Kathmandu University

Dhulikhel, Kavre

Department of Computer Science and Engineering



A Mini Project Report

On

"AIS Computer"

[COMP-315]

(For the partial fulfillment of 3^{rd} Year/ 1^{st} Semester in Computer Engineering)

Submitted by

Group 1

Salina Koirala (27)

Ishar Maharjan (29)

Aron Shrestha (42)

Submitted to

Pankaj Dawadi

Department of Computer Science and Engineering

Submission Date: July 8, 2021

Acknowledgment

Our mini-project would not have come this far without the invaluable contribution of the Department of Computer Science and Engineering, who gave us the platform to explore new ideas and how to proceed further on our mini-project. We would like to express our heartfelt gratitude to our teacher Mr. Pankaj Dawadi for the opportunity.

Again, we would thank all those individuals who have helped us give a tiny bit of assistance in our project.

Sincerely,

Salina Koirala (27)

Ishar Maharjan (29)

Aron Shrestha (42)

Abstract

"AIS computer" is a theoretical model of a basic computer of size 32k*22. It has a 2-bit addressing mode, 5-bit opcode, and 15-bit address. It has 22-bit registers connected to a 22-bit common bus system and uses 15-bit address lines to refer to the memory location of the system. Similarly, our system has got 9 Memory Reference Instructions, 13 Register Reference Instructions, and 6 Input/Output Instructions.

Table of Contents

Acknowledgment	i
Abstract	ii
Table of Contents	iii
List of Tables	iv
List of figures	v
Chapter 1: Introduction	1
Chapter 2: Design Considerations	2
2.1 Instruction Format	2
2.2 Registers	2
2.3 Flip Flops	4
2.4 Common Bus System:	5
2.5 Types of Instructions	6
Addressing Mode Decoder	6
2.6 Instruction cycle	11
2.7 Interrupt cycle	12
2.8 Control signals	12
Chapter 3: Individual Designs	16
3.1 Registers	16
3.2 Flip Flop	21
3.3 Memory	25
3.4 ALU	26
Chapter 4: Conclusion	29

List of Tables

1.	Registers used in AIS Computer	3
2.	Addressing Mode Decoder	7
3.	Memory reference instruction set	10
4.	Register reference instruction set	11
5.	I/O instruction set	28

List of figures

Figure 2.1: 22-bit memory word

Figure 2.2: Control Unit of AIS Computer

Figure 2.3: Direct Memory Reference Instruction Format

Figure 2.4: Indirect Memory Reference Instruction Format

Figure 2.5: Register Reference Instruction Format

Figure 2.6: Input/Output Instruction Format

Figure: Instruction cycle flowchart

Figure 3.1.1: Address Register

Figure 3.1.2: Data Register

Figure 3.1..3: Accumulator

Figure 3.1.4: Program Counter

Figure 3.1.5: Temporary Register

Figure 3.1.6: Instruction Register

Figure 3.1.7: Output Register

Figure 3.2.1: E flipflop

Figure 3.2.2: R flipflop

Figure 3.2.3: S flipflop

Figure 3.2.4: Sequence counter

Figure 3.2.5: IEN flipflop

Figure 3.2.6 FGI flip flop

Figure 3.2.7 FGO flip flop

Figure 3.3.1: Memory write

Figure 3.3.1: Memory read

Figure 3.4.1 :ALU

Figure 3.4.2 : Control for Common Bus System

Chapter 1: Introduction

The Basic computer generally has two components, a processor and a memory. A processor is responsible for performing different operations of the system. Based on the different sets of instructions it can perform, different types of registers used, size of memory, timing and control system, interrupts, etc., different organizations of basic computers are determined.

AIS Computer, $32K \times 22$, has a very simple architecture compared to today's complex computers. Our computer has a 2-bit addressing mode, 5-bit opcode, and 15-bit address. It supports basic arithmetic and logic operations such as addition, subtraction, AND, OR, etc. An accumulator is used to store the output of an ALU operation. The data register is used to store data from memory before performing any operations on them. Address register and stack pointer are both used to specify memory address with the help of a common address bus. Data is transferred between registers using a common address bus. Program counter stores the next instruction to be executed, and the computer interfaces with the user using input and output registers. All the instructions are executed using the signals generated by the control unit consisting of a sequence counter and an extensive collection of logic gates.

