 15 16 17 18 19 20 21 22 23 24 25 26 27	0x 60000 0x 60000
Address 0x00400000 0x004000020 0x00400040 0x00400080 0x00400080 0x00400080 0x0040080 0x0040080 0x0040080 0x0040080 0x0040080 0x0040080 0x0040080 0x0040080 0x0040080 0x0040080 0x0040080 0x0040080 0x0040080 0x0040080 0x004000000 0x00400000 0x0040000 0x0040000 0x0040000 0x0040000 0x0040000 0x0040000 0x0040000 0x004000 0x004000 0x004000 0x004000 0x0040 0x0040 0	0x00100513 0x0000000 0x00000000 0x00000000 0x000000	0x0fc10597 0x00000000 0x00000000 0x00000000 0x000000	0xffc58593 0x00000000 0x000000000 0x00000000 0x00000000	0x01500613 0x00000000 0x00000000 0x00000000 0x000000	0x0400093 0x00000000 0x00000000 0x00000000 0x000000	0x00000073 0x00000000 0x00000000 0x00000000 0x000000	0x 00000000 0x 00000000 0x 00000000 0x 00000000		03       34       35       36       97       52       63       84       55       86       67       88       99       811       t3       t4	14 15 16 17 18 18 20 20 21 22 23 24 25 26 26 26 27 28 29	0x 60000 0x 00000 0x 60000 0x 60000

## Instructions Overview I

LA: The Load Address (la) loads the location address of the specified SYMBOL.

Syntax
la rd, SYMBOL
Usage
Usage
.data NumElements: .byte 6
.text <b>la</b> x5, NumElements # <i>assign addr[NumElements] to x5</i>
La Ko, Mandreneo " dobryn ddar[Mandreneo] co Ko

LI: The Load Immediate (LI) loads a register (rd) with an immediate value given in the instruction.





**li** x5,100 # assign 100 to x5

LD: The Load Double word (LD) instruction does the fetching of 64-bit value from memory and loads into the destination register (rd).

Syntax ld rd, offset(rs1) Usage **1d** x4, 1352(x9) # assign memory[x9+1352] to x4

SD: The Store Double word (SD) instruction does the copying of 64-bit value from register (rs2) and loads into the memory(rs1).

## Instructions Overview III

#### Syntax

sd rs2, offset(rs1)

#### Usage

**sd** x4, 1352(x9) # assign mem[x9+1352] to x4

SLL: Shift Logical Left (SLL) performs logical left on the value in register (rs1) by the shift amountheld in the register (rs2) and stores in (rd) register.

# Syntax sll rd, rs1, rs2 Usage

li x5, 4 # assign 4 to x5
li x3, 2 # assign 2 to x3
sll x1, x5, x3 # assign x5 << x3 to x1</pre>

SRL: Shift Logically Right (SRL) performs logical Right on the value in register (rs1) by the shift amount held in the register (rs2) and stores in (rd) register.

```
        Syntax

        srl rd, rs1, rs2

        Usage

        li x5, 1024 # assign 1024 to x5

        li x3, 2 # assign 2 to x3

        srl x1, x5, x3 # assign x5 >> x3 to x1
```

SLLI: Shift Logically Left Immediate (SLLI) performs logical left on the value in register (rs1) by the shift amount held in the register (imm) and stores in (rd) register.

Syntax
slli rd, rs1, imm
Usage
slli x1, x1, 3 # assign x1 << 3 to x1

SRLI: Shift Logically Right Immediate (SRLI) performs logical Right on the value in register (rs1) by the shift amount held in the register (imm) and stores in (rd) register.



#### srli rd, rs1, imm

### Usage

srli x1, x1, 1 # assign x1 >> 1 to x1

For more information about RISC-V instructions and assembly programing you can refer to:

- 1 Lecture slides and textbook.
- **RARS** Help: F1
- 3 https:

//github.com/riscv/riscv-asm-manual/blob/master/riscv-asm.md

#### 4 https:

//web.eecs.utk.edu/~smarz1/courses/ece356/notes/assembly/

# **RISC-V ISA Simulator – RARS**

- RARS is the RISC-V Assembler, Runtime and Simulator for RISC-V assembly language programs
- RARS supports RISC-V IMFDN ISA base (riscv32 & riscv64).
- **RARS** supports debugging using breakpoints like *ebreak*.
- **RARS** supports side by side comparison from psuedo-instruction to machine code with intermediate steps.
- You need Java environment to run RARS

Dowload it here: https://github.com/TheThirdOne/rars/releases/ download/continuous/rars\_f0c874c.jar Execute the command to start RARS: java -jar <rars jar path>

