CS2100: Computer Organisation

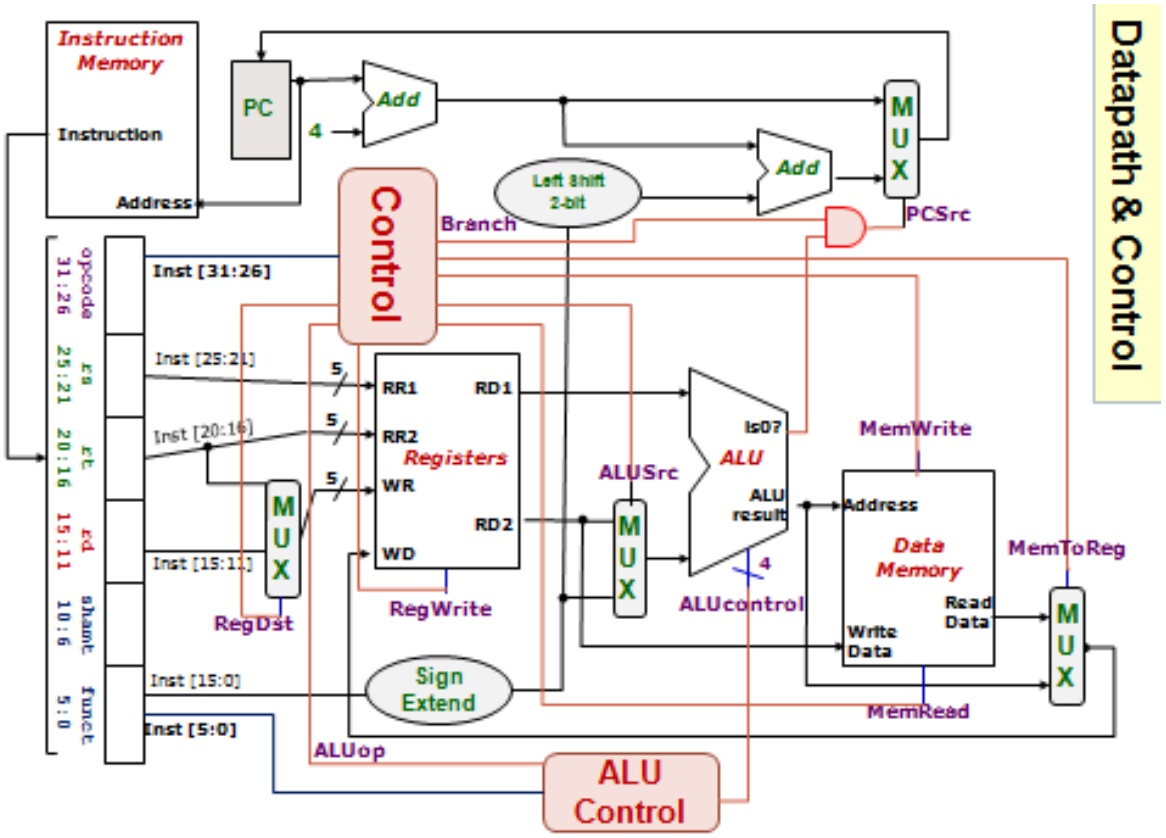
Tutorial #1: Number Systems and C

Answers

Tutors:

- Please do NOT reveal answers to students. You may ask them other questions to test their understanding. I will post selected answers on IVLE after the tutorials.
- Please encourage students to participate in the IVLE forums, and help to answer their questions on the forums.
- Since this is the first tutorial, I've made this short to give you time to introduce yourself to your students and interact a bit with them.
- Remember to mark attendance and enter the attendance on google worksheet.

1. Sign extension



You will learn about the above datapath diagram later in this module. ☺ Can you spot the "Sign Extend" unit in the diagram? This unit *sign extends* a 16-bit complement number to a 32-bit number. Sign extension works in the same way for both 1's complement and 2's complement. In the above diagram, 2's complement is used. In the question below, we will use 1's complement instead.

When we port values to a wider representation (from n bits to m bits, where $n < m$), we can retain the value by a simple strategy called ***sign extension***. Find out how sign extension works and show how you would sign-extend the following 4-bit 1's complement numbers to 8-bit 1's complement numbers without changing their original values: $(0101)_{1s}$ and $(1001)_{1s}$.

Answers:
Sign extension is done by padding the extra leading bits with the same leading bit of the original value.
$(0101)_{1s}$ → $(00000101)_{1s}$ [value is **5**]
$(1001)_{1s}$ → $(11111001)_{1s}$ [value is **–6**]

2. We generalize (r – 1)'s-complement (also called radix diminished complement) to include fraction as follows:

(r – 1)'s complement of $N = r^n - r^{-m} - N$

where n is the number of integer digits and m the number of fractional digits. (If there are no fractional digits, then m = 0 and the formula becomes $r^n - 1 - N$ as given in class.)

For example, the 1's complement of 011.01 is $(2^3 - 2^{-2}) - 011.01 = (1000 - 0.01) - 011.01 = 111.11 - 011.01 = 100.10$.

