

CS061:

Machine Organization

& Assembly Language

Lab 8

Review Q 1

Which of the following does not set the flags (condition codes) for BR (branch)?

A) ST

B) LD

C) NOT

D) AND

Answer: ST

- Condition codes set by instructions that modify a register.
- ST does not *modify* any register.

Review Q 2

Which of the following does not transfer a value between memory and registers?

- A) LD
- B) STI
- C) STR
- D) LEA

Answer: LEA

- Puts the address of a label into a register.
- Address is known at assembly time and does not touch memory at all.

Review Q 3

Select all instructions that do not assemble:

A) AND R0, R0, #-16

B) ADD R0, R0, #16

C) BR xAB

D) JSR x3000

Why 11-bits and not 12-bits?

- Op-code = 4 bits.
- BUT JSR and JSRR have same opcode.
- 1 bit to tell them apart.

Answer:

- ADD and AND only accept a 5-bit 2s complement number.
 - 16 = 010000
- JSR only accepts 11-bit 2's complement number.
 - x3000 = 0011 0000 0000 0000

Agenda

1. Presentation:
 - a. Programming Assignment 4 Review
 - b. Printing out decimal numbers
 - c. Counting Bits + Parity Checking
 - d. Lab Descriptions
1. Work Time / Questions / Demos

PA 4

- Take in a 5-digit decimal number: "12345"
- Store it to a register.
 - E.g. R4 = 12345
- Concepts:
 - Input Validation: handling when users enter in incorrect input.
 - Number Parsing:
 - Converting digit character to digit. E.g. '9' (57) -> 9
 - Appending digit to a base 10 number. E.g. '32' -> $30 + 2$

Printing out Numbers

- How to print out numbers from a register?
 - The reverse of Programming Assignment 4!
 - E.g. R0 = 12345. Print out "12345"
- Say we have R0 = 13.
 - Must print left to right!
 - Print '1' first then print '3'!
 - Console will display it as "13"!
- Does that mean we need to get the 1 (from 13) first?
 - No, just have to *print* it first!
 - Can get the 3 (from 13) first!

Isolating Digits

- How to get the 1 (from 13)?

1		3
<hr/>		
10s		1s

- In 13, there is 1 instance of 10.
 - i.e. Can subtract 10 **once** from 13.
 - E.g. $13 - 10 = 3$ (Positive)
 - E.g. $13 - 10 - 10 = -7$ (Negative).
 - Tells you value of the 10s place!
- Can do this for all digits!
 - For 5 digits, there's 10,000s, 1,000s, 100s, 10s, 1s

Isolating Digits Ex.

Counter: 0

1	2	3
100s	10s	1s

Output: "" "1"

- Setup:

- Counter (to represent value in a digit place - e.g. 100s place, 10s place, etc)
 - Can reuse the counter too.
 - Initialize counter to 0.

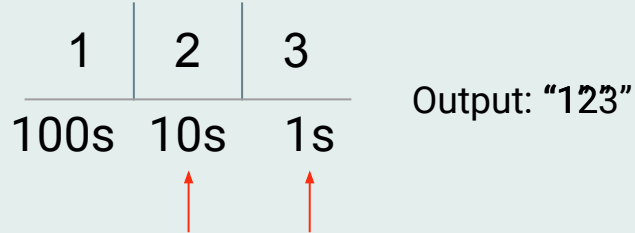
1. $123 - 100 = 23$

- Result not negative, so continue!
- Increment counter.

2. $23 - 100 = -23$

- Result negative!
- Hence there's only 1 in the 100s place!
- Print counter value! (Counter: 1)

Isolating Digits Ex.



- Reset counter! (Counter = 0)
- $23 - 10 = 13$
 - Counter++ (Counter = 1)
- $13 - 10 = 3$
 - Counter++ (Counter = 2)
- $3 - 10 = -7$
 - Negative (don't change counter)
 - Print counter value!

- Reset counter!
- $3 - 1 = 2$
 - Counter++ (Counter = 1)
- $2 - 1 = 1$
 - Counter++ (Counter = 2)
- $1 - 1 = 0$
 - Counter++ (Counter = 3)
- $0 - 1 = -1$
 - Negative (don't change counter)
 - Print counter value!

