

CS7381 Project 2

MIPS Assembly Code Programming Using MARS Tool

Name: Bingying Liang

ID: 48999397

Distance

February 19 2023

In the previous project (Project 1), you were introduced to the MARS tool. For this project, you will use the MARS tool to develop and run a MIPS assembly language program. You will create and run MIPS assembly code for the following high-level language (HLL) code. I have provided both C++ and Python versions of the HLL code.

C++ Version

```
int j = 10;
int x = 1;
do
{
    x = x + 3*j;
    cout << x << " ";
    j--;
} while (j > 0);
cout << endl;
```

Java Version

```
j = 10
x = 1
while True:
    x = x + 3 * j
    print(x, end = " ")
    j -= 1
    if (j == 0):
        break
```

Screen shot of the MARS console

Screen shot of the MARS console showing the execution of a MIPS assembly program.

The window title is: `/Users/eve/Desktop/CS7381_Computer_Architecture/Program/Project_2/Code/bylian2.asm - MARS 4.5`

The menu bar includes: File, Edit, Run, Settings, Tools, Help.

The toolbar includes icons for file operations, editing, and execution. The status bar indicates: Run speed at max (no interaction).

The main window is divided into several sections:

- Text Segment:** Displays the assembly code and its source.

Bkpt	Address	Code	Basic	Source
	0x00400000	0x2409000a	addiu \$9,\$0,10	5: li \$t1, 10 # j = 10
	0x00400004	0x240a0001	addiu \$10,\$0,1	6: li \$t2, 1 # x = 1
	0x00400008	0x01295820	add \$11,\$9,\$9	15: add \$t3, \$t1, \$t1 # t3 = t1+t1 = 2 * j
	0x0040000c	0x01695820	add \$11,\$11,\$9	16: add \$t3, \$t3, \$t1 # t3 = t3 + t1 = 3 * j
	0x00400010	0x014b5020	add \$10,\$10,\$11	17: add \$t2, \$t2, \$t3 # t2 = t2 + t3 -> x = x + 3 * j
	0x00400014	0x000a2021	addu \$4,\$0,\$10	20: move \$a0, \$t2 # move t2(x) value to \$a0
	0x00400018	0x24020001	addiu \$2,\$0,1	21: li \$v0, 1 # v0 = 1, syscall -> print int
	0x0040001c	0x0000000c	syscall	22: syscall # print x
	0x00400020	0x3c011001	lui \$1,4097	25: la \$a0, space # load address la space
	0x00400024	0x34240000	ori \$4,\$1,0	
	0x00400028	0x24020004	addiu \$2,\$0,4	26: li \$v0, 4 # \$v0 = 4, syscall -> print_string
	0x0040002c	0x0000000c	syscall	27: syscall
	0x00400030	0x2129ffff	addi \$9,\$9,-1	29: addi \$t1, \$t1, -1 # j--
	0x00400034	0x11200001	beq \$9,\$0,1	31: beq \$t1, \$zero, end # if (j==0): break
	0x00400038	0x0c100002	jal 0x00400008	32: jal loop # else (j != 0): loop
	0x0040003c	0x2402000a	addiu \$2,\$0,10	34: end: li \$v0, 10
	0x00400040	0x0000000c	syscall	35: syscall
- Data Segment:** Displays memory addresses and their values.

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	32	0	0	0	0	0	0	0
0x10010020	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0
0x10010140	0	0	0	0	0	0	0	0
0x10010160	0	0	0	0	0	0	0	0
0x10010180	0	0	0	0	0	0	0	0
- Registers:** Displays the state of MIPS registers.

Name	Number	Value
\$zero	0	0
\$at	1	268500992
\$v0	2	10
\$v1	3	0
\$a0	4	268500992
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	0
\$t1	9	0
\$t2	10	166
\$t3	11	3
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	0
\$s1	17	0
\$s2	18	0
\$s3	19	0
\$s4	20	0
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$t8	24	0
\$t9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	4194364
pc		4194372
hi		0
lo		0
- Mars Messages:** Displays messages from the program.


```
31 58 82 103 121 136 148 157 163 166
-- program is finished running --
```

The bottom section includes a "Clear" button and a "Run I/O" button.

