Implementation and Analysis of SCU ISA

COEN 210: Aman Nagarkar, Priyanshi Shah, Srinagesh Keerti

Table of Contents

Introduction

Design a 32-bit pipelined CPU for the given SCU Instruction Set Architecture (SCU ISA) to find the maximum of *n* numbers described below:

MAX = max{a_1, a_2,..., a_n}

Input: Array A = [a_1, a_2,..., a_n]
Output: Max of A_i for i = 1,2,...,n

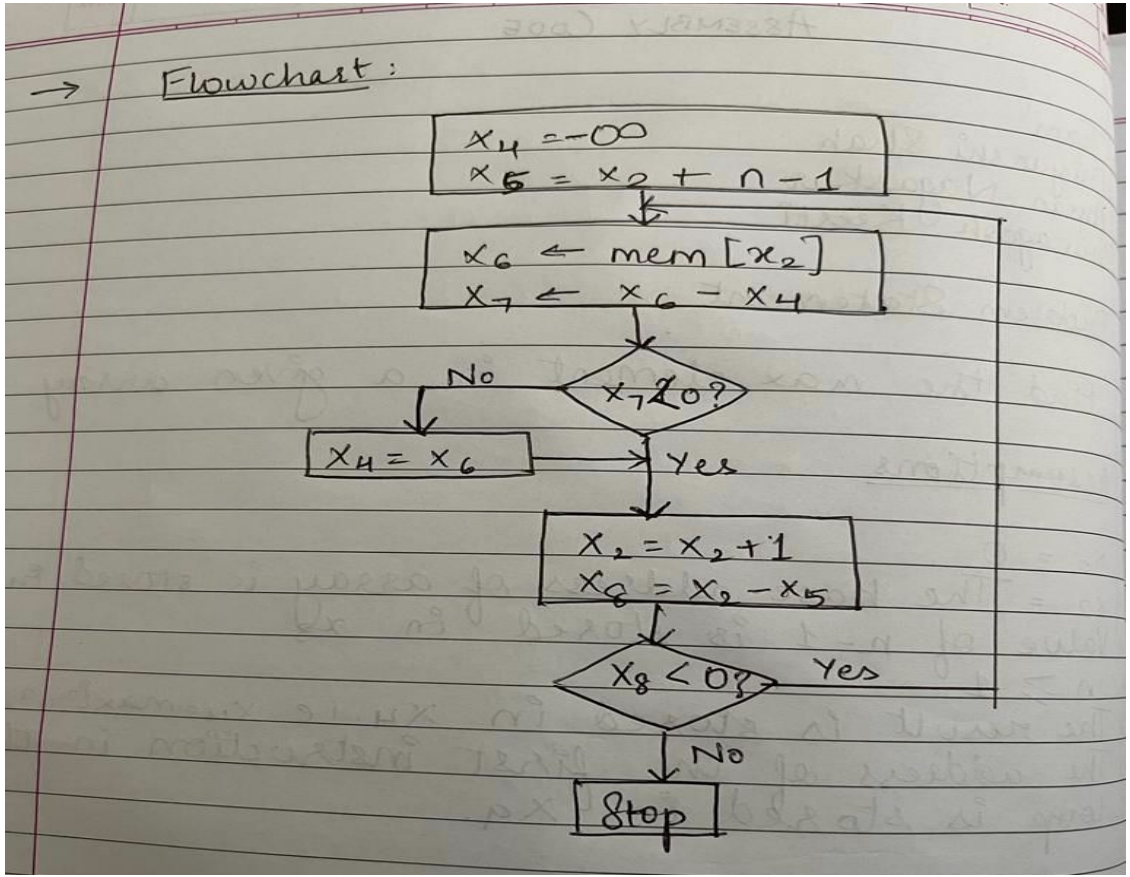
SCU Instruction set Architecture

Instruction	Symbol	Opcode	rd	rs	rt	Function
No operation	NOP	0000	×	×	×	No operation
Save PC	SVPC rd, y	1111	rd	y		xrd ← PC+y
Load	LD rd, rs	1110	rd	rs	×	xrd ← M[xrs]
Store	ST rt, rs	0011	×	rs	rt	M[xrs] ← xrt
Add	ADD rd, rs, rt	0100	rd	rs	rt	xrd ← x rs + x rt
Increment	INC rd, rs, y	0101	rd	rs	y	x rd ← x rs + y
Negate	NEG rd, rs	0110	rd	rs	×	x rd ← - x rs
Subtract	SUB rd, rs, rt	0111	rd	rs	rt	x rd ← x rs – x rt
Jump	J rs	1000	×	rs	×	PC ← x rs
