

ELEC 385: Computer System Design
MARS Tutorial


MARS is a software simulator for MIPS assembly language, intended for educational use. This tutorial was written by Pete Sanderson at Otterbein College and Ken Vollmar at Missouri State University.

MARS may be downloaded from www.cs.missouristate.edu/MARS.

Basic MARS Use

The example program is **Fibonacci.asm** to compute everyone's favorite number sequence.

1. Start MARS from the Start menu or desktop icon.

2. Use the menubar File...Open or the Open icon  to open Fibonacci.asm in the default folder. *(All icons have menubar equivalents; the remainder of these steps will use the icon whenever possible.)*

3. The provided assembly program is complete. Assemble the program using the icon



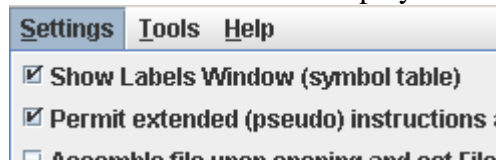
4. Identify the location and values of the program's initialized data. Use the checkbox to toggle the display format between decimal and hexadecimal ☐ **Hexadecimal Values**.
 - The nineteen-element array **fib** is initialized to zero, at addresses 0x10010000 ... 0x10010048.
 - The data location **size** has value 19_{ten} at 0x1001004c.
 - The addresses 0x10010050 ... 0x1001006c contain null-terminated ASCII strings.

Use the checkbox to toggle the display format between decimal and hexadecimal,

☐ **Hexadecimal Values**.

5. Use the Settings menu to configure the MARS displays. The settings will be retained for the next MARS session.

- The Labels display contains the addresses of the assembly code statements with a label, but the default is to *not* show this display. Select the checkbox from the







Settings menu.

- Select your preference for allowing pseudo-instructions (programmer-friendly instruction substitutions and shorthand).
 - Select your preference for assembling *only one* file, or *many* files together (all the files in the current folder). This feature is useful for subroutines contained in separate files, etc.
 - Select the startup display format of addresses and values (decimal or hexadecimal).
6. Locate the Registers display, which shows the 32 common MIPS registers. Other tabs in the Registers display show the floating-point registers (Coproc 1) and status codes (Coproc 0).
 7. Use the slider bar to change the run speed to about 10 instructions per second.

 This allows us to “watch the action” instead of the assembly program finishing directly.

8. Choose how you will execute the program:

- The  icon runs the program to completion. Using this icon, you should observe the yellow highlight showing the program’s progress and the values of the Fibonacci sequence appearing in the Data Segment display.
- The  icon resets the program and simulator to initial values. Memory contents are those specified within the program, and register contents are generally zero.
- The  icon is “single-step.” Its complement is , “single-step backwards” (undoes each operation).



9. Observe the output of the program in the Run I/O display window:

The Fibonacci numbers are:
 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181
 -- program is finished running --

10. Modify the contents of memory. (Modifying a register value is exactly the same.)


- Set a breakpoint at the first instruction of the subroutine which prints results. Use the checkbox at the left of the instruction whose address is 0x00400060 (This will likely be 0x00400058 instead).

<input checked="" type="checkbox"/>	0x00400060	0x00044020	add \$8,\$0,\$4	50: print:a
-------------------------------------	------------	------------	-----------------	-------------

- Reset  and re-run  the program, which stops at the breakpoint.
- Double-click in one of the memory locations containing the computed Fibonacci numbers. The cell will be highlighted and will accept keyboard entry, similar to a

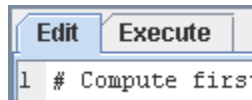
spreadsheet. Enter some noticeably different value, and use the Enter key or click outside the cell to indicate that the change is complete.

- Click  to continue from the breakpoint. The program output includes your entered value instead of the computed Fibonacci number.


11. Open the Help  for information on MIPS instructions, pseudoinstructions, directives, and syscalls.




12. Modify the program so that it prompts the user for the Fibonacci sequence length.

- Select the Edit tab in the upper right to return to the program editor.



- The MIPS comment symbol is #. All characters on the line after the character # are ignored.
- Un-comment lines 12-19. The newly exposed program fragment will prompt the user for the length of the Fibonacci sequence to generate, in the range $2 \leq x \leq 19$. (The length of the sequence must be limited to the size of the declared space for result storage.)
- Determine the correct **syscall** parameter to perform “read integer” from the user, and insert the parameter at line The correct **syscall** parameter may be found at

Help  ... Syscall tab...read integer service. The completed line will have the form `li $v0, 42` (where in this case 42 is not the right answer).

- Reset  and re-run  the program. The program will stop at the breakpoint you inserted previously. Continue and finish with .