ile <u>E</u> dit	Ban šetīnajs Loois Help						
Edit E		-1	Registers	Floating Point	Control and Status		
testas				tame	Number		Value
			2 er o			0	0000000000000000000000000000000000000
88	# 12 iss/rvb4u/srliw.S 2	-18	YA.			1	0000000000000000000000000000000000000
89			63			2	0x00000007fffef:
90			0			3	0.00000000100080
	text		10			5	0,0000000000000000000000000000000000000
91			tl			6	0,0000000000000000000000000000000000000
92	.globl_start		t2			7	020000000000000000000000000000000000000
93	_start: hop		e0 e1			3	0.0000000000000000000000000000000000000
			*0			10	0.0000000000000000000000000000000000000
94			a1				0.0000000000000000000000000000000000000
96	f		2			12	0.0000000000000000000000000000000000000
96	# Arithmetic tests		a3			13	0.0000000000000000000000000000000000000
97			14			14	0000000000000000000000000000000000000
			10 10			15 16	000000000000000000000000000000000000
98			1			17	0-0000000000000000000000000000000000000
99	test_2: li x1, 0xffffffff80000000		12			18	0-0000000000000000000000000000000000000
00	arlig s14, s1, 0		s3			19	000000000000000000000000000000000000
			24			30	0.0000000000000000000000000000000000000
01	11 x7, 0xfffffff8000000		e5			21	000000000000000000000000000000000000
12	11 pp. 2		9/5			22	0=0000000000000000000000000000000000000
13	bue z14, z7, fail		17			24	0=0000000000000000000000000000000000000
	UIT AT AN INI		19			35	0+0000000000000000000000000000000000000
14			\$10			26	0000000000000000000000000000000000000
16	rest_3: li x], Oxffffffff80000000		\$11				0x0000000000000000000000000000000000000
06	srliw x14, x1, 1		13			28	0.4000000000000000000000000000000000000
			14			29	0.0000000000000000000000000000000000000
07	11 x7, 0x0000004000000		10			31	0,0000000000000000000000000000000000000
08	11 gp. 3		11				0.000000000040000
29	bme x14, x7, fall						
	ALT ALT ALL						
10							
11	rest_4: li w1, Owrfffffff80000000	- 1					
		-11					
ne: 100	Johannes 19 (v) Show Line Numbers						
-		- 1					
Mossin	s Ren 10						
		-1					
	Assembles: assembling 7: Unaver-shinist/TAUCHROAD2Vtools/test, ass						
	Fursing in F. Unterrebusis/IA/CHR50420/tools/test and lize 312 relams 2: BAMS does not recognize the .global directive. Ignored						
	Vernig in F. Marsardhunisr[IAA/EBBSH20](tool)(test, am line 318 column 2: MAMS down not recognize the .global directive. Ignored						
Clear	termble: operation completed successfully.						

t Execute									Registers	Floating Point	Control and Status	
Text Segment								ด้ ตั้		Name	Number	Value
	Code Basic				Source				242.0		0	0=0000000000000000000000000000000000000
	00000013 adds z0. z0. 0	93: start: pop			John Sector Sect				78.		1	0+000000007666
	90000015 Addi 10, 80, 0 90000017 lui 11, 0:fff800000		L 0x111111110000000					-	37		2	0+000000001000
	00000008 addix s1.s1.0		.,						0		3	0_0000000000000000000000000000000000000
	0000471h mrliv m14.ml.0	100; mliv x14, x1	0						10			0_0000000000000000000000000000000000000
	000003%7 lug x7. 0xfff800000	101: li x?, 0rffff							1		4	0_0000000000000000000000000000000000000
	00038399 addix x7.x7.0								12		7	0,0000000000000000000000000000000000000
0+00400013 0+0	00200193 addi #3. #0. 2	102: li m 2							80		3	0_0000000000000000000000000000000000000
0w0040001c 0x5	34771e63 hes s14. s7.0s00000358	103; bme x14, x7,	fail						11		2	0x00000000000
	000000%7 lui s1.0sff600000	105: test_3: li s	. 0 <del>xffffffff8000000</del>						a0		10	0.0000000000000000000000000000000000000
	0000309% adds #1.#1.0								a1		11	0::000000000000
	0104715 selis #14. #1.1	105: seliv x14, x1							12		12	0::000000000000000000000000000000000000
	\$2000337 Lui x7, 0x00040000	107: li x7, 0x0000	0004000000						a3		13	00000000000000
	00038395 addis x7, x7, 0								n4		14	0.0000000000000000000000000000000000000
	00300193 addi x3, x0, 3	105: li gp. 3							ső.		15	000000000000
	22771+63 has x14, x7, 0x00000334	109: bas x14, x7,							#0		16	0.0000000000000000000000000000000000000
	900000b7 lui s1, 0sffff90000	111: test_4: li s	. Cuffffffffff5000000					<b>T</b>	a2		17	0::000000000000
a second as									s2		18	0000000000000
									sJ		19	0=0000000000000000000000000000000000000
Data Segment								5 G	54		20	0000000000000
Address	Value (+0)	Value (+4)	Value (+0)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)	55		21	0.0000000000000000000000000000000000000
									<b>s</b> 5		22	0,0000000000000000000000000000000000000
0x10010000	0x00000000	0v00000000	0x00000000	0w00000000	0x00000000	Cw00000000	0.000000000	Cv00000000 +	87		23	0.0000000000000000000000000000000000000
0x10010020 0x10010040	0x0000000	0x00000000	000000000	Cw00000000 Cw000000000	000000000	Cw00000000 Dw000000000	0,00000000	Cu00000000	4		24	0_0000000000000000000000000000000000000
	0.00000000		0x00000000						a10		25	
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0.00000000	Cu00000000 Du00000000	0200000000	Cu00000000	#10 #11		20	0,0000000000000000000000000000000000000
0x100100#0	0x0000000	010000000	0x00000000	0x00000000	0x00000000	0x0000000	0,00000000	0x0000000	111		25	0+0000000000000000000000000000000000000
0x100100x0	0,00000000	0+000000000	0x00000000	0x00000000	0x00000000	0x00000000	0,00000000	0x00000000 =	114		20	0,0000000000000000000000000000000000000
0x100100+0	0.00000000	0=000000000	0.00000000	0x00000000	0-00000000	0x00000000	0,00000000	0x00000000	10		30	0.0000000000000000000000000000000000000
0x10010100	0.00000000	0+00000000	0x00000000	0x00000000	0+00000000	0x00000000	0,00000000	0x00000000	ed		31	0-0000000000000000000000000000000000000
0x10010120	0-00000000	0=00000000	0-00000000	0+00000000	0-00000000	0+00000000	0.00000000	0+00000000	80			0+00000000040
0x10010140	0.00000000	0+00000000	0=00000000	0+00000000	0-0000000	0x00000000	0+00000000	0x00000000				
0x10010100	0=00000000	0+00000000	0=00000000	0+00000000	0-00000000	0x00000000	0+00000000	0+00000000				
0x10010130	0=00000000	0+00000000	0=00000000	0+00000000	0=00000000	0x00000000	0=00000000	0x00000000				
0x100101x0	0-00000000	0+00000000	0-0000000	0~00000000	0-00000000	0~00000000	0+00000000	0+00000000				
								•				
			0x10010000 (.data)	Hexadecimal Addre	sses 🖃 Hexadecimal Vali	es 🗌 ASCI						
ssages Run I/O												
Assemble: wase	mbling F:\Research\misc\TAVCES	03420\tools\test.asm										
Ferning in Fri	(Research/misc/TA/(ER03420/test	alternt and line 212 and		and a she which it does not	in Immed							
	Renearch/mirc/TA/CEBG3420/test											
		airear and vine 310 cold	ma a. notar does not rec	Savas res - Global direct	tin. Allowane							
	ration completed successfally.	sytest and line bio cold	an 2: DAS cost lat 140	eguite tas groom direct	ive. ignored.							