Branch if zero	BRZ rs	1001	×	rs	×	PC ← x rs, if Z = 1
Jump memory	JM rs	1010	×	rs	×	PC ← M[x rs]
Branch if negative	BRN rs	1011	×	rs	×	PC ← x rs, if N = 1
MAX	MAX, rd, rs, rt	0001	rd	rs	rt	See *

Assumptions

1. $X_0 = 0$
2. X_2 = The base address of array is stored in this.
3. X3 = Value of n-1 is stored in this.
4. $X_4 = -2^{31}$
5. N >= 1
6. The result is stored in the X_4 i.e $X_4 = \max[a_1, a_2, \text{till } a_n]$
7. The address of the first instruction in the loop is stored in X_9

Flow Chart


Flowchart :
$x_4 = -\infty$
$x_5 = x_2 + n - 1$
$x_6 \leftarrow mem[x_2]$
$x_7 \leftarrow x_6 - x_4$
No
$x_7 \geq 0$?
$x_4 = x_6$
Yes
$x_2 = x_2 + 1$
$x_8 = x_2 - x_5$
$x_8 < 0$?
Yes
No
Stop

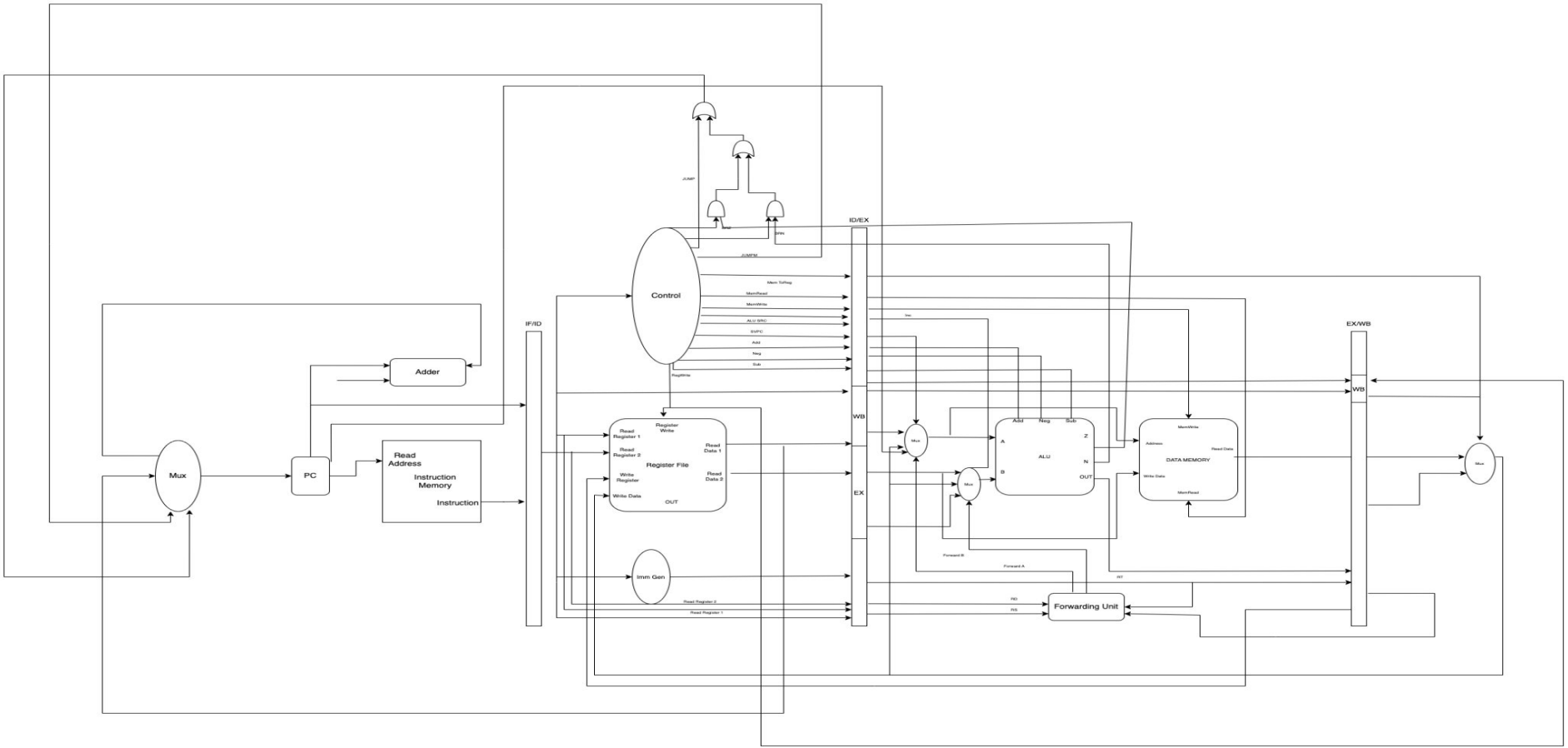
Software Version of Assembly code

```
ADD X5,X2,X3
LD X6,X2
SVPC X10,4
SUB X7,X6,X4
BRN X10
ADD X4,X6,X0
INC X2,X2, 1
SUB X8, X2, X5
BRN X9
BRZ X9
```

