

COMP3350 Spring, 2024

Lab 1

Important Notes:

- Solutions turned in must be your own. Please, mention references (if any) at the end of each question.
- The code (if any) must be accompanied by adequate comments. Writing only the code could lead to **zero** credits. Likewise, the screenshots should be accompanied with adequate descriptions.
- Partial score may be provided based on the intermediate steps and results (even if they are not completely correct).
- All units must be mentioned wherever required.
- All submitted lab reports must be in the PDF format unless otherwise mentioned. Texts of the reports are expected to be typed in for which you could use software programs like L^AT_EX, Microsoft Word etc.

Please Follow the Lab Instructions and Complete the following Tasks.

1. Task 1: In Step 1 Download and Launch MARS, attach a screenshot after you have launched MARS. (35 points)
2. Task 2: In Step 2 Run Example Code. Change the values of the multiplicand and multiplier. Recompile and run. Observe the changes in the machine codes and register values. Please attach the following 2 screenshots: 1) a screenshot of the code with the changed values (30 points), 2) a screenshot that clearly shows the compiled code and register values after assembling and running (35 points).
3. Task 3 (Bonus Task): In Step 3, change the values of the operands based on the instructions. Compile and run the code in each case, then
 - 1) answer whether there are exceptions (Yes or no) and explain the reasons (15 points)

Please answer whether there are exceptions and explain the reasons using the following format:

a. (yes or no). Reason: ...

b. (yes or no). Reason: ...

c. (yes or no). Reason: ...

d. (yes or no). Reason: ...

e. (yes or no). Reason: ...

2) record the console output result as well as the hexadecimal values in registers \$t0, \$t1, and \$t2 **when there is no exception** (15 points).

Attach a screenshot clearly showing values of registers \$12, \$13, \$14 **when there is an exception** (10 points).

Lab Instructions

1. Download and Launch MARS

1. Download MARS from

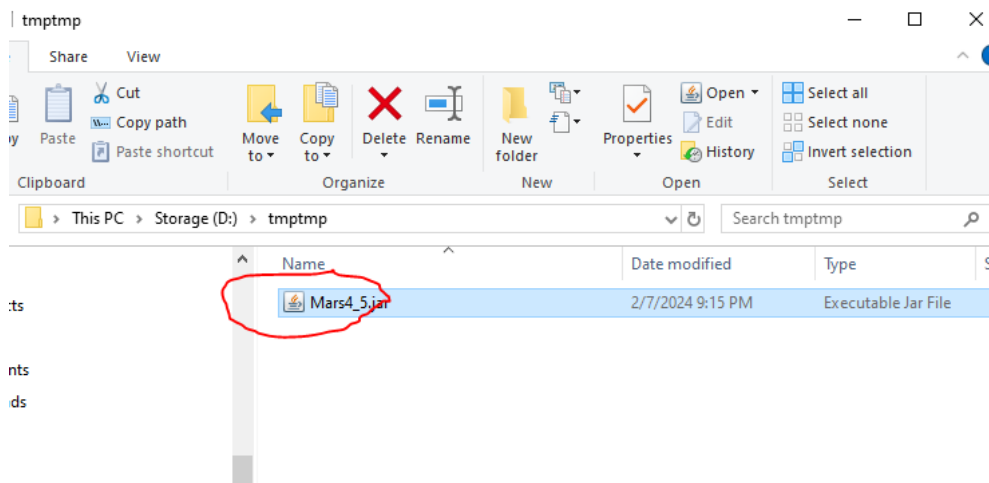
<https://courses.missouristate.edu/kenvollmar/mars/download.htm>

If you don't have Java, install JDK 17

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>



You can launch the emulator by double clicking it (in Windows and Mac):



Alternatively, you can run the emulator from command line (terminal) by using the `cd` command to change the directory to where MARS is located, and then typing

```
java -jar <MARS .jar name>
```