RARS execution panel

## **RARS Basic Introduction**

Eile Edit	Bun šetings Icols Help					
	🖀 🕭 🚴 👌 👌 🗊 🖨 🥙 🎾 🖓 🖓 🔍 🔍 🔘 🚳 🚳 🚱 Run speed at max (so interaction)	ools panel				
Edit E	xecute		Registers	Floating Point	Control and Status	
testasn				Name	Number	Value
1	# 12 "isa/rv64ui/srliw.S" 2		2 er o		0	0000000000000000000000000000000000000
~~	* 12 158/1/081/5111#-5 D		YA.			0.0000000000000000000000000000000000000
89			9 D		2	
90			19		4	000000000000000000000000000000000000
91	text		60		5	
			t1 t2		6	020000000000000000000000000000000000000
92			12 10			0
93	_start: hop		1		2	
94			w0		10	
95	2		a1 a2		11	
	# Arithmetic tests	E	12		12	
96			14		14	
97	g		aŭ.		15	
98			ed.		16	
99	test 2: 1i x1. 0xfffffff8000000		12		17	
	srliw rl4, rl, 0		13		19	
100			14		20	
101	11 x7, 0xfffffff50000000		15		21	
102	11 m, 2		95		22	
103	bme x14, x7, fail		13		24	
			69		25	
104			\$10		26	
105	test_3: li x1, 0xffffffff80000000		s11 s2		27	
106	srliw x14, x1, 1		14		29	0x0000000000000000000000000000000000000
107	11 x7, 0x00000004000000		15		37	
108	1i gp. 3		66		31	0000000000000000000000000000000000000
			*			0.0000000000000000000000000000000000000
109	bne x14, x7, fail					
110						
111	rest_4: 11 s1, 0xfffffff8000000	×				
	Column: 18 Show Line Numbers			Regist	ers panel	
Message	es Ran IO					
	Assumble: assumbling F: URasearch/mise/TA/CERRO422/treals/test.mam					
	Furning in F \Lessarch\mise\TA\CENCHEV\tools\test.am line 212 column 2: 1485 does not recognize the .global directive. Tgeored					
	Furning in F. Merser Amine (MARMAN AND AND AND AND AND AND AND AND AND A	information panel				
Clear	Arcmalic operation registed conversionly.					