The hardware components of AIS Computer are:

- A memory unit of 32K (32768) words with 22 bits each.
- Nine registers: AR, PC, DR, AC, IR, TR, OUTR, INPR, and SC.
- Eight flip-flops: I₁, I₀, S, E, R, IEN, FGI, and FGO.
- A 22 -bits common bus.
- Control logic gates.
- Decoders

Chapter 2: Design Considerations

2.1 Instruction Format

The instruction format of the AIS computer is as shown in the figure below:

- 2 bits addressing mode
- 5 bits opcode
- 15 bits memory address

21 19 14 0

A.M.	Opcode	Address
2 bit	5 bit	15 bit

Figure 2.1: 22-bit memory word

2.2 Registers

Registers	Bit Size	Bits		
PC	15	0-14		
AR	15	0-14		
IR	22	0-21		
TR	22	0-21		
DR	22	0-21		
AC	22	0-21		
INPR	8	0-7		
OUTR	8	0-7		
SC				

Table 2.1: Registers used in AIS Computer

A brief description of registers:

- Program Counter (PC): It contains the address of the memory location where
 the next instruction is located. During each execution of an instruction, the
 value of PC is incremented by one pointing at the next instruction that needs
 to be executed. AIS PC is a 15-bit register.
- Address Register(AC): It holds the address of the operand in the memory.
 This address lets the specific operand come out from memory and go to the memory. It is 15 bits as only 15 bits are required to specify a particular address in a memory having 2¹⁵ registers.
- Instruction Register(IR): It stores the 22-bit instruction code read from memory. Both the input and output of the instruction register come from and go to the central bus system.
- Temporary Register (TR): It stores temporary data during the working of the computer. It is of 22 bits.
- Data Register(DR): It is a 22-bit register used to store the operand read from the memory to be processed. The output of the data register is interfaced with the ALU. Any value that is to be operated must at first be transferred to the Data register.
- Accumulator (AC): Accumulator is the main register of this computer. It is 22-bit and stores the result of any operation after its completion. The input of the accumulator comes from the output of the ALU. The output of the AC goes to the common bus system as well as to ALU as the first operand, the second being the Data register.

- Input Register (INPR): It is an 8-bit register that holds the input character feed on an input device (e.g., a keyboard). Its output is connected to ALU as the value entered by the user needs to be processed with the accumulator.
- Output Register (OUTR): It is 8-bit and holds the output character that needs to be displayed to the user. Its input is connected to the central bus system.

2.3 Flip Flops

Flip-flop is a circuit that maintains a state until directed by input to change the state. The AIS computer has seven flip-flops. The flip flops with their uses are as follows:

- I_0 , I_1 : These flip-flops determine the addressing mode.
- R: This flip flop determines whether you are in the instruction cycle or interrupt the cycle.
- S: It determines the start-stop state of the program.
- E: This flag determines the carry.
- IEN; This flag indicates that an interrupt has occurred.
- FGI: Indicates interrupt due to input device.
- FGO: Indicates interrupt due to output device.

2.4 Common Bus System:

It is a 22-bit common bus system that connects all the registers to the memory. Three selection lines control the input of the bus. The following figure is a block diagram of the common bus system of the computer.

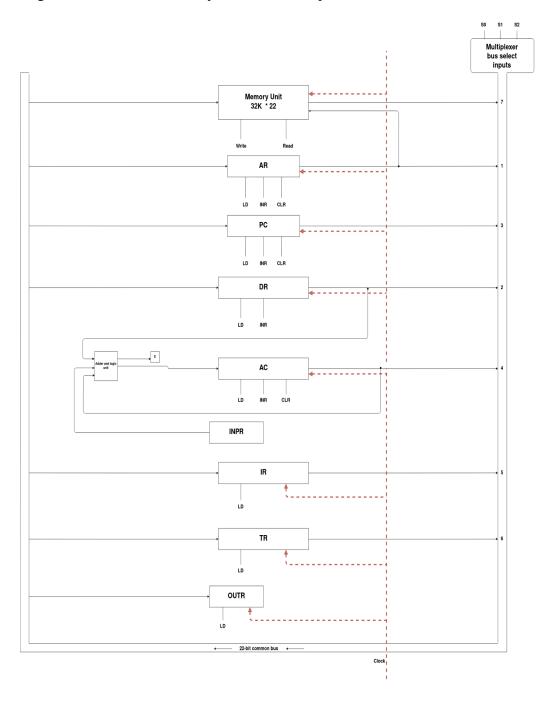


Figure 2.2: Control Unit of AIS Computer

2.5 Types of Instructions

The instruction code is divided into three parts, an opcode (Operation Code) that specifies the operation for that instruction and an address that specifies the registers and/or locations in memory to use for that operation. Also, it has two bits for determining addressing mode.

