

SCC150 – MIPS/Assembly

Week 12 Practical

Angie Chandler

SCC – Lancaster University

Week 12

Outline

- Bitmap Display
- Using lui and sw
- Displaying memory in colour
- Creating a counter
- Drawing a picture!

Why Draw Pictures in MIPS?

- Two reasons
 - It's a good way to practice memory access – you can see what you've done!
 - The original PlayStation actually used MIPS, so in honour of that...
 - PS1 is quite old, so to give you an idea, my favourite PS1 game was Wipeout 2097, see image and video for the curius
 - <https://www.youtube.com/watch?v=GmTRFf4Pquk>



Bitmap Display Introduction

- Open MARS
 - In MARS, go to Tools on the Menu and select Bitmap Display
 - Press the button that says Connect to MIPS on Bitmap Display
 - You are now connected to the display...
-
- Notice that it says “base address for display” is 0x10010000
 - This means that anything stored in that register will be represented as colour on the display

Accessing Memory for Display (lui and sw)

#address of memory for image (lui – load upper immediate)

addi \$s0,\$zero,0 #sets \$s0 to zero – initialise

lui \$s0, 0x1001 #loads upper register with 0x1001

#this sets register \$s0 to 0x10010000 (address of memory for bitmap)

#set the content of the register at address 0x10010000

addi \$a0, \$zero,0x00ff #set register \$a0 to 0xff

sw \$a0, 0(\$s0) #store in \$s0

Colour!

- Congratulations! You've now coloured a pixel in blue!
 - Of course, it's one pixel, so you can't see it...
 - But the register shows it
- How do I know it's blue?
 - RGB – look up hex colours online
 - 0x00000000
 - You need to either use `lui` or `sll` to reach the red bits

Exercise 1 – Visible Colour

- Obviously, you can't see just one pixel, so you need to keep going.
- Remember word size, each time you move to the next register, you need to add 4
- Note, what you want to do is:

`sw $a0, 4($s0)` `#store contents of $a0 in $s0+4`

- But you can't (the number (i.e. 4) can't be a variable), so you have to add 4 to the base address

Exercise 2 – Background Colour

- Fill in the entire background in a colour of your choosing
- Is it stopping in the right place, or are you going too far?
- 512 x 256 pixels, 4 bytes per word
- Counter
- Branch - to detect the end
- Got it working? Try a different colour!

Exercise 3 – Draw a Line

- This requires better control
 - Fill in the background, then draw a line in a different colour.
 - Start with a horizontal line
 - Then try a vertical line.
-
- Remember 512 x 256 pixels, 4 bytes per register

Exercise 4 (optional) – Draw a Square

- Extend what you've learnt to draw a square on the display
- How might you draw multiple squares?
- How might you set the program up to change colours easily?

Examples – Drawing

C:\Users\chandlak\OneDrive - Lancaster University\Teaching\computing\scc150 assembler\mars\wk12square.asm - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt

Bitmap Display, Version 1.0

Unit Width in Pixels: 1

Unit Height in Pixels: 1

Display Width in Pixels: 512

Display Height in Pixels: 256

Base address for display: 0x10010000 (static data)

Tool Control

Disconnect from MIPS Reset Help Close

Registers

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00ff0000
\$a1	5	0x0000ff00
\$a2	6	0x000000ff
\$a3	7	0x00ffffff
\$t0	8	0x00000004
\$t1	9	0x00000800
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000001
\$t6	14	0x00000002
\$t7	15	0x00000003
\$s0	16	0x10010000
\$s1	17	0x10020040
\$s2	18	0x10020140
\$s3	19	0x10040040
\$s4	20	0x10040140
\$s5	21	0x10040140
\$s6	22	0x10040140
\$s7	23	0x00000000
\$t8	24	0x00000004
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7fffffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x004000c8

Mars Messages Run I/O

Clear

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

09:37 08/01/2021

Examples – Drawing

C:\Users\chandlak\OneDrive - Lancaster University\Teaching\computing\scc150 assembler\mars\wk12square.asm - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10020000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10020020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10020040	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00
0x10020060	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00
0x10020080	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00
0x100200a0	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00
0x100200c0	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00
0x100200e0	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00
0x10020100	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00
0x10020120	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00	0x00ffff00
0x10020140	0x00ffff00	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10020160	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10020180	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100201a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100201c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100201e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

0x10010000 (.data) Hexadecimal Addresses Hexadecimal Values ASCII

Mars Messages Run I/O View next higher address range; hold down for rapid fire

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

Registers Coproc 1 Coproc 0

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000

Bitmap Display, Version 1.0

Unit Width in Pixels 1

Unit Height in Pixels 1

Display Width in Pixels 512

Display Height in Pixels 256

Base address for display 0x10010000 (static data)

Disconnect from MIPS

0x10010000 0x00000000

0x10010008 0x10008000

0x10010010 0x7ffffc

0x10010018 0x00000000

0x10010020 0x00000000

0x10010028 0x004000c8

Note – the green arrows allow you to move along the memory

See the non-zero memory addresses? Essentially, that's all you're trying to do