COMPSCI 210 S1T 2007

Aim for the tutorial:

In this tutorial, we will learn Memory Allocation Reading information in the window “Memory”, and how to use Arithmetic integer operate instructions with integer register.

1.2 Memory Allocation and Reading information in the window “Memory”

Memory allocation.

Allocate memory for string constants

Example1:

```
const
{
        align quad;
        message0:
        asciiz “hi”
        align quad;
        message1:
        asciiz “good”
}
```

Notes:

- Strings are stored in big endian
- One char is stored in one byte

Example2:

```
const
{
        align quad;
        message2:
        asciiz “Type some input”;
        align quad;
        message3:
        asciiz “ok\n”;
}
```

Notes:

- Allocated space is multiple of the size of the specified type (here is quad).

Example 3:

```
const
{
        align quad;
        message4:
        asciiz “question”
        align quad;
        message5:
        ascii “question”
}
```

Notes:

- Asciiz directive: an extra zero byte is allocated and added on the end

Example 4 (Allocate memory for variables and arrays):

```
abs{
        BUFFERSIZE = 10
}

data{
        align quad;
        buffer:
        byte [BUFFERSIZE + 1];
        align quad;
        value:
        quad 9;
}
```

Notes

• No initial variables are initialized as zero

Example 5:

```
entry main.enter;
import "../IMPORT/register.h";
import "../IMPORT/callsys.h";
import "../IMPORT/proc.h";
import "../IMPORT/callsys.lib.s";
import "../IMPORT/string.lib.s";
import "../IMPORT/number.lib.s";
import "../IMPORT/io.lib.s";

block main uses register, CALLSYS {
data{
        align quad;
        value0:
        quad 0xffffffffffffffff;

        align quad;
        value1:
        quad 0x55;

        align quad;
        value2:
        quad 0x78;

        align quad;
        value3:
        quad 0xffffffffff78;
}
```

```
code {
public enter:
        ldiq $t0, value0; // this command line just load the “value0” address to $t0
        ldq $t0, ($t0); // this command line will get the value of “value0” from the memory address
        ldiq $t1, value1;
        ldq $t1, ($t1);
        ldiq $t2, value2;
        ldq $t2, ($t2);
        ldiq $t3, value3;
        ldq $t3, ($t3);
        subq $t0, 1;
        addq $t1, 0x45;
        addq $t2, 0x94;
        addq $t3, 0x94;
        mulq $t1, 0x4;
        mov $t1, $a0;
        mov 16, $a1;
        bsr Number.toSigned.enter;
        mov $v0, $a0;
        bsr IO.print.enter;
        bsr IO.newline.enter;
        mov $t2, $a0;
        mov 16, $a1;
        bsr Number.toSigned.enter;
        mov $v0, $a0;
        bsr IO.print.enter;
        bsr IO.newline.enter;
        mov $t3, $a0;
        mov 16, $a1;
        bsr Number.toSigned.enter;
        mov $v0, $a0;
        bsr IO.print.enter;
        bsr IO.newline.enter;
    }
}
```

Answer:

$t0: ffff ffff ffff fffe

$t1 0x000000000000009a 0x0000000000000268

$t2 0x000000000000010c

$t3 0x000100000000000c

2. Exercise:

1. Assume the value of register T0 is 0. After the execution of the following instructions, show the value of these registers:

```
subq $T0, 1, $T1
addq $T0, 0x12
addq $T0, $T1, $T2
subq $T2, $T1, $T3
```

T0 = ?

T1 = ?

T2 = ?

T3 = ?

Answer:

T0 = 0x0000 0000 0000 0012

T1 = 0xffff ffff ffff ffff

T2 = 0x0000 0000 0000 0011

T3 = 0x0000 0000 0000 0012