A = [8,5,10,7]

X4 will store result (curr max)
X9 will store address of 1st instruction

Datapath and Control



Truth Table

Sr. No.	Control Signal / Instruction	RegWrite	MemtoReg	MemRead	MemWrite	BRZ	BRN	Jump	JumpM	Add	Sub	Neg	ALU Src1	SVPC
1	SavePC	1	0	0	0	0	0	0	0	1	0	0	1	1
2	Load	1	1	1	0	0	0	0	0	0	0	0	0	0
3	Store	0	x	0	1	0	0	0	0	0	0	0	0	0
4	Add	1	0	0	0	0	0	0	0	1	0	0	0	0
5	Increment	1	0	0	0	0	0	0	0	1	0	0	1	0
6	Negate	1	0	0	0	0	0	0	0	0	0	1	0	0
7	Subtract	1	0	0	0	0	0	0	0	0	1	0	0	0
8	Jump	0	x	0	0	0	0	1	0	0	0	0	0	0
9	Branch if Zero	0	x	0	0	1	0	0	0	0	0	0	0	0
10	Jump Memory	0	x	0	0	0	0	0	2	0	0	0	0	0
11	Branch if Negative	0	x	0	0	0	1	0	0	0	0	0	0	0

Performance Analysis

- The given delay is:

Instruction Memory	2ns
Data Memory	2ns
ALU	2ns
Adders	2ns
Reg File	1.5ns

- Cycle Time = 2ns
- Clock Rate = 1/Cycle Time = 1/2 ns = $0.5 * 10^9$

Depending on how many comparisons need to be made, the initial instruction in the loop only runs once, while the subsequent instructions run n times.

Therefore, total number of instructions = 1 + 7n

Performance Analysis

Since the datapath is pipelined, each instruction is fetched, decoded, and further processed at each level. As a result, each stage's delay time is running parallelly.

Now, except for the branch instruction, which requires two cycles, all instructions run in a single cycle. The pipeline is in four phases. As a result, the first instruction requires four cycles, whereas the subsequent instructions each need one cycle. Additionally, since a branch instruction requires two cycles, the following equation requires n additional cycles:

- Suppose our cycle time is ct and total elements in the list is n,

  So, No. of cycles = 4 + (7n) + n
  = 4 + 8n cycles

- Execution time = cycle time x no. of cycles = 2*(4+8n)= 8 +16n ns
- Hence, CPI = No of cycles/no of instructions = (4+8n)/(1+7n)

Thank You