burblebook         20           burblebook <th></th> <th>100: mliw s14, s1, 0 101: li s7, 0sffffff 102: li gs, 2 103: bms s14, s7, fwi 105: test_3: li s1, 106: seliw s14, s1, 3 107: li s7, 0sc000000 100: li gs, 3 109: bms s14, s7, fai</th> <th>(#8000000     (****************************</th> <th>Value (+c)</th> <th>Seurce Seurce Value (+10) Concorrect</th> <th></th> <th></th> <th>0 0 0</th> <th>Name  res  res  res  res  res  res  res  r</th> <th>Namber 0 3 2 3 4 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4</th> <th>Value </th>		100: mliw s14, s1, 0 101: li s7, 0sffffff 102: li gs, 2 103: bms s14, s7, fwi 105: test_3: li s1, 106: seliw s14, s1, 3 107: li s7, 0sc000000 100: li gs, 3 109: bms s14, s7, fai	(#8000000     (****************************	Value (+c)	Seurce Seurce Value (+10) Concorrect			0 0 0	Name  res  res  res  res  res  res  res  r	Namber 0 3 2 3 4 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4	Value
000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           0004000000         00           0004000000         00           0004000000         00           0004000000         00           0004000000         00           00040000000         00           0004000000         00           00040000000         00           00040000000         00           00040000000         00           00040000000         00           000400000000         00           000400000000         00           0004000000000000         000000000000000000000000000000000000	Δ     Δ	<ol> <li>Test 2: [i s1, 10]</li> <li>Test 2: [i s2, 10]</li> <li>Test 3: 0</li> <li>Test 4: 0</li></ol>	1 1 1 1 1 1 1 1 1 1 1 1 1 1		Value (*10)	Value (+14)	Value (+18)	a* (3*	16           17           17           17           18           10           11           12           13           14           14           14           14           14           14           14           14           15           16           17           18           18           19           11           12           13           14	9 50 11 12 13 14 15 15 15 15 15 15 15 19 19 19 20 20 20 20	Control C
000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           000400000         00           0004000000         00           0004000000         00           0004000000         00           0004000000         00           0004000000         00           00040000000         00           0004000000         00           00040000000         00           00040000000         00           00040000000         00           00040000000         00           000400000000         00           000400000000         00           0004000000000000         000000000000000000000000000000000000	Δ     Δ	92.         test_2: [i s1, 100]           101:         11: s1, 0           101:         11: s1, 0           102:         11: gp. 2           103:         11: s1, 0           105:         test_2: 1: s1, 100           106:         test_2: 1: s1, 100           107:         test_2: 1: s1, 100           108:         test_2: 1: s1, 100           109:         test_3: 1: s1, 0           100:         test_3: 1: s1, 0           101:         test_4: s1, 1           101:         test_4: s1, 1           101:         test_4: s1, 1           102:         test_4: s1, 1           103:         test_4: s1, 1           104:         test_4: s1, 1           105:         test_4: s1, 1           106:         test_4: s1, 1           107:         test_4: s1, 1           108:         test_4: s1, 1           111:         test_4: 1: 1           111:         test_4: 1: 1           111:         test_4: 1: 1           112:         test_4: 1: 1           113:         test_4: 1: 1	1 1 1 1 1 1 1 1 1 1 1 1 1 1		Value (*10)	Value (+14)	Value (+18)	a* (3*	9 9 9 10 11 12 12 12 14 14 14 14 14 14 14 14 14 14 14 14 14	9 50 11 12 13 14 15 15 15 15 15 15 15 19 19 19 20 20 20 20	1000000004 20100000004 2000000004 2000000004 200000000
Dedeboors         0:           Dedeboors         0: <td>0.4000011/1.01 a. 1. 6.4789000 0.4000011/1.01 a. 1. 6.1 0.0000011/1.01 a. 1. 6.1 0.0000011/1.01 a. 1. 6.1 0.0000010101 a. 1. 6.1 0.000000101 a. 1. 6.1 0.000001011 a. 1. 6.1 0.00000011 a. 1. 6.1 0.00000011 a. 1. 6.1 0.000000011 a. 1. 6.1 0.000000011 a. 1. 6.1 0.0000000011 a. 1. 6.1 0.0000000000000000000000000000000000</td> <td>92.         test_2: [i s1, 100]           101:         11: s1, 0           101:         11: s1, 0           102:         11: gp. 2           103:         11: s1, 0           105:         test_2: 1: s1, 100           106:         test_2: 1: s1, 100           107:         test_2: 1: s1, 100           108:         test_2: 1: s1, 100           109:         test_3: 1: s1, 0           100:         test_3: 1: s1, 0           101:         test_4: s1, 1           101:         test_4: s1, 1           101:         test_4: s1, 1           102:         test_4: s1, 1           103:         test_4: s1, 1           104:         test_4: s1, 1           105:         test_4: s1, 1           106:         test_4: s1, 1           107:         test_4: s1, 1           108:         test_4: s1, 1           111:         test_4: 1: 1           111:         test_4: 1: 1           111:         test_4: 1: 1           112:         test_4: 1: 1           113:         test_4: 1: 1</td> <td>1 1 1 1 1 1 1 1 1 1 1 1 1 1</td> <td></td> <td></td> <td>Value (+14)</td> <td>Value (+18)</td> <td>a* (3*</td> <td>명           명</td> <td>9 50 11 12 13 14 15 15 15 15 15 15 15 19 19 19 20 20 20 20</td> <td>Devices of the second sec</td>	0.4000011/1.01 a. 1. 