For example, if the file name is `Mars4_5.jar`

We type

```
java -jar Mars4_5.jar
```

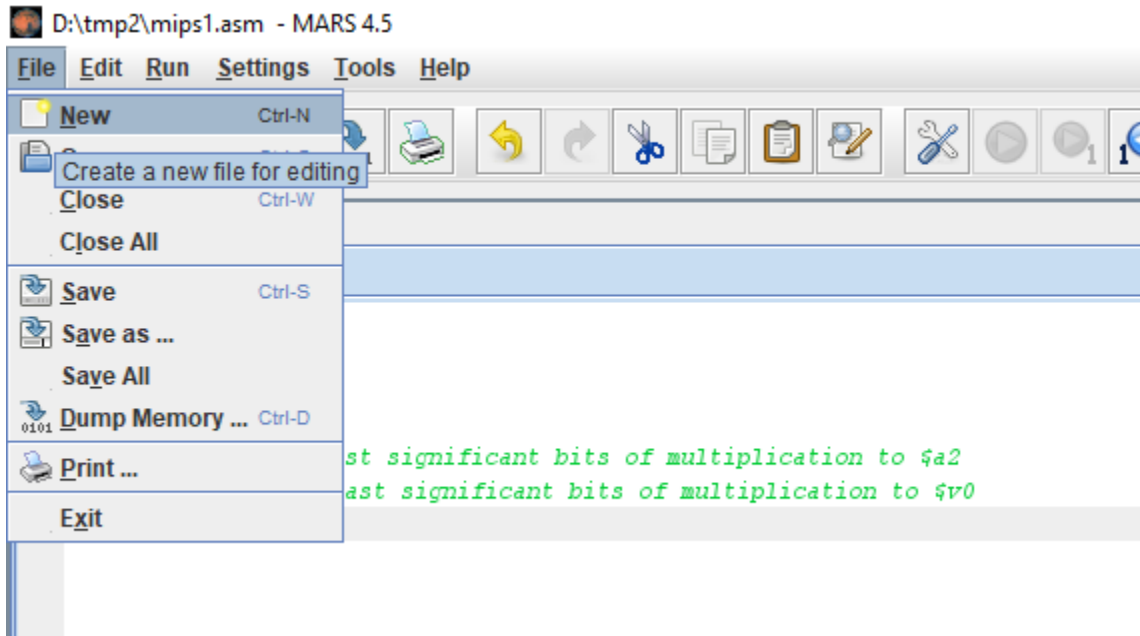
Here is an example of the steps in Windows:

```
C:\>d:  
  
D:\>cd tmptmp  
  
D:\tmptmp>java -jar Mars4_5.jar
```

In this example, the MARS executable file is located at D:\tmptmp directory.

2. Run Example Code to Multiply Two Numbers and Make Observations

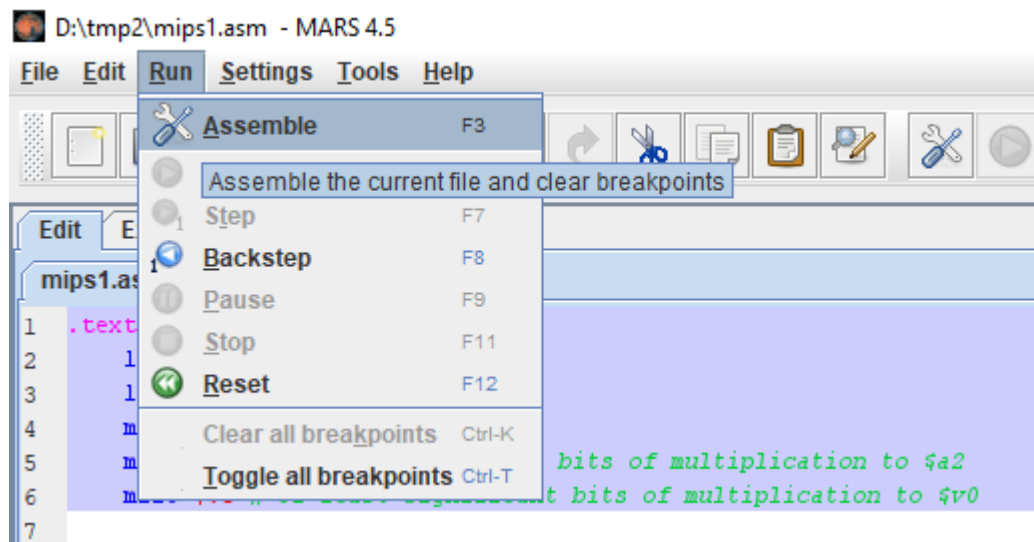
First, we will create a new file:



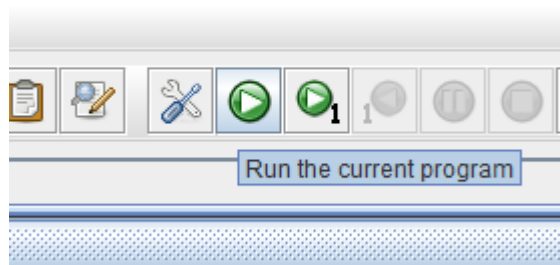
We then copy the following code into the editor:

```
.text  
  
li $a0, 5  
  
li $a1, 3  
  
mult $a0, $a1  
  
mfhi $a2 # 32 most significant bits of multiplication to $a2  
  
mflo $v0 # 32 least significant bits of multiplication to $v0
```

