ICS 2022 Problem Sheet #10

**Problem 10.1:** assembler programming

The following program has been written for the simple central processing unit introduced in class. The table below shows the initial content of the 16 memory cells. The first column denotes the memory address and the second column shows the memory content in hexadecimal notation.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cell | Hex | Binary | Assembler | Description |
| 0 | 2f | 001 0 1111 | LOAD 15 | Load the value of memory location 15 into the accumulator |
| 1 | 6a | 011 0 1010 | ADD 10 | Add the value of memory location 10 to the accumulator |
| 2 | 4f | 010 0 1111 | STORE 15 | Store the value of the accumulator in memory location 15 |
| 3 | 21 | 001 0 0001 | LOAD 1 | Load the value of memory location 1 into the accumulator |
| 4 | 71 | 011 1 0001 | ADD #1 | Add the value 1 to the accumulator |
| 5 | 41 | 010 0 0001 | STORE 1 | Store the value of the accumulator in memory location 1 |
| 6 | a9 | 101 0 1001 | EQUAL 9 | Skip instruction if accumulator equal to memory location 9 |
| 7 | d0 | 110 1 0000 | JUMP #0 | Jump to instruction 0 (set program counter to 0) |
| 8 | e0 | 111 0 0000 | HALT | Stop execution |
| 9 | 6f | 011 0 1111 | ADD 15 | Add the value of memory location 15 to the accumulator |
| 10 | 01 | 000 0 0001 |  |  |
| 11 | 02 | 000 0 0010 |  |  |
| 12 | 03 | 000 0 0011 |  |  |
| 13 | 04 | 000 0 0100 |  |  |
| 14 | 05 | 000 0 0101 |  |  |
| 15 | 06 | 000 0 0110 |  |  |

1. Convert the machine code from hexadecimal notation into binary notation.
2. Write down the assembler code for the machine code. Add meaningful descriptions.
3. The program leaves a result in memory cell 15 when it halts. What is the value? Explain how the program works.

After it halts the program leaves the result 21 in memory cell 15, because it loops 5 times and adds something every time, so 6 (the original value stored in memory cell 15) + 1(the value stored in memory cell 10) + 2(the value stored in memory cell 11) + 3(the value stored in memory cell 12) + 4(the value stored in memory cell 13) + 5(the value stored in memory cell 14) = 21.

The program works at it follows: it will increase the value of memory location 1, originally 6a, with 1 every time it loops until it will be equal with the value of memory location 9, that is 6f, so the program will loop 5 times, while it will also add the values in memory cells 10,11,12,13 and 14 (as the value from memory cell 1 increases, from 6a to 6f) to the value in memory cell 15.

1. What happens if the value stored in memory cell 9 is changed to 0x70 before execution starts? Explain.

It the value stored in memory cell 9 is changed to 0x70 before execution starts it means that the value in the memory location 1 will have to increase to 70 instead of 6f, so one more time, so the program will loop one additional time, so the value in memory cell 15 will also increase one more time, meaning that the program would leave the result 42 instead of 21 in memory cell 15, as it will also add the value stored in its memory cell.

**Problem 10.2:** integer multiplication in risc-v rv32i assembler

The 32-bit RISC-V base integer instruction set (rv32i) does not support multiplication and division operations. To deal with this, a compiler may call a function when a multiplication is needed. For example, gcc expects that a function mulsi3(unsigned int a, unsigned int b) is provided to multiply two integers. A multiplication can be carried out by repeated additions and shifts:

unsigned int mulsi3 (unsigned int a, unsigned int b)

{

unsigned int r = 0;

while (a) {

if (a & 1) {

r += b;

}

a >>= 1;

b <<= 1;

}

return r;

}

1. Translate the above C code into equivalent RISC-V rv32i assembler code. Comment the assembler code to explain how the calculation proceeds. Note that the arguments are passed via the registers a0 (x10) and a1 (x11) and that the result is returned in a0 (x10).
2. Does the function need a function call prolog and a function epilog? Explain why or why not.