Computer Organization: Homework 1

陈键飞

CST03 2010011291

2012 年 9 月 18 日

1 General

1.1 程序的结束

在模拟器中，代表程序结束的是RET（0xFFFF），但汇编器并不能产生这个代码。所以我们在每个代码的最后一行加上了一句NOP，然后用十六进制编辑工具手动修改成RET（0xFFFF）。

2 The Fibonacci Sequence

代码如下：

```
; Calculate the first 50 Fibonacci numbers. F1 = 1, F2 = 1
; The result is stored in 0x8500
LI R1 0x01
LI R2 0x01
LI R3 0x85
SLL R3 R3 0x00
LI R4 0x19
SW R3 R1 0x00
SW R3 R2 0x01
ADDU R1 R2 R1
ADDU R1 R2 R2
ADDIU R3 0x02
ADDIU R4 0xff
BNEZ R4 0xf9
```

程序运行结果如图 1，我们输出了0x33=51个数，前50个是Fibonacci，最后一个未被修改，所以为0。

3 ASCII Printable Characters

ASCII中可打印的字符集从0x20到0x7F。代码如下。

```
; Print all ascii printable characters: from 20 to 7E
; Author:        Jianfei CHEN
; Date:          2012−9−18


LI R6 0xBF                    ; R6 is the IO port
SLL R6 R6 0x00


LI R1 0x20                    ; Set initial value: print r1


loop:
SW R6 R1 0x00                 ; PRINT R1
ADDIU R1 0x01
CMPI R1 0x80
BTNEZ loop
NOP


finish:
JR R6                         ; Crash the machine
NOP
```

程序运行结果如图 2。

4 Lower Case Conversion

要将大写字母转为小写字母，只需将其加上0x20。

实验中，函数subRoutine从R1起连续R2个地址读取数据，将其输出后转为小写字母写回内存。由于模拟器未实现JALR，使用MFPC计算返回地址，存入R7；用JR R7来返回。

代码如下：

```
>>v 8500 33
        [8500] 0000000000000001    <0001>
        [8501] 0000000000000001    <0001>
        [8502] 0000000000000010    <0002>
        [8503] 0000000000000011    <0003>
        [8504] 0000000000000101    <0005>
        [8505] 0000000000001000    <0008>
        [8506] 0000000000001101    <000d>
        [8507] 0000000000010101    <0015>
        [8508] 0000000000100010    <0022>
        [8509] 0000000000110111    <0037>
        [850a] 0000000001011001    <0059>
        [850b] 0000000010010000    <0090>
        [850c] 0000000011101001    <00e9>
        [850d] 0000000101111001    <0179>
        [850e] 0000001001100010    <0262>
        [850f] 0000001111011011    <03db>
        [8510] 0000011000111101    <063d>
        [8511] 0000101000011000    <0a18>
        [8512] 0001000001010101    <1055>
        [8513] 0001101001101101    <1a6d>
        [8514] 0010101011000010    <2ac2>
        [8515] 0100010100101111    <452f>
        [8516] 0110111111110001    <6ff1>
        [8517] 1011010100100000    <b520>
        [8518] 0010010100010001    <2511>
        [8519] 1101101000110001    <da31>
        [851a] 1111111101000010    <ff42>
        [851b] 1101100101110011    <d973>
        [851c] 1101100010110101    <d8b5>
        [851d] 1011001000101000    <b228>
        [851e] 1000101011011101    <8add>
        [851f] 0011110100000101    <3d05>
        [8520] 1100011111100010    <c7e2>
        [8521] 0000010011100111    <04e7>
        [8522] 1100110011001001    <ccc9>
        [8523] 1101000110110000    <d1b0>
        [8524] 1001111001111001    <9e79>
        [8525] 0111000000101001    <7029>
        [8526] 0000111010100010    <0ea2>
        [8527] 0111111011001011    <7ecb>
        [8528] 1000110101101101    <8d6d>
        [8529] 0000110000111000    <0c38>
        [852a] 1001100110100101    <99a5>
        [852b] 1010010111011101    <a5dd>
        [852c] 0011111110000010    <3f82>
        [852d] 1110010101011111    <e55f>
        [852e] 0010010011100001    <24e1>
        [852f] 0000101001000000    <0a40>
        [8530] 0010111100100001    <2f21>
        [8531] 0011100101100001    <3961>
        [8532] 0000000000000000    <0000>
```

图 1: Runtime result of the Fibonacci Number program

```
>>c
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmno
pqrstuvwxyz{|}~△
```

图 2: Runtime result of the ASCII Printable Characters

```
LI  R6  0xBF              ; R6  is  the  IO  port
SLL  R6  R6  0x00


LI  R1  0xC0              ; R1  is  the  starting  address
SLL  R1  R1  0x00


LI  R2  0x05             ; read  5  numbers


LI  R3  0x41            ; Write  'APPLE'
SW  R1  R3  0x00
LI  R3  0x50
SW  R1  R3  0x01
LI  R3  0x50
SW  R1  R3  0x02
LI  R3  0x4C
SW  R1  R3  0x03
LI  R3  0x45
SW  R1  R3  0x04


MFPC  R7                 ; save  return  address  to  R7
ADDIU  R7  0x03
B  subRoutine            ; call  subRoutine
NOP


finish :
JR  R6                   ; Crash  the  machine
NOP


subRoutine :             ; Read  and  output  $R2  consequent  characters  starting  from

NOP
loop :
LW  R1  R3  0x00         ; Load  to  R3
SW  R6  R3  0x00         ; Print  to  screen
ADDIU  R3  0x20          ; R3  += 20  =  lower (R3)
```

4

```
SW R1 R3 0x00                ; Write back

ADDIU R1 0x01                ; Address++
ADDIU R2 0xFF                ; Count——

BNEZ R2 loop                 ; loop
NOP

JR R7                        ; return
NOP
```

在实验中，我们向C000开始的5个地址写入了字符串'APPLE'.
程序运行结果如图 3。

图 3: Runtime result of the Lower Case Conversion

5　Echo of Digit Characters

代码如下：

```
LI R6 0xBF
SLL R6 R6 0x00

loop:
LW R6 R1 0x00                ; Read a character

SLTI R1 0x30                 ; If R1 < 30 then finish
```
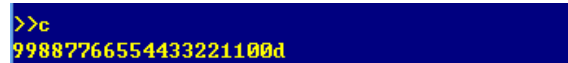
```
BTNEZ finish
NOP
SLTI R1 0x40              ; If R1 >= 40 then finish
BTEQZ finish
NOP

SW R6 R1 0x00             ; echo
B loop
NOP

finish:
JR R6
NOP
```

程序运行结果如图 4。

图 4: Runtime result of the Echo of Digit Characters

6 实验完成情况

7 实验体会