Ahmed Almass

ITSC204
Computer Architecture – Exploitation and Security
Module 1: Architecture of General Purpose Processor
Lab Manual

Objectives

- Review number systems
- Describe the building blocks of a processor.
- Define Harvard and von Neumann architectures
- Discuss the advantages and disadvantages of Harvard and von Neumann architectures.
- Define RISC and CISC architectures.
- Discuss the advantages and disadvantages of RISC and CISC architectures.
- Explain the interaction among the building blocks.
- Describe byte, word, doubleword, and quadword.
- Perform processor-activity simulation (by hand).

1.

Background Reading

Notes common to all lab and home assignment problems

For every lab and home assignment, all work should go into your personal repository, subdirectory named mXX, where XX stands for the module number. For each problem, carefully name the program as described. The programs are extracted from your repository by a Python script, and errors in the program name will result in the instructor never seeing your program, and your mark for it will be ZERO!

There are always many ways how to solve a programming problem, and usually one or two ways which are fast, compact and elegant.

Make sure to push your work to the server often, and have pushed the working version of the program by the deadline specified. The script extracting your programs from your repository will be run at any time after the deadline.

Lab specific intro

You can use any decimal calculator. Calculators capable of hexadecimal or any base-2 operations are not allowed.

Table of powers of 2

0	1	0000 0000 0000 0001
1	2	0000 0000 0000 0010
2	4	0000 0000 0000 0100
3	8	0000 0000 0000 1000
4	16	0000 0000 0001 0000
5	32	0000 0000 0010 0000
6	64	0000 0000 0100 0000
7	128	0000 0000 1000 0000
8	256	0000 0001 0000 0000
9	512	0000 0010 0000 0000
10	1024	0000 0100 0000 0000
11	2048	0000 1000 0000 0000
12	4096	0001 0000 0000 0000
13	8192	0010 0000 0000 0000

14	16384	0100 0000 0000 0000
15	32768	1000 0000 0000 0000
16	65536	1 0000 0000 0000 0000

Typically, decimal numbers are written in the ordinary form, binary numbers are prefixed by "0b", hexadecimal numbers are prefixed by "0x" or postfixed with letter "h", and octal numbers have a leading zero.

Problem 1

Perform the following number operations:

Operand 1	Operand 2	Operation	Result
1	1	ADD	2
8	9	ADD	17
6144	1056	SUB	5088
60855	4680	OR	65535
60855	4680	XOR	65535
0xedb7	0xc000	SUB	2DB7
255	255	AND	255
127	127	OR	127
63	127	XOR	64
4096	256	AND	0

