

### Vocab

Three MIPS steps: Fetch, Decode, Execute  
C Compiler is preprocessor and translator  
Linker assembles all functions and make final exe  
Loader loads program into memory

### Memory Allocation

```
int A = 200;
float B[2][2] = {{1.0,2.0}, {3.0,4.0}};
float *p2 = &B[1][1];
int *p1 = &A;
char s[ ] = "foo";
```

|      |         |      |      |      |
|------|---------|------|------|------|
| 2000 | A       |      |      |      |
| 2004 | B[0][0] |      |      |      |
| 2008 | B[0][1] |      |      |      |
| 2012 | B[1][0] |      |      |      |
| 2016 | B[1][1] |      |      |      |
| 2020 |         |      |      |      |
| 2024 | p2      |      |      |      |
| 2028 |         |      |      |      |
| 2032 | p1      |      |      |      |
| 2036 |         |      |      |      |
| 2040 | s[0]    | s[1] | s[2] | s[3] |

### Swap

- (7 pts) Write a MIPS code fragment that adds the value 0x4FA2C3 to register \$1 and puts the result in \$2. Modify only registers \$1 and \$2. (You may use hexadecimal immediate values.) For maximum credit, include comments.

| Label | Instruction          | Comment                 |
|-------|----------------------|-------------------------|
|       | lui \$2, 0x4F        | # load upper 16 bits    |
|       | ori \$2, \$2, 0xA2C3 | # load rest of constant |
|       | add \$2, \$2, \$1    | # add it to \$1         |

### Values after

$x = 7, y = 7, \text{ and } z = 2$   
if ( $z = x < y$ ) {  $x += 3$ ;  $y -= 1$ ; }  
else  $x = y++$ ;  
 $x = 7, y = 8, z = 0$

### Constant Type

29LU = unsigned long integer  
"t" = string  
-39.54Lf = long double  
7hd = short integer

### Values

$(p1 = 2000)(*p1 = 200)(\&p1 = 2032)(p2 = 2016)$   
 $'p2 = 4)(\&p2 = 2024)(\&(s[1]) = 2041)(s+3) = 0)$

### Loop Array C

int A[]; int i = 0; do {p = p\*A[i]; i++;} while (A[i]);

### Swap MIPS

```
void swap(int *v1, int *v2)
{
    int temp;
    temp = *v1;
    *v1 = *v2;
    *v2 = temp;
}
```

Parameters: \$2 = address of v1  
\$3 = v2  
Returns address in \$31

| Label | Instruction       | Comment     |
|-------|-------------------|-------------|
| Swap: | add \$3, \$3, 2   | # \$3 = v2  |
|       | add \$4, \$4, \$2 | # \$4 = v1  |
|       | lw \$5, 0(\$4)    | # \$5 = *v1 |
|       | lw \$6, 0(\$3)    | # \$6 = *v2 |
|       | sw \$5, 0(\$3)    | # *v1 = \$5 |
|       | sw \$6, 0(\$4)    | # *v2 = \$6 |
|       | j \$31            | # return    |

### Loop C to Mips

```
Sum = 0;
while (i != 0)
    Sum = Sum + i;
    i = i - 1;
```

| Label | Instruction        | Comment            |
|-------|--------------------|--------------------|
|       | addi \$0, \$0, 0   | # Sum = 0          |
| Loop: | beq \$1, \$0, Exit | # exit if i == 0   |
|       | addi \$3, \$1, 1   | # \$3 = i+1        |
|       | addi \$2, \$2, \$3 | # Sum = Sum + i    |
|       | ori \$1, \$1, 1    | # i = i - 1        |
|       | j Loop             | # continue looping |
| Exit: |                    |                    |



By theninja111

Published 28th April, 2014.  
Last updated 11th May, 2016.  
Page 1 of 1.

Sponsored by **Readable.com**  
Measure your website readability!  
<https://readable.com>