CS241 Tutorial 3

Graham Cooper

May 29th, 2015

Topics

- Outputting bytes
- outputting isntructions
- MERL
- Relocating

Q1

Write psuedocode for a function output_word which takes a 32bit integer and outputs each of its 4 bytes to standard output. Assume the existence of a function output_byte which takes an int and outputs the loworder byte if the integer can be represented in 8 bits

```
Void output_word(int word){
char c;
c = word >> 24;
output_byte(c);
c = word >> 16;
output_byte(c);
c = word >> 8l
output_byte(c);
c = word;
output_byte(c);
}
```

```
0110
 >>1
----
0011

0110
 <<1
```

```
----
1100

1100
>>1
----
1110


1100
<<1
----
1000
```

Q2

Write a function assemble_add which assembles add instructions of the form add $d, $s, $t given d, s and t and returns an int representing the binary encoding of the add instruction.

```
int assemble_add(int d, int s, int t){
return  (s << 21) | (t < 16) | (d << 11) | 32;
}
```

MERL Mips executible relactable linkable
Stick notes:

- REL 0x1 loc
- ESD 0x5 val len
- ESR 0x11 loc len

Q3

```
beq $0, $0, 2
.word filelen
.word codelen
```

```
lis $6
.word 0x18
sw $31, -4($30)
lis $31
.word 4
sub $30, $30, $31
lis $3
Rel1:
.word proc

lis $11
.word 1

loop:
beq $2, $0, end
jalr $3
sub $2, $2, $11
beq $0, $0, loop

end:
add $3, $1, $0

lis $31
.word 4
add $30, $30, $31
lw $31, -4($30)
jr $31
proc:
add $1, $1, $6
jr $31

codelen:
.word 0x1
.word Rel1


file len:
```