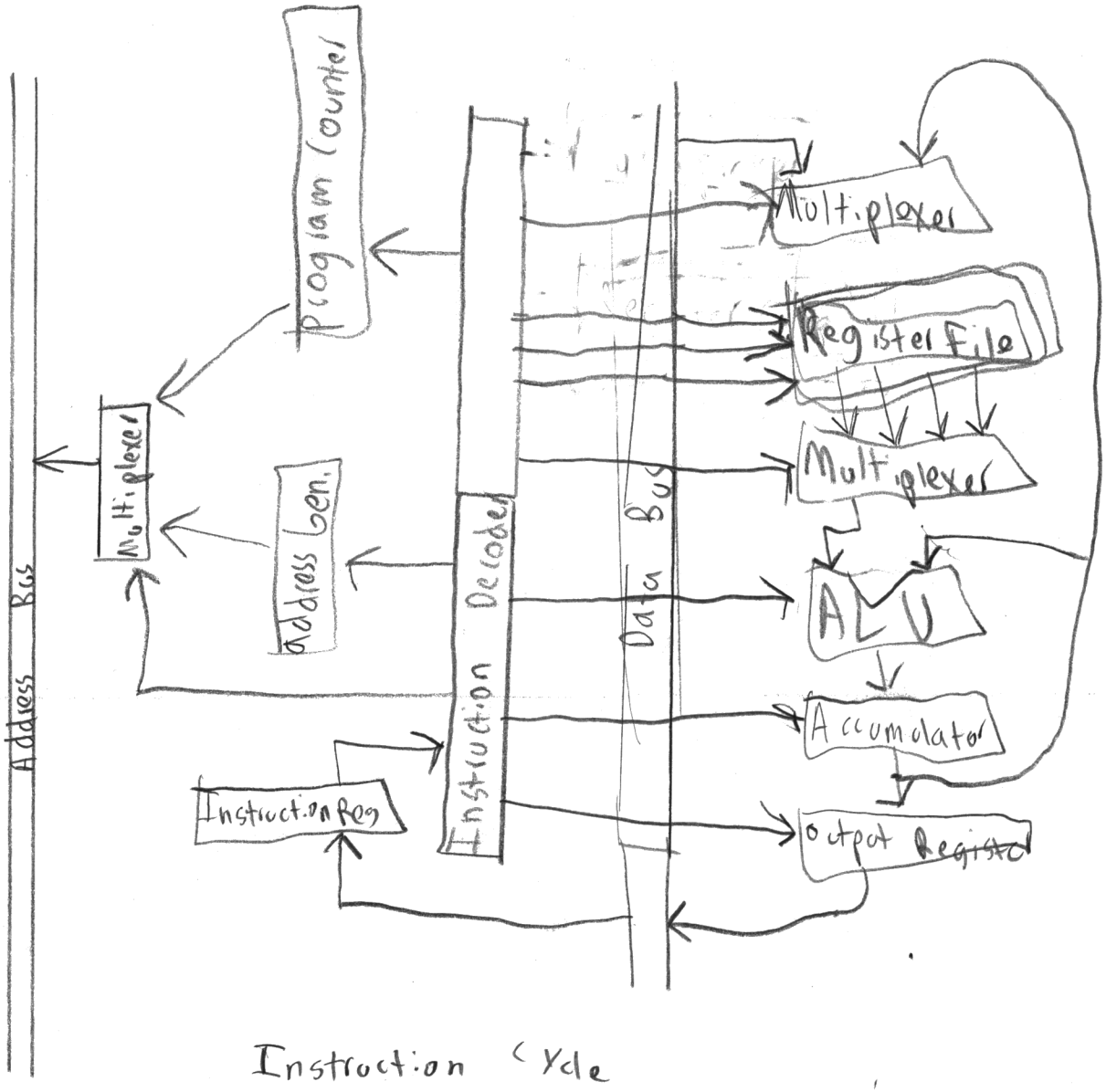
Problem 2

Perform the following number conversions:

Number to convert	Binary	Octal	Hexadecimal	Decimal
0x1400	1 01 0000000000	12000	0x1400	5120
64	100000	100	0x40	64
0x248	0000 0000 0010 0100	1110	0x248	584
0b1101001101101100	110 100 110 110 100	151554	0xD36C	54124

Problem 3

Draw a block diagram of minimalist general purpose processor,:

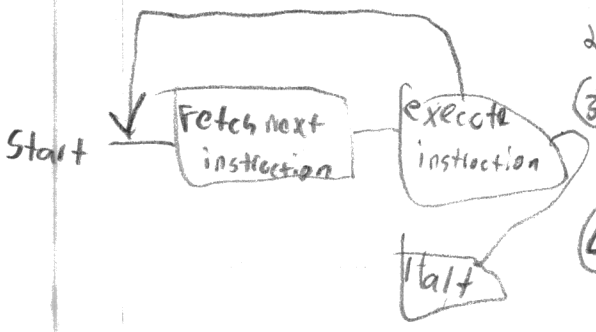


Instruction Cycle

Pg.

1. Fetch & instruction from memory
2. Decode the instruction
3. read effective address from memory, if the instruction has an indirect address
4. execute the instruction

store result (write back memory data)



Problem 4

Describe the individual data movement steps executing an instruction which loads data from memory to register, using the content of another register as an address.

Problem 5

Using the instruction notation OPERATION DESTINATION,SOURCE1,[SOURCE2], and the following form as master for a spreadsheet **`m01p051ab.csv`**, perform hand simulation of following sequence of generic instructions:

```
mov   r1,[adr1]
mov   r2,[adr2]
add   r1,r1,r2
mov   r3,#80
mul   r1,r1,r3
mov   r4,#123
mov   [r4],r1
```

r1 = r1+r2

Generate random numbers for the initial values of ip, adr1, adr2, and the memory content at those addresses.

CLK	IP	INSTRUCTION	ADR	DATA	R6	R5	R4	R3	R2	R1	R0
1											
2											
3											
4											
5											
6											
7											
8											
9											
10											
11											
12											
13											
14											
15											
16											
17											
18											
19											
20											
21											
22											
23											
24											
25											

Problem 6

Download the file **`m01p06lab.csv`** from your git repository and calculate the results in hexadecimal. The resulting lines should have 4 columns and follow the examples:

Operand1	Operand2	Operation	Result
123	5	ADD	0x80

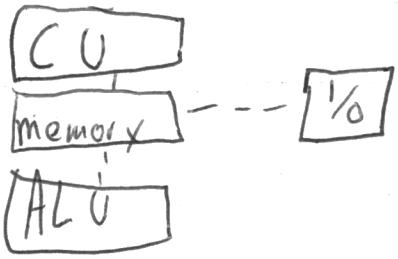
Problem 7

Describe how the Generic CPU would have to be modified to conform to the Von Neumann architecture.

Problem 8

Describe how the Generic CPU would have to be modified to conform to the Harvard architecture.

#7. Generic CPU would have to be modified to allow control unit & ALU connect to common memory to be Von architecture.



(8) The Havard Architecture changes the generic CPU to have two seperate memories. one for programing