Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Department of Computer Science

Computer Architecture II
CSU34021

Tutorial 3
RISC-I Instruction Set Architecture

Syed Asad Alam

Document History

Rev.	Date	Comment	Author
0.1	01-11-2020	Initial Draft	SAA
0.2	08-11-2020	Added some assumptions	SAA
0.3	10-11-2020	Added details	SAA & IK
1.0	10-11-2020	Tutorial released	SAA

1 Learning Outcomes

This lab satisfies the following learning outcomes of the course:

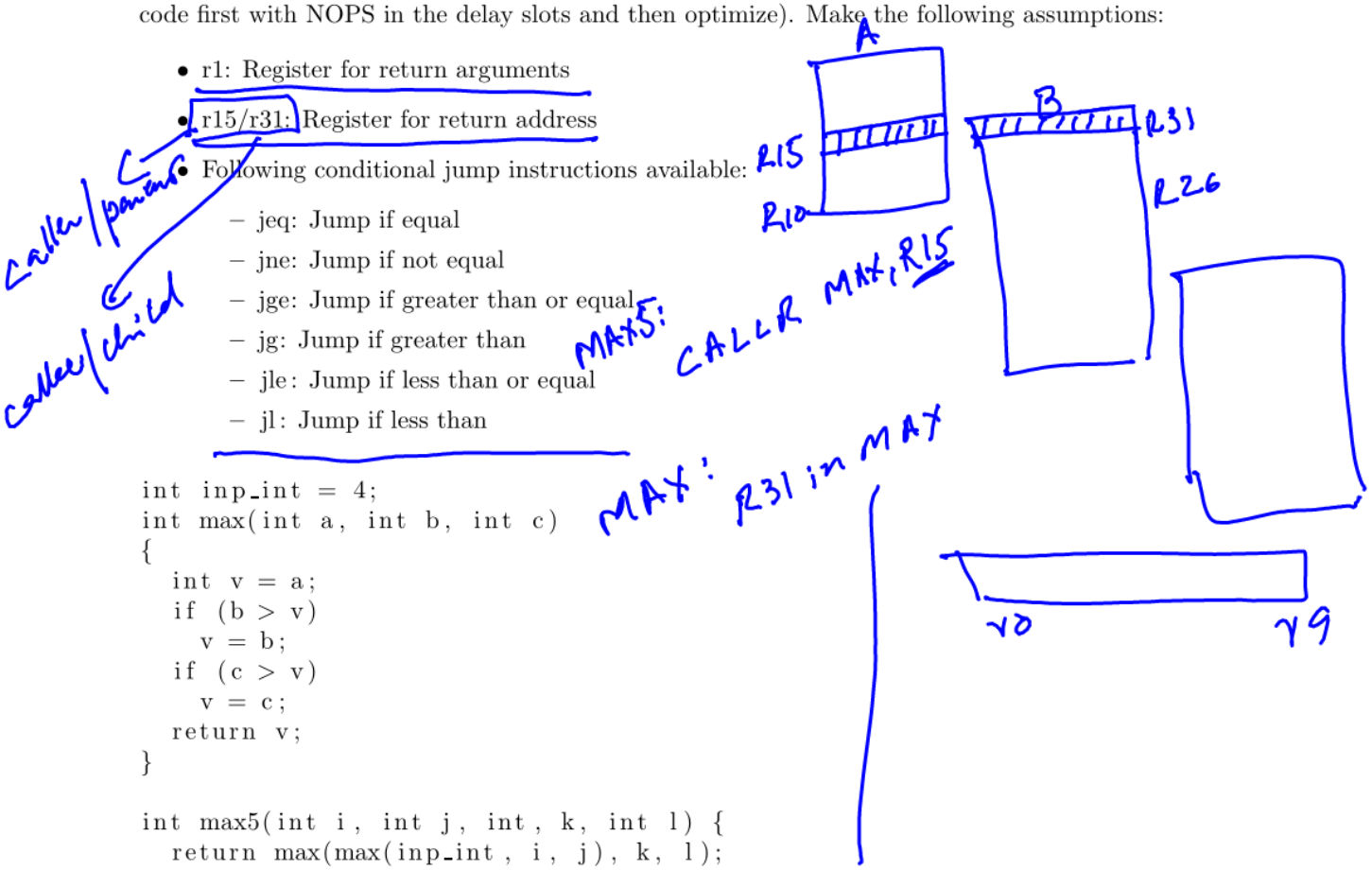
LO4 Evaluate and write RISC-I assembly

2 Questions

Question 1:

Translate the following C/C++ code segment into RISC-I assembly language (hint: generate unoptimized code first with NOPS in the delay slots and then optimize). Make the following assumptions:

- r1: Register for return arguments
- r15/r31: Register for return address
- Following conditional jump instructions available:
  - jeq: Jump if equal
  - jne: Jump if not equal
  - jge: Jump if greater than or equal
  - jg: Jump if greater than
  - jle: Jump if less than or equal
  - jl: Jump if less than




```
int inp_int = 4;
int max(int a, int b, int c)
{
  int v = a;
  if (b > v)
    v = b;
  if (c > v)
    v = c;
  return v;
}

int max5(int i, int j, int, k, int l) {
  return max(max(inp_int, i, j), k, l);
}
```

Question 2:

Translate the following C/C++ code segment into RISC-I assembly language (hint: generate unoptimized code first with NOPS in the delay slots and then optimize). Make the following assumptions:

- r1: Register for return arguments
- r15/r31: Register for return address
- Following conditional jump instructions available:
  - jeq: Jump if equal
  - jne: Jump if not equal
  - jge: Jump if greater than or equal

- jg: Jump if greater than
- jle: Jump if less than or equal
- jl: Jump if less than

Also, assume there are subroutines available to calculate the modulus and divide(you do not need to implement it). The subroutines are to be called with appropriate arguments.

```
int fun(int a, int b)
{
    if (b == 0)
        return 0;
    if (b % 2 == 0)
        return fun(a + a, b/2);

    return fun(a + a, b/2) + a;
}
```

Question 3:

Consider the following program:

```
int compute_pascal(int row, int position)
{
  if(position == 1)
    {
      return 1;
    }
  else if(position == row)
    {
      return 1;
    }
  else
    {
      return compute_pascal(row-1, position) + compute_pascal(row-1, position-1);
    }
}
```

Determine, by instrumenting the above code, the number of procedure calls, maximum register window depth, the number of register window overflows and the number of register window underflows that would occur during the calculation of compute_pascal(30,20) given a RISC-I processor with 6, 8 and 16 overlapping register windows. Also calculate the number of register windows pushed onto the stack for each case of overlapping register windows.

Assume that overflow only occurs when **all** register windows have been utilized and underflow occurs when less than **two** register windows are left active. In other words, the minimum number of active register windows is **two**.

Modify your code with the assumption that overflow occurs when there is one empty register window and compare the two approaches.

Important to note is that you are NOT being asked to translate the compute_pascal function into RISC-1 code.

Question 4:

Determine how long it takes to calculate compute_pascal(30, 20) on your computer. Make sure you time the release version of your code. Briefly describe your approach and comment on its accuracy.

Submission

For the questions where an actual implementation is required on a computer, you can use both Microsoft Visual Studio, GCC or any other compiler that you are most comfortable with.

For the RISC-I codes, either type it in an editor or on a piece of paper where it should be legible. For questions 3 and 4, you also need to submit your program as well as screenshots of the outputs. It may even be one program and one screen shot showing all the outputs required. Comments and descriptions should be in the form of a document. You can, if you so wish, create a small report where you paste your code for all questions along with screen shots and comments. But even then, you need to submit the code for Q3/Q4 separately so that we can run it to verify.

In what ever form you decide to submit your response, make sure you eventually submit a single file which may be a report or a .zip file containing all pieces of your submission. It is very important that you name your file in way that identifies you and submit it on Blackboard.

Marks Distribution

This coursework will be marked out of 100. Question 1, 2 and 4 carry 20 marks each while Question 3 carry 40 marks.

Deadline

The deadline is: Monday, November 23, 2020 9 pm.