CSCU9A2 Practical 2B (Week 3) Spring 2017

Brookshear Machine Programming (2 February)

Remember to register your practical attendance at the start of each session:

- **Double click on the My Computer icon, then on Groups on Wide (the V: drive), then on CSCU9A2, and double click on the Register icon.**

If you get stuck or need help at any time during the practical, ask a demonstrator.

Remember that these are teaching sessions, and during the practicals you should not be surfing the Web, nor texting nor chatting online with your friends.

Remember: This week's checkpoints must be completed by the end of week 5.

THIS WORKSHEET:

In this practical you will practice using a very simple Brookshear machine simulator. In the second part you can try your skill writing a Brookshear program for yourself.

A BROOKSHEAR MACHINE EXERCISE

- Launch a browser and visit page **http://joeledstrom.github.io/brookshear-emu/**
  This shows a simple Brookshear Machine simulator:
  - The main memory is in the centre, with byte address 00 at the top left, 01 next to it on the right, 02 next and so on. If you hover the mouse over a memory location, the Inspector shows you its contents. You can alter the contents of a memory location by highlighting it with the mouse, typing two hexadecimal digits and then **Enter** – the highlight moves automatically to the next location ready for altering that.
  - On the far left the contents of the 16 general purpose registers (GPRs) are displayed. They cannot be edited.
  - Next to the registers you can see the Program Counter (PC), Instruction Register (IR), and various control buttons.
  - To use the Brookshear Machine, you must enter machine instructions from memory address 00 onwards, and then the CPU should be **Clear**ed, which puts 00 into the PC, and then you click either **Step** or **Run**. **Run** allows the Brookshear Fetch-Execute cycle to run freely until a Halt instruction is encountered. Clicking **Step** repeatedly steps the Brookshear Machine through the stages of the Fetch-Execute cycle so that you can see what is happening.
- The Brookshear Machine reference sheet is attached, or can be found on the module's Lecture slides web page.
- Continued over page…

- Here is a simple program to add the two numbers in memory locations 80 and 81 together and leave the result in location 82: The memory address is on the left, the assembly code is on the right and the corresponding hexadecimal bytes are in the middle:

```
00    1A80          MOV [80] -> RA
02    1B81          MOV [81] -> RB
04    5CAB          ADDI RA, RB -> RC
06    3C82          MOV RC -> [82]
08    C000          HALT
```

- Enter the hexadecimal codes into the memory bytes from memory address 00: So 1A into address 00, 80 into the next location, 1B into the next, 81 the next, and so on.
- Before running the program, it needs some data to work on, so enter, say, 03 into memory location 80, and 06 into location 81. [So the eventual result should be 09 in location 82.]
- Click **Clear** and then **Run**. Things happen quickly, and fairly soon the machine halts with the correct result in location 81 – check that it is correct!
- Alter locations 80 and 81 back to 03 and 06.
- Click **Clear** (the PC goes back to 00), and then this time click **Step** repeatedly, but quite slowly – after each click look carefully at the CPU: at the PC, the IR and the contents of the registers A, B and C. Make sure that understand what is happening. If you cannot follow what is going on, then ask for help.
- Here is another Brookshear Machine program, this time in assembly language:

```
00    MOV 01 -> RA
02    MOV [80] -> RB
04    ADDI RA, RB -> RC
06    MOV RC -> RB
08    MOV RC -> [80]
0A    JMPEQ 04, R0
0C    HALT
```

- Work out what this program does. Why is the HALT instruction not really necessary?
- Translate ("assemble") the program into hexadecimal codes, and enter it into the memory.
- Clear the CPU and run or step through the program. Watch carefully, and check whether your understanding of the program was correct.

Checkpoint [Brookshear]:

Show a demonstrator your translated code for the second Brookshear program, explain what it does, and demonstrate the program running. Answer any questions they ask you.

MORE ON THE NEXT PAGE…

WRITING YOUR OWN BROOKSHEAR PROGRAM

- Here is the description of a task computable by the Brookshear Machine:
  - There will two numbers in memory locations 80 and 81. Let's refer to them as m and n. The number m *must* be less than the number n – for example 01 and 09.
  - The program is to compute a result and store it in memory location 82.
  - The result should be the sum of the numbers from m to n inclusive:
    m + (m+1) + (m+2) + …+ n
    For example, with m is 01 and n is 04 the result should be A0 (that is ten).
- Write a Brookshear program to carry out this computation. The first instruction should be at address 00, and there should be a HALT instruction at the end. You should write the program in Brookshear assembly language and then translate it.

  Hints:
  - Work it out on paper first!
  - Think about using a register to count up from m to n, adding its value onto the contents of location 82 at each step.
  - Location 82 will need to be started at 00! (Why?)
  - Counting up needs a loop *very similar* to the second Brookshear program in the first part of this lab sheet.
- Translate ("assemble") the program into hexadecimal codes, and enter it into the memory.
- Enter your program into the Brookshear Machine. Clear the CPU, insert some initial data into locations 80 and 81, and run or step through the program. Watch carefully, and check whether your program is correct!

That's all for today.

SBJ 27 January 2017