6.4789000 0.4000011/1.01 a. 1. 6.1 0.0000011/1.01 a. 1. 6.1 0.0000011/1.01 a. 1. 6.1 0.0000010101 a. 1. 6.1 0.000000101 a. 1. 6.1 0.000001011 a. 1. 6.1 0.00000011 a. 1. 6.1 0.00000011 a. 1. 6.1 0.000000011 a. 1. 6.1 0.000000011 a. 1. 6.1 0.0000000011 a. 1. 6.1 0.0000000000000000000000000000000000	92.         test_2: [i s1, 100]           101:         11: s1, 0           101:         11: s1, 0           102:         11: gp. 2           103:         11: s1, 0           105:         test_2: 1: s1, 100           106:         test_2: 1: s1, 100           107:         test_2: 1: s1, 100           108:         test_2: 1: s1, 100           109:         test_3: 1: s1, 0           100:         test_3: 1: s1, 0           101:         test_4: s1, 1           101:         test_4: s1, 1           101:         test_4: s1, 1           102:         test_4: s1, 1           103:         test_4: s1, 1           104:         test_4: s1, 1           105:         test_4: s1, 1           106:         test_4: s1, 1           107:         test_4: s1, 1           108:         test_4: s1, 1           111:         test_4: 1: 1           111:         test_4: 1: 1           111:         test_4: 1: 1           112:         test_4: 1: 1           113:         test_4: 1: 1	1 1 1 1 1 1 1 1 1 1 1 1 1 1			Value (+14)	Value (+18)	a* (3*	명           명	9 50 11 12 13 14 15 15 15 15 15 15 15 19 19 19 20 20 20 20	Devices of the second sec
control (C)     contro(C)     control (C)     control (C)     control (C)     control (C)	Decomposition 4 at a disc stat a dis	100: mliw 314, 53, 6 101: li s7, 0+ffffff 102: li s9, 2 103: bms 314, 57, fwi 105: erst 2; li s1, 106: mliw 314, 87, 75, 107: li s7, 0+0000000 109: li s9, 3 101: test_4 s7, 76, 111: test_4	1 1 1 1 1 1 1 1 1 1 1 1 1 1			Value (+14)	Value (+18)	a* (3*	19 00 11 12 12 14 14 14 14 14 14 14 14 14 14 14 14 14	9 50 11 12 13 14 15 15 15 15 15 15 15 19 19 19 20 20 20 20	
science of the second sec	0000211/1014/014-014-01 00000315/1014-014-01 00000315/1014-014-01 00000315/1014-014-02 00000315/1014-014-02 000000315/1014-014-02 000000315/1014-014-01 000000315/1014-014-00 000000315/1014-014-00 000000315/1014-014-00 00000001 0000000000000000000	101: 11 x7, 0:ffffff 102: 11 x7, 0:ffffff 103: hes x4, x7, fwi 105: test_3: 11 x1, 106: erlis x14, x5, fwi 107: 11 x7, 0:cont 107: 11 x7, 0:cont 111: test_4: 11 x2, Value (=4) 0:cont 0:con	(#8000000     (****************************			Value (+14)	Value (+18)	a* (3*	이 11 12 22 20 20 20 20 20 20 20 20 20 20 20 20	9 50 11 12 13 14 15 15 15 15 15 15 15 19 19 19 20 20 20 20	
	0-0003093 webs -1, s1, 0 0-0437103 we -1, s1, 0 0-1437103 he -14, s1, 0-0000378 0-0437103 he -14, s1, 0-0000378 0-0000007 he -16, webser 0-0000007 he -1, 0-000000 0-00000000 he -1, s1, 0 0-00000000 he -1, s1, 0 0-00000000 he -1, s1, 0 0-00000000 he -1, 0 0-00000000 he -1, 0 0-000000000 he -1, 0 0 0-0000000000 he -1, 0 0 0-000000000 he -1, 0 0 0-0000000000 he -1, 0 0 0-00000000000000000000000000000000	102: 1i mp. 2 103: ham std. st. st. 105: test_3: 1i st. 106: wrliw std. st. 3 107: 1i st. 0x000000 108: 1i mp. 3 109: ham std. st. fai 111: test_4: 1i st. 111: test_4: 1i st. 111: test_4: 1i st.	1 0xfffffff8000000 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1			Value (+14)	Value (+18)	a* (3*	द म म म म म म म म म म म म म म म म म म म	9 50 11 12 13 14 15 15 15 15 15 15 15 19 19 19 20 20 20 20	Dad000000000     Dad00000000     Dad000000000     Dad000000000     Dad000000000     Dad000000000     Dad000000000     Dad000000000     Dad0000000000     Dad0000000000     Dad000000000     Dad000000000     Dad000000000     Dad000000000     Dad000000000     Dad00000000     Dad00000000     Dad00000000     Dad00000000     Dad00000000     Dad00000000     Dad00000000     Dad000000000     Dad000000000     Dad000000000     Dad000000000     Dad0000000000000     Dad00000000000000     Dad000000000000000000000000000000000
evidencial de evidencial de evidenci	0-0000193 kills al. el. 2 0-0000193 kills al. el. 2 0-0000097 kills al. el. 3 0-0000097 kills al. el. 3 0-0000097 kills al. el. 3 0-0000197 kills al. el. 3 0-0000197 kills al. el. 3 0-0000197 kills al. el. 3 0-0000197 kill al. el. 3 0-0000197 kill al. el. 3 0-00000197 kill al. el. 3 0-00000197 kill al. el. 3 0-00000197 kill al. el. 3 0-00000000 kills al. el. 3 0-000000000 kills al. el. 3 0-00000000000 kills al. el. 3 0-000000000 kills al. el. 3 0-0000000000 kills al. el. 3 0-0000000000 kills al. el. 3 0-0000000000 kills al. el. 3 0-00000000000000000000000000000000000	103: bes std. s7, de 105: test_3: li s1. 105: utst_3: li s2. 106: wliw std. s1. 107: li s7, 0x000000 100: li gs. 3 100: bes std. s7, dei 111: test_6: li s2, Value (+4) 0x0000000	Cuttererrerrerrerrerrerrerrerrerrerrerrerr			Value (+14)	Value (+18)	a* (3*	20 21 20 22 22 23 24 25 25 25 25 25 25 25 25 25 25 25 25 25	9 50 11 12 13 14 15 15 15 15 15 15 15 19 19 19 20 20 20 20	Cucrossection     Cucross
