Intro to Computer Science Assignment 9

Yiping Deng

2017-11-16

9.1

a)

#	Machine Code	Assembly Code	Description
0	001 1 0001	LOAD #0001	Load memory at 0001 into the acc
1	010 0 1111	STORE 1111	Store the acc value to memory 1111
2	001 1 0000	LOAD #0000	Load 0000 into the acc
3	101 1 0100	EQUAL #0100	If 0100 is equal to the acc, skip next instruction
4	110 1 0110	JUMP #0110	Jump to 0110
5	111 1 0000	HALT #0000	Stop the program,
6	001 0 0011	LOAD 0011	Load memory location 0011 into acc
7	100 1 0001	SUB #0001	Substract 0001 from the acc
8	010 0 0011	STORE 0011	Store acc to memory location 0011
9	001 0 1111	LOAD 1111	Load memory at 1111 to acc
10	011 0 1111	ADD 1111	Add memory at 1111 to acc
11	010 0 1111	STORE 1111	Store acc to memory at 1111
12	110 1 0010	JUMP #0010	Jump back to line 2
13	000 0 0000	NOP	Nothing happens
14	000 0 0000	NOP	Nothing happens
15	000 0 0000	NOP	Nothing happens

b)

1. Copy the value from memory 1 to memory 15
2. if the value at memory 3(the constant 0100, or 4) is is equal to 0, jump to line 6, elsewise stop the program
3. substact memory at 3 by 1
4. double the memory at 15
5. stop the program

c) the initial value of memory block 1 is binary 0001 = decimal 1, it is copied into memory 15, and later memory block 15 is doubled to 2.
then you jump back and doubled again, for 3 times. (memory block is decresed by 1 every iteration).
When it hits 0, it will terminate.
It means that it will be executed 4 times, in theory, you will have 1*2*2*2*2 = 16
It means it will overflow, eventually you will get 0

9.2

a) RAX, RBX, RCX, RDX, RBP, RSI, RDI, and RSP, R8, R9, R10, R11, R12, R13, R14, R15 are all the available register in x64 architecture
RAX is a 64-bit register
EAX is a 32-bit register, mapping to the lower 32-bit in RAX
AX is a 16-bit register, mapping to the lower 16-bit in EAX
AL is a 8-bit register, mapping to the lower 8-bit in AX
AH is a 8-bit register, mapping to the higher 8-bit in AX(bit 8 to bit 15)

b) The common addressing mode in x64 system is

1. Immediate: the value is stored in the instruction.
2. Register to register: the operand is a memory address.
3. Indirect: by 8-bit, 16-bit, 32-bit displacement along any general register for base and index. Can be multiplied by a factor of 1,2,4,8.
4. RIP-relative addressing: use a address relative to RIP register.

The stack pointer, RSP, RBP, is maintained in the register(in CPU), RSP pointer points to the stack top, and RBP pointer points to the stack base.

c)

```
    .global main ;make the symbol global so that other can access this main function
    .text ; tell the assembler it is text segment(instruction)
main:
    xor     %rax, %rax ; clear the rax register
    mov     %rax, %rbx ; copy the value of rax register into rbx register
.L1:
    add     $1, %rbx ; add 1 to rbx
    add     %rbx, %rax ; add value of rbx to rax
    cmp     $10, %rbx ; set condition flag by minus rbx - 10
    jne     .L1 ; jump to line L1 if rbx is not 10
    ret ; return the value in rax
```

The assembly code simply add nums from 1 to 10, and return. It does the following job:

```
int main(){
    int a = 0;
    int b = 0;
    do{
        b = b + 1;
        a = a + b;
    } while(b != 10);
    return a;
}
```

Eventually, it will return the result in %rax