Addressing Mode Decoder

I_1I_0	Output (Instruction Type)
00	Input/Output Instruction
01	Direct Memory Reference Instruction
10	Indirect Memory Reference Instruction
11	Register Reference Instruction

Table 2.2: Addressing Mode Decoder

These instruction codes are a set of binary numbers in the memory word, which makes the processor do a certain task. Based on the op-code, the addressing mode flag (I), three categories of instructions are MRI, RRI, and IOI.

1. Memory Reference Instruction(MRI):

When the opcode (15-19 bits) is from 00000 to 01000, the instruction is recognised as an MRI. The 0-14 bits are used for the address of an operand. The 20-21 bits I_0I_1 determine the addressing mode of the instruction, which can be direct or indirect when I_0I_1 is 01 or 10 respectively.

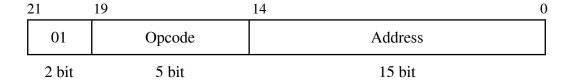


Figure 2.3: Direct Memory Reference Instruction Format

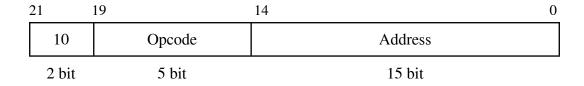


Figure 2.4: Indirect Memory Reference Instruction Format

2. Register Reference Instructions(RRI):

Register reference instructions are recognized by the control when $I(I_0I_1) = 11$. The instructions use the 5 bits opcode from 01000 to 10101 of the instruction register i.e. IR(15-19) to specify the type. The rest of the bits are neglected in this case and can be used to increase the number of instructions if needed.

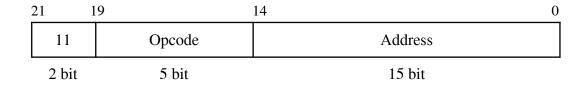


Figure 2.5: Register Reference Instruction Format

3. Input Output Instruction(IOI):

If the bits I_0I_1 (20-21) is 00, then the instruction is input-output instruction. The type of instruction is specified by the 5 bits opcode from 10110 to 11011. Here, 0-14 bits are neglected.

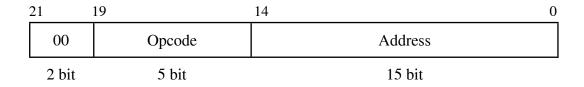


Figure 2.6: Input/Output Instruction Format

Here, x is the hexadecimal equivalent of the binary bits.

Memory reference instructions

Hex Code		Crombol	Decemention	Opcode bits	
Direct	Indirect	Symbol	Description	DILS	
100xxx	200xxx	AND	AND memory word to AC	00000	
107xxx	207xxx				
108xxx	208xxx	OR	OR memory word to AC	00001	
10Fxxx	20Fxxx				
110xxx - 117xxx	210xxx - 217xxx	ADD	Add memory word to AC	00010	
118xxx - 11Fxxx	218xxx - 21Fxxx	LDA	Load memory word to AC	00011	
120xxx - 127xxx	220xxx - 227xxx	STA	Store content of AC in memory	00100	
128xxx - 12Fxxx	228xxx - 22Fxxx	BUN	Branch unconditionally	00101	
130xxx - 137xxx	230xxx - 237xxx	BSA	Branch and save the return address	00110	
138xxx - 13Fxxx	238xxx - 23Fxxx	ISZ	Increment and skip if zero	00111	
140xxx	240xxx	XOR	XOR memory word to AC	01000	
147xxx	247xxx				

Table 2.3: Memory reference instruction set

Register reference instructions

Hex Code	Opcod e	Description	Opcode bits
348xxx - 34Fxxx	CLA	Clear AC	01001
350xxx - 357xxx	CLE	Clear E	01010
358xxx - 35Fxxx	CMA	Complement AC	01011
360xxx - 367xxx	CME	Complement E	01100
368xxx - 36Fxxx	CIR	Circulate right AC and E	01101
370xxx - 377xxx	CIL	Circulate right AC and E	01110
377xxx - 37Fxxx	INC	Increment AC	01111
380xxx - 387xxx	DEC	Decrement AC	10000
388xxx - 38Fxxx	SPA	Skip next instruction if AC is positive	10001
390xxx - 397xxx	SNA	Skip next instruction if AC is negative	10010
398xxx - 39Fxxx	SZA	Skip next instruction if AC is zero	10011
3A0xxx- 3A7xxx	SZE	Skip next instruction if E is zero	10100
3A8xxx- 3AFxxx	HLT	Halt Operation	10101