After saving the file, we click assemble (or type F3)



We then click run the current program (or type F5)



The screenshot shows the Mars MIPS simulator interface. The 'Text Segment' window displays assembly code with addresses and instructions. The 'Registers' window shows the state of MIPS registers. Red circles highlight specific values in the registers: \$v0 (0x0000000f), \$a0 (0x00000005), \$a2 (0x00000003), and \$lo (0x0000000f).

Bkpt	Address	Code	Basic	Source
	0x00400000	0x24040005	addiu \$4,\$0,0x00000005	2: li \$a0, 5
	0x00400004	0x24050003	addiu \$5,\$0,0x00000003	3: li \$a1, 3
	0x00400008	0x00850018	mult \$4,\$5	4: mult \$a0, \$a1
	0x0040000c	0x00003010	mfhi \$6	5: mfhi \$a2 # 32 most significant bits of multiplication to \$a2
	0x00400010	0x00001012	mflo \$2	6: mflo \$v0 # 32 least significant bits of multiplication to \$v0

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x0000000f
\$v1	3	0x00000000
\$a0	4	0x00000005
\$a1	5	0x00000003
\$a2	6	0x00000003
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7fffffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400014
hi		0x00000000
lo		0x0000000f

Mars Messages: Run I/O

-- program is finished running (dropped off bottom) --

We can see that the computations have been completed by observing corresponding register values.

In Task 2, try to change the values of the multiplicand and multiplier (highlighted in blue):

.text

li \$a0, 5

li \$a1, 3

mult \$a0, \$a1

mfhi \$a2 # 32 most significant bits of multiplication to \$a2

mflo \$v0 # 32 least significant bits of multiplication to \$v0

Record your observations and explain the reasons of the changes.

3. Run Example Code to Add Two Numbers and Make Observations on the Results

Copy the following code into the editor:

```
.text

# load the 1st operand

li $t1, -21474837 # Please change the value here

# load the 2nd operand

li $t2, 2 # Please change the value here

add $t0, $t1, $t2

# print the result

li $v0, 1 #Place 1 in $v0 in order to print an integer

addi $a0, $t0, 0 #Load the integer from memory into $a0

syscall #syscall to print the integer # print the result
```


Assemble and Run, we observe that the program will compute the correct output in the console in this case:

The screenshot shows the MARS 4.5 IDE with the following components:

- Text Segment:**

Bkpt	Address	Code	Basic
	0x00400000	0x3c01feb8	lui \$1, 0xfffffeb8
	0x00400004	0x342951eb	ori \$9, \$1, 0x000051eb
	0x00400008	0x240a0002	addiu \$10, \$0, 0x0000...
	0x0040000c	0x012a4020	add \$8, \$9, \$10
	0x00400010	0x24020001	addiu \$2, \$0, 0x00000001
	0x00400014	0x21040000	addi \$4, \$8, 0x00000000
	0x00400018	0x0000000c	syscall
- Data Segment:**

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000
- Registers:**

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0xfeb80000
\$v0	2	0x00000001
\$v1	3	0x00000000
\$a0	4	0xfeb851ed
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0xfeb851ed
\$t1	9	0xfeb851eb
\$t2	10	0x00000002
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7ffffc00
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x0040001c
hi		0x00000000
lo		0x00000000
- Mars Messages:**

```
-- -21474835
-- program is finished running (dropped off bottom) --
```

We can also observe that there is no exception when we select and observe the Coproc 0 tab:

Registers	Coproc 1	Coproc 0
Name	Num	selected Coprocessor 0 (exceptions and interrupts) registers
\$8 (vaddr)	8	0x00000000
\$12 (status)	12	0x0000ff11
\$13 (cause)	13	0x00000000
\$14 (epc)	14	0x00000000

In Task 3, please replace the operands in the code with the operands in following 5 cases and record your answers/observations in each case:

- a. $1073741824 + 1073741824 = ?$
- b. $-2147483647 + 30 = ?$
- c. $-2147483647 + (-1) = ?$
- d. $-2147483648 + (-1) = ?$
- e. $38 + (-40) = ?$