0x/04001c         0.           0x/0400020         0.           0x/0400024         0.           0x/0400024         0.           0x/0400024         0.           0x/0400028         0.           0x/0400023         0.           0x/0400023 <td>0:4377.03 New 34.4.7.0.40000038 0:0000007 Div 4.0.4000000 0:0000007 Div 47.0.400000 0:0000007 Div 47.0.400000 0:0000000000 0:000000000000 0:000000</td> <td>103: bes std. s7, de 105: test_3: li s1. 105: utst_3: li s2. 106: wliw std. s1. 107: li s7, 0x000000 100: li gs. 3 100: bes std. s7, dei 111: test_6: li s2, Value (+4) 0x0000000</td> <td>Cuttererrerrerrerrerrerrerrerrerrerrerrerr</td> <td></td> <td></td> <td>Value (+14)</td> <td>Value (+18)</td> <td>a* (3*</td> <td>11 12 12 12 13 14 14 14 15 15 15 15 15 15 15 15 15 15 15 15 15</td> <td>9 50 11 12 13 14 15 15 15 15 15 15 15 19 19 19 20 20 20 20</td> <td>0.000000000 0.000000000 0.000000000 0.000000</td>	0:4377.03 New 34.4.7.0.40000038 0:0000007 Div 4.0.4000000 0:0000007 Div 47.0.400000 0:0000007 Div 47.0.400000 0:0000000000 0:000000000000 0:000000	103: bes std. s7, de 105: test_3: li s1. 105: utst_3: li s2. 106: wliw std. s1. 107: li s7, 0x000000 100: li gs. 3 100: bes std. s7, dei 111: test_6: li s2, Value (+4) 0x0000000	Cuttererrerrerrerrerrerrerrerrerrerrerrerr			Value (+14)	Value (+18)	a* (3*	11 12 12 12 13 14 14 14 15 15 15 15 15 15 15 15 15 15 15 15 15	9 50 11 12 13 14 15 15 15 15 15 15 15 19 19 19 20 20 20 20	0.000000000 0.000000000 0.000000000 0.000000
cw00400220     dw00400220     dw0040022     dw0040022     dw00400022     dw00400022     dw00400023     dw0040002     dw004002     dw004002     dw004002     dw004002     dw004002     dw00400     dw004002     dw00400     dw004002     dw004002     dw00400     dw0040     dw004     dw0040     dw0040     dw0040	0-0000007 [bit st. 0-0000000 0-0000007 [bit st. 0.1.0 0-001007 [bit st. 0.1.0 0-001007 [bit st. 0.1.1 0-0000007 [bit st. 0.0 0-0000007 [bit st. 0.0 0-0000000000 [bit st. 0.0 0-0000000000 [bit st. 0.0 0-000000000 [bit st. 0.0 0-000000000 [bit st. 0.0 0-000000000 [bit st. 0.0 0] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	105: test_3: list. 105: erli*s14. s1. 3 107: list. 0x000000 109: list. 3 109: list. 3 109: list. 4, 7, fail. 111: test_4: list. Value (+4) 0x00000000	Cuttererrerrerrerrerrerrerrerrerrerrerrerr			Value (+14)	Value (+18)	a* (3*	20 22 23 24 24 25 25 25 25 25 25 25 25 25 25 25 25 25	00 11 12 13 14 15 16 16 17 17 18 19 20 20 21	0x000000000 0x000000000 0x000000000 0x000000
0x0040024         0x0040024         0x0040024         0x00400224         0x00400224         0x00400224         0x00400234         0x00400234         0x00400234         0x004000234         0x04000234         0x0400	0-00000000 k-0.1.1.0.0 0-00000001 k-0.1.0.1 0-0000001 k-0.1.0.1 0-00000000 k-0.1.0.0 0-00000000 k-0.1.0.0 0-0000000 k-0.1.0.0 0-0000000 k-0.1.0.0 0-0000000 k-0.1.0.0 0-0000000 k-0.0.0 0-0000000 k-0.0.0 0-000000 k-0.0.0 0-0000000 k-0.0.0 0-000000 k-0.0.00 0-000000 k-0.0.0 0-0000000 k-0.0.0 0-0000000 k-0.0.0 0-0000000 k-0.0.0 0-0000000 k-0.0.0 0-0000000 k-0.0.0 0-00000000 k-0.0.0 0-00000000 k-0.0.0 0-0000000 k-0.0.0 0-0000000 k-0.0.0 0-0000000 k-0.0.0000000 k-0.0.00000000 k-0.0.0000000000	106: srliv sl4, sl. 1 107: li sr. 0x000000 100: li gy, 3 109: bas sl4, sr. fai 111: rest_6: li sl. Value (+4) 0x00000000	Value (+8) outcoccocc			Value (+14)	Value (+18)	a* (3*	11 12 13 14 16 16 16 17 17 17 17 17 17 17 17 17 17 17 17 17	11 12 13 14 15 15 17 17 19 19 20 20 21	0.000000000 0.00000000 0.000000000 0.000000
Cv00400223     Gv00400223     Gv00400223     Gv0040023     Gv00400234     Gv00400234     Gv00400234     Gv00400234     Gv00400234     Gv00400234     Gv0040023     Gv0040023     Gv0010000     Gv0010002     Gv10010000     Gv10010002     Gv10010000     Gv1001000     Gv1001000     Gv1001000     Gv1001000     Gv100100     Gv10010     Gv100100     Gv100100     Gv10010     Gv100100     Gv100100     Gv10010     Gv10010     Gv10010     Gv10010     Gv10     Gv10010     Gv10010     Gv10010     Gv10     G	00010271% w14.8.1.1 000000371% w15.9.0.000000 000003393 w464.4.7.0 0000030101 w45.8.7.0 0000030101 w45.8.7.0 000000011 w15.0000000011 000000001 w15.0000000011 00000000000000000000000000	107: 11 s7, 0x000000 100: 11 gs, 3 109: has x14, x7, fai 111: test_4: 11 x1, Value(+4) 0x00000000	Value (+0) 0x0000000			Value (+14)	Value (+18)	a* (3*	द्ध च च च च च च च च च च च च च च च च च च च	12 13 14 15 16 17 17 19 19 20 21	0.000000000 0.00000000 0.00000000 0.000000