Perform the following binary subtractions of values represented in 1's complement representation by using addition instead. (Note: Recall that when dealing with complement representations, the two operands must have the same number of digits.)

Is sign extension used in your working? If so, highlight it.

Check your answers by converting the operands and answers to their actual decimal values.

(a) 0101.11 – 010.0101 (b) 010111.101 – 0111010.11

Answers:
(a) 0101.1100 – 0010.0101 → 0101.1100 + 1101.1010 → **0011.0111_{1s}**
(Check: 5.75 – 2.3125 = 3.4375)

(b) 0010111.101 – 0111010.110 → 0010111.101 + 1000101.001 →
1011100.110_{1s} = -0100011.001_2
(Check: 23.625 – 58.75 = –35.125)

Note that sign-extension is used above.

Note that two trailing zeroes are added. (This is not sign extension.)

Note that a leading zero is added. This is sign extension.

3. [Adapted from AY2018/2019 Semester 2 Term Test #1]
Given the following hexadecimal value in the IEEE 754 single-precision floating-point number representation:

0x B D A 0 0 0 0 0

What decimal value does it represent?

Answer: **-0.078125**

B D A 0 0 0 0 0	*write down the value*
1011 1101 1010 0000 0000 0000 0000 0000	*convert to binary*
1 01111011 01000000000000000000000	*grouped based on IEEE 754*
Sign = 1_2 = negative	*check sign bit*
Exponent = $01111011_2 = 123_{10} = 123_{10} - 127_{10} = -4_{10}$	*excess 127*
Mantissa = $01000000000000000000000_2$	*get mantissa value*
Value = $- 1.01000000000000000000000_2 \times 2^{-4}$	*add hidden leading bit*
$= - 0.000101_2$	
$= - 0.078125_{10}$	*by repeated multiplication*

4. Given the partial C program shown below, complete the two functions: **`readArray()`** to read data into a character array (with at most 10 elements) and **`isPalindrome()`** to check if the given array is a palindrome. If the sequence is a palindrome, return 1, otherwise return 0. A sequence of character is a palindrome if they read the same forward and backward. For example, "racecar" is a palindrome. For **`isPalindrome()`**, you are to provide two versions: an iterative version and a recursive version. For the recursive version, you may write an auxiliary/driver function to call the recursive function.
Challenge: Can you do it without auxiliary/driver function?

```
#include <stdio.h>
#define MAX 10

int readArray(char [], int);
int isPalindrome(char [], int);
void printArray(char [], int);

int main(void) {
  char array[MAX];
  int numElem, result;

  numElem = readArray(array, MAX);
  result  = isPalindrome(array, numElem);
  if (result == 0) {
    printf("'%s' is not a palindrome\n", array);
  } else {
    printf("'%s' is a palindrome\n", array);
```

```
  }

  return 0;
}

int readArray(char arr[], int limit) {
  // ...
  printf("Enter up to %d characters, terminating with a
whitespace ' '.\n", limit);
  // ...
}

int isPalindrome(char arr[], int size) {
  // ...
}
```

Answers:

```
int readArray(char arr[], int limit) {
  int i;
  char input;

  printf("Enter up to %d characters, terminating with a
whitespace ' '.\n", limit);
  i = 0;
  scanf("%c", &input);
  while (input != ' ') {
    arr[i] = input;
    i++;
    scanf("%c", &input);
  }
  arr[i] = 0;

  return i;
}
```

```
// Iterative Version
int isPalindrome(char arr[], int size) {
  int l_idx = 0, r_idx = size-1;

  while (l_idx < r_idx) {
    if (arr[l_idx] != arr[r_idx]) {
```

```
      return 0;
    }
    l_idx++;
    r_idx--;
  }

  return 1;
}
```

```
// Recursive Version
int isPalindrome(char arr[], int size) {
  int l_idx = 0, r_idx = size-1;

  if (size <= 0) {
    return 1;
  } else if (arr[l_idx] != arr[r_idx]) {
    return 0;
  } else {
    return isPalindrome(arr+1, size-2);
  }

  return 1;
}
```

5. When you try to compile the following code, you encounter an error saying "`prog.c:13:8: error: unterminated comment`".
   (a) What is the problem with the code? How would you correct the error?
   (b) Trace the program manually (*do not run it on a computer*) and write out its output. When you present your solution, draw diagrams to explain.

```
#include <stdio.h>

int main(void) {
  int a = 3, *b, c, *d, e, *f;

  b = &a;
  *b = 3;
  c = *b*5;
  d = b;
  e = *b+c;
```

```
  *d = c+e;
  f = &e;
  a = *b/*f;
  *f = *d-*b;

  printf("a = %d, c = %d, e = %d\n", a, c, e);
  printf("*b = %d, *d = %d, *f = %d\n", *b, *d, *f);

  return 0;
}
```

Remember to post on the LumiNUS forum if you have any queries.

Answers:

(a) At line 13, we have `*b/*f`. The symbols `/*` indicates the start of a comment. We will need to put a space between the two symbols to make it `*b / *f`.
(b)The output is as follows

```
a = 1, c = 15, e = 0
*b = 1, *d = 1, *f = 0
```