Table 2.4: Register reference instructions

Input/output instructions

Hex Code	Opco de	Description	Opcode bits
0B0xxx - 0B7xxx	INP	Input character to AC	10110
0B8xxx - 0BFxxx	OUT	Output character from AC	10111
0C0xxx - 0C7xxx	SKI	Skip on the input flag	11000
0C8xxx - 0CFxxx	SKO	Skip on output flag	11001
0D0xxx - 0D7xxx	ION	Interrupt on	11010
0D8xxx - 0DFxxx	IOF	Interrupt off	11011

Table 2.5: I/O instruction set

2.6 Instruction cycle

The instructions in the memory need to be executed. Until a stop instruction is encountered, the instructions are executed serially. Every instruction goes through different processes (cycles) to be executed, which is called the instruction cycle as a whole. It is mainly divided into three parts: Fetch, Decode and Execute.

1. Fetch: In this cycle, the content of the PC is transferred to AR. then, the memory content in the address AR is loaded into the instruction register (IR). Also, the PC is incremented by 1 to point to the next instruction for later. RTLs for this cycle is given below:

$$R'T_0$$
: $AR \leftarrow PC$
 $R'T_1$: $IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$

2. Decode: During this cycle, the 20-21 bits of IR is loaded into I₁I₀ flip flops. The bits from 15-19 are decoded and the type of instruction is determined which can be MRI, RRI or I/O instruction. Remaining 0-14 bits are loaded into AR. The RTLs for this cycle is given below:

$$R'T_2: I_1I_0 \leftarrow IR(20-21),$$

$$A_0...A_8, B_0....B_7, C_1....C_6 \leftarrow IR(15-19), AR \leftarrow IR(0-14)$$

$$I_0'I_1T3: AR \leftarrow M[AR]$$

3. Execute: This cycle is where the actual execution of instruction occurs accordingly. The complete set of RTl and control functions for different types of instructions are listed later in the report.

2.7 Interrupt cycle

During the execution cycle of each instruction, the computer keeps checking for an interrupt to occur. If any output or input devices are triggered, it is said that an interrupt has occurred and the IEN flag is set to 1, then a special cycle known as interrupt cycle is executed. When any input interrupt occurs, the FGI flag is set to 1 and when any output interrupt occurs, the FGO flag is set to 1. During the interrupt cycle, the return address is saved in a specific memory address. Then, the program branched to the pre-contained input/output program in the memory and after that, it returns back to the program where it was before interruption. Until the interrupt cycle is executed fully, no further interrupts are allowed to occur. RTLs for this cycle is given below:

```
T0'T1'T2'(IEN)(FGO+FGI): R\leftarrow1
RT0 : AR\leftarrow0, TR\leftarrowPC
RT1 :M[AR]\leftarrowTR, PC\leftarrow0
RT2 :PC\leftarrowPC+1, IEN\leftarrow0, R\leftarrow0, SC\leftarrow0
```

2.8 Control signals

The complete set control signals of all the instructions are given below:

Fetch:

$$\begin{split} R'T_0: AR &\leftarrow PC \\ R'T_1: IR &\leftarrow M[AR], PC \leftarrow PC + 1 \\ \textbf{Decode}: \\ R'T2: I_0I_1 &\leftarrow IR(20\text{-}21), A_0...A_8, B_0....B_7, C_1....C_6 \leftarrow IR(15\text{-}19), \\ AR &\leftarrow IR(0\text{-}14) \\ I_0'I_1T3: AR &\leftarrow M[AR] \end{split}$$

Interrupt:

```
T0'T1'T2'(IEN)(FGO+FGI): R\leftarrow1
RT0 : AR\leftarrow0, TR\leftarrowPC
RT1 :M[AR]\leftarrowTR, PC\leftarrow0
RT2 :PC\leftarrowPC+1, IEN\leftarrow0, R\leftarrow0, SC\leftarrow0
```

Execute:

Memory reference instructions

AND:

A0 T4: DR
$$\leftarrow$$
 M[AR]

A0 T5: AC
$$\leftarrow$$
 AC \wedge DR, SC \leftarrow 0

OR:

A1 T4: DR
$$\leftarrow$$
 M[AR]

A1T5:
$$AC \leftarrow AC \lor DR, SC \leftarrow 0$$

ADD:

A2 T4: DR
$$\leftarrow$$
 M[AR]

A2 T5: AC
$$\leftarrow$$
 AC + DR, E \leftarrow Cout, SC \leftarrow 0

LDA:

A3T4: DR
$$\leftarrow$$
 M[AR]

A3 T5: AC
$$\leftarrow$$
 DR, SC \leftarrow 0

STA:

A4 T4: M[AR]
$$\leftarrow$$
 AC, SC \leftarrow 0

BUN: Branch Unconditionally

A5 T4: PC
$$\leftarrow$$
 AR, SC \leftarrow 0

BSA: Branch anA Save Return Address

A6 T4:
$$M[AR] \leftarrow PC$$
, $AR \leftarrow AR + 1$

A6 T5: PC
$$\leftarrow$$
 AR, SC \leftarrow 0

ISZ: Increment an A Skip-if-Zero

A7 T4: DR
$$\leftarrow$$
 M[AR]

A7 T5: DR
$$\leftarrow$$
 DR + 1

A7 T6:
$$M[AR] \leftarrow DR$$
, if $(DR = 0)$ then $(PC \leftarrow PC + 1)$,

$$SC \leftarrow 0$$

XOR: Exclusive OR

A8 T4: DR
$$\leftarrow$$
 M[AR]

A8 T5: AC
$$\leftarrow$$
 AC \oplus AR, SC \leftarrow 0

Register reference instructions

$$R = I_0 I_1 T_3$$

B_i (specifies the type of RRI)

$$R: SC \leftarrow 0$$

CLA:

$$rB_0$$
: AC $\leftarrow 0$

$$rB_1: E \leftarrow 0$$

$$rB_2$$
: $AC \leftarrow AC'$

CME:

$$rB_3$$
: $E \leftarrow E'$

CIR:

$$rB_4$$
: AC \leftarrow shr AC, AC(21) \leftarrow E, E \leftarrow AC(0)

CIL:

$$rB_5$$
: AC \leftarrow shl AC, AC(0) \leftarrow E, E \leftarrow AC(21)

INC:

$$rB_6$$
: AC \leftarrow AC + 1

DEC:

$$rB_7$$
: AC \leftarrow AC - 1

SPA:

$$rB_8$$
: If $(AC(21) = 0)$ then $(PC \leftarrow PC + 1)$

SNA:

$$rB_9$$
: If $(AC(21) = 1)$ then $(PC \leftarrow PC + 1)$

SZA:

$$rB_A$$
: If (AC = 0) then (Pc \leftarrow Pc + 1)

SZE:

$$rB_B$$
: If (E = 0) then (PC \leftarrow PC + 1)

HLT:

$$rB_C: S \leftarrow 0$$

Input output instructions

$$P = I_1'I_0'T_3$$

C_i (specifies the type of I/O I)

INP:

$$pC_1$$
: $AC(0-7) \leftarrow INPR$, $FGI \leftarrow 0$

OUT:

$$pC_2$$
: OUTR \leftarrow AC(0-7), FGO \leftarrow 0

SKI:

$$pC_3$$
: if (FGI = 1) then (PC \leftarrow PC + 1)

SKO:

$$pC_4$$
: if (FGO = 1) then (PC \leftarrow PC + 1)

ION:

$$pC_5$$
: IEN $\leftarrow 0$

IOF:

$$pC_6$$
: IEN $\leftarrow 0$

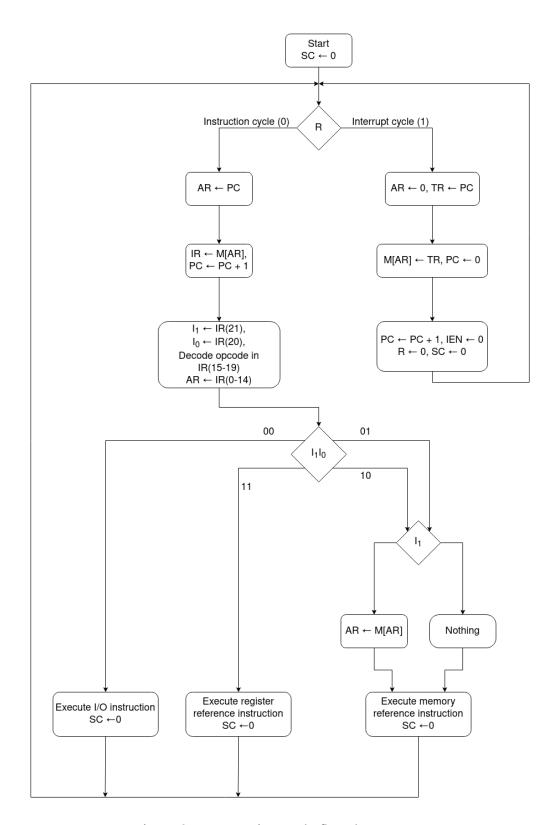


Figure 2.7: Instruction cycle flowchart

Chapter 3: Individual Designs

3.1 Registers

1)Address Register