0x0040002c 0x 0x0040003 0x 0x0040003 0x 0x0040003 0x 0x0040003 0x 0x0040003 0x 0x0040003 0x 0x0040003 0x 0x0040003 0x 0x10010000 0x10010000 0x10010000	0+e000017 (but al. 0+000000 0+0000139 (but al. 0+1 al. 0+00000139 (but al. 0+1 al. 0+00000119 (but al. 0+1 al. 0+0000011 (but al. 0+11120000 0+00000001 (but al. 0+11120000 0+000000000000000000000000000	107: 11 s7, 0x000000 100: 11 gs, 3 109: has x14, x7, fai 111: test_4: 11 x1, Value(+4) 0x00000000	Value (+0) 0x0000000			Value (+14)	Value (+18)	a* (3*	ವ ಕತ ಕತ ಕ ಕ ಕ ಕ ಕ ಕ ಕ	13 14 15 16 17 18 19 20 21	0.000000000 0.00000000 0.00000000 0.000000
0x00400030 0x 0x00400034 0x 0x000400032 0x 0x00400032 0x 0x00400032 0x 0x00000000 0x00010000 0x10010000 0x10010000 0x10010000	04000330144 443.4 x7.0 0 04003001301464 x7.4 x7.0 0 0400300130148 444 x7.0 4000000334 0400000037 1 441 x1.0 47f950000 0400000000 1 041 x1.0 47f950000 00 0400000000	100: li gp. 3 100: has x14, x7, fai 111: text_4: li x1, Value (+4) Gu00000000	1 cufffffffb0000000 Value (+0) Gu0000000			Value (+14)	Value (+18)	a* (3*	64 65 62 62 52 53 54	14 15 16 17 18 19 20 21	0x000000000 0x000000000 0x000000000 0x000000
0x00400034 03 0x00400033 03 0x00400034 03 5egment Address 0x10010000 0x10010000 0x10010000	0x000001821x46x x3, x0, 3 0x2271421 bas x14, x7, 0x00000033e 0x0000001 1xx x14, x7, 0x00000033e 0x0000001 1xx x1, 0x17850000 0x00000000 1xx x14, x14, 0x00000000 0x000000000 0x0000000000000	109: has 214, 27, fai 111: tert_4: li 21, Value (+4) Gu0000000	Value (+8) 0x0000000			Value (+14)	Value (+18)	a* (3*	15 18 17 17 19 19 19 19	15 16 17 18 19 20 21	0x000000000 0x000000000 0x000000000 0x000000
0x00400032 0x 0x00400034 0x 5egment Address 0x10010000 0x10010000 0x10010000	0:12771 M3 New 154, 07, 0:00003374 0:00000017 Usi x1, 0:07280000 Value (+0) 0:00000000	109: has 214, 27, fai 111: tert_4: li 21, Value (+4) Gu0000000	Value (+8) 0x0000000			Value (+14)	Value (+18)	a* (3*	40 47 52 53 54	16 17 18 19 20 21	0x0000000000 0x0000000000 0x0000000000
Segment         0///           Address         0///000000000000000000000000000000000	0x80000017[bui x1, 0x87880000 Value (+0) 00 0x0000000	Value (+d)	Value (+8) 0x0000000			Value (+14)	Value (+18)	a* (3*	12 12 13 14	17 18 19 20 21	0x000000000 0x000000000 0x000000000 0x000000
Segment Address 0x10010000 0x10010000 0x10010004 0x10010040	Value (+0) 00 0x0000000	Value (+4) 6x00000000	Value (+0) 0x0000000			Value (+14)	Value (+18)	a* (3*	s) s4	18 19 20 21	0+000000000 0+000000000 0+000000000 0+000000
Address 0x10010000 0x10010020 0x10010040 0x10010060	000000000000000000000000000000000000000	0w00000000	0x00000000			Value (+14)	Value (+18)	a* (3*	s) s4	19 20 21	0+0000000000000000000000000000000000000
Address 0x10010000 0x10010020 0x10010040 0x10010060	000000000000000000000000000000000000000	0w00000000	0x00000000			Value (+14)	Value (+18)		54	20	0+0000000000
Address 0x10010000 0x10010020 0x10010040 0x10010060	000000000000000000000000000000000000000	0w00000000	0x00000000			Value (+14)	Value (+18)			21	0.0000000000
0x10010000 0x10010020 0x10010040 0x10010040	000000000000000000000000000000000000000	0w00000000	0x00000000			Value (+14)	Value (+18)	Value (+1c)			
0x10010000 0x10010020 0x10010040 0x10010040	000000000000000000000000000000000000000	0w00000000	0x00000000						49	22	0,00000000
0x10010020 0x10010340 0x10010060							0_00000000	Du00000000 +	1	21	0,000000000
0x10010040 0x10010060			0x00000000	Cw00000000	0_000000000	0x00000000	0/0000000	Cuccocococ	d	24	0,000000000
0x10010000	47 Ou00000000	0x00000000	0,00000000	0x00000000	0x00000000	0x00000000	0/0000000	Cu00000000	4	25	0,000000000
		0x00000000	0.000000000	0x00000000	0x00000000	0x00000000	0,00000000	Cuccocococo	410	25	0,0000000000
		0x00000000	0.000000000	0x00000000	0x00000000	0x00000000	0,00000000	0x00000000	111	27	0,0000000000
0x100100w0		0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0200000000	0x00000000	1	25	0,0000000000
0x100100.c		0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0200000000	0x00000000	14	22	0,0000000000
0x100100+0		0x00000000	0x00000000	0=00000000	0x00000000	0x00000000	0±00000000	0x00000000	1	30	0,0000000000
0x10010100		0=000000000	0x00000000	0+000000000	0x00000000	0x00000000	0.00000000	0x00000000		31	0,0000000000
0x10010130		0=000000000	0x00000000	0=00000000	0+00000000	0x00000000	0±00000000	0+00000000	10		0.0000000000
0x10010140		0+000000000	0=00000000	0+000000000	0=00000000	0x00000000	0-00000000	0x00000000	<i>.</i>		
0x10010100		0+00000000	0=00000000	0+00000000	0=00000000	0x00000000	0+00000000	0+00000000			
0x10010130		0+00000000	0=00000000	0~00000000	0=00000000	0x00000000	0=00000000	0+00000000			
0x100101x0		0+00000000	0=00000000	0+00000000	0=00000000	0x00000000	0+00000000	0+00000000 -			
								•	R	egisters panel	
			x10010000 (.data) 👻	Rexadecimal Address	ses 🕑 Hexadecimal Valu	8 🔲 ASCII					
es Rus IO											
Arranhla' arr	ssembling F:\Research\misc\TAVCER	are test/class							1		
	E:\Besearch\misc\TA\CEM03420\tesls					ram informati	ion panel				
	F:\Renearch\misc\TA\CEBG3420\tesl:	\tert am line 318 column	2: BARS does not recom	aize the .global directiv	a. Ignored.	in anoman	tore burney				
Assemble: ope	peration completed successfally.										