Division/Modulo Method

- Can divide/modulo by 10!
 - Need an extra stack for this!
- E.g. $123 / 10 = 12 \text{ rem } 3$
 - Push 3 onto the stack.
- $12 / 10 = 1 \text{ rem } 2$
 - Push 2 onto the stack.
- $1 / 10 = 0 \text{ rem } 1$
 - Push 1 to stack
- How to divide by 10?
 - No instruction in LC-3!
 - Can find out how many times you can subtract the number by 10!
- E.g. $123 - (10 * x) = y$
 - $x = 12, y = 3$

Pop off stack to print number in order:

- Pop and print (Prints 1) Console: "1"
- Pop and print (Prints 2) Console: "12"
- Pop and print (Prints 3) Console: "123"

Counting Bits

- How to count the number of 1s (or even 0s) in a binary number?
 - E.g. in **11011010** = 5 bits that are 1s.
- Very similar to Programming Assignment 3!

Remember this?

Pseudocode:

```
for(i = 15 downto 0):  
    if (msb is a 1):  
        print a 1  
    else:  
        print a 0  
    shift left
```

- Shift Left Review:
 - 0101 (5) -> 1010 (10)
 - Adding a number to itself will bit shift it to the left!
- Instead of printing, just increment a counter if the MSB is a 1!

Why count bits?

- Counting bits is useful in something called “**parity checking**”!
 - **Parity**: Being even or odd.
- Parity checking is good for **error detection**.
- Suppose we're trying to send 8-bits across a network:
 - E.g. sending '11001101'
- How to know if the bits were sent correctly?
 - Can send the bits + a parity bit:
 - E.g. '11001101' and '1' = '11001101**1**'
- Count the parity bit in the number:
 - '110011011' has 6 ones.
 - If parity count is even, then its correct (in the case of even parity)!

Character	Sender	Parity Bit	Receiver	Parity
"E"	1000101	1	1000101 1	Even
"A"	1000001	0	1000001 0	Even
"C"	1000011	1	1110011 1	Even
"q"	1110001	0	1110000 0	Odd Error!

Exercise 1

- Do the reverse of assignment 4!
- Two sub-routines.
- LOAD_FILL_VALUE_3200:
 - Load a hardcoded value into a register!
- OUTPUT_AS_DECIMAL_3400:
 - Print out register value to console!
 - See previous slides for help!
 - **Must handle negative numbers!**
- Test Harness (Main Program):
 - Call LOAD_FILL_VALUE_3200 sub-routine.
 - Add 1 to the value.
 - Call OUTPUT_AS_DECIMAL_3400 sub-routine.

Exercise 2

- Count number of 1s in a binary number!
- Create a sub-routine:
 - Parameter (Rx): Value to count.
 - Post-condition: Count number of 1s in the binary form of Rx.
 - Return Value (Ry): Number of 1s.
- Test Harness (Main Program):
 - Ask user to input a character.
 - Call sub-routine on the ASCII value of the character entered (e.g. 'A' -> 65)
 - **Print out “The number of 1’s in ‘<char>’ is: x”**
 - E.g. “The number of 1’s in ‘A’ is: 2”

Exercise 3

- How to right-shift a value?
- No coding required for this exercise: just think of the algorithm!
- Right-shifting:
 - '1011' -> '0101'
- Can we:
 - Subtract the number from itself? *Nope*
 - Can we divide the number by 2? *Yes - but **impractical** for large numbers.*
- Can we left-rotate?
 - Left-rotating: Left-shifting but adding MSB back to number.
 - E.g. '1011':
 - MSB is '1'
 - Left-shift: '0110'
 - Add MSB: '0111'
 - How can we use left-rotating to right shift?

Demo Info

- **Lab Grade Breakdown:**
 - 3 points for attendance.
 - 7 points for demoing (+1 bonus point demo'd before/during Friday).
 - 3 point penalty if lab is demo'd during the next lab session.
- **Tips before you demo:**
 - ***Understand your code!*** (Know what each line does & the input/output)
 - ***Test your code!*** (Check for correct output and that there are no errors)