RARS execution panel

- Create a new source file: Ctrl + N
- Close the current source file: Ctrl + W
- Assemble the source code: F3
- Execute the current source code: F5
- Step running: F7
- Instructions & System call query: F1

# **System Service in RARS**

RARS provides a small set of operating system-like services through the system call (ecall) instruction. Register contents are not affected by a system call, except for result registers in some instructions.

- Load the service number (or number) in register a7.
- Load argument values, if any, in a0, a1, a2 ..., as specified.
- Issue ecall instruction.
- Retrieve return values, if any, from result registers as specified.

Name	Number	Description	Inputs	Outputs
PrintInt	1	Prints an integer	a0 = integer to print	N/A
PrintFloat	2	Prints a float point number	fa0 = float to print	N/A
PrintString	4	Prints a null-terminated string to the console	a0 = the address of the string	N/A
ReadInt	5	Reads an int from input console	a0 = the int	N/A
ReadFloat	6	Reads a float from input console	fa0 = the float	N/A
ReadString	8	Reads a string from the console	a0 = address of input buffer, a1 = maximum number of characters to read	N/A
Open	1024	Opens a file from a path Only supported flags (a1), read-only (0), write-only (1) and write- append (9)	a0 = Null terminated string for the path, a1 = flags	a0 = the file decriptor or -1 an error occurred
Read	63	Read from a file descriptor into a buffer	a0 = the file descriptor, a1 = address of the buffer, a2 = maximum length to read	a0 = the length read or -1 i error
Write	64	Write to a filedescriptor from a buffer	a0 = the file descriptor, a1 = the buffer address, a2 = the length to write	a0 = the number of charcter written
LSeek	62	Seek to a position in a file	a0 = the file descriptor, a1 = the offset for the base, a2 is the begining of the file (0), the current position (1), or the end of the file (2)}	a0 = the selected position fro the beginning of the file or is an error occurred

# An Example of System Calls in RARS I

An example shows how to use system calls in RARS

```
Using system call
```

```
# Comment giving name of program and description
# sys-call.asm
# Bare-bones outline of RISC-V assembly language program
.globl _start
.data
msg: .asciz "Hello,_world!\n"
.text
_start:
li a7, 4  # system call code for PrintString
la a0, msg  # address of string to print
ecall  # Use the system call
# End of program, leave a blank line afterwards is preferred
```

You can check the output in Run/IO of the program information panel.

- *li* loads a register with an immediate value given in the instruction.
- *la* loads an address of the specified symbol.
- *.asciz* emits the specified string within double quotes and includes the terminated zero character at the end.

# Lab 1-1 Assignment

Write a RISC-V assembly program step by step as shown below:

- Define two variables var1 and var2 which have initial value 15 and 19, respectively. (var1 = 15 and var2 = 19)
- **2** Print MEMORY addresses of var1 and var2 using syscall.
- **③** Increase var1 by 1 and multiply var2 by 4.
- 4 Print var1 and var2 again.
- **(5)** Swap var1 and var2 and print them. (var1 and var2 are changed)

### Submission Method:

Submit the source code and report after the whole lectures of Lab1 into Blackboard.

- Variables should be declared following the .data identifier.
- 2 <name>: .<datatype> <value>
- **(3)** Use la instruction to access the RAM address of declared data.
- ④ Use system call to print integers.
- **5** Do not forget exit system call.
- 6 You should print a new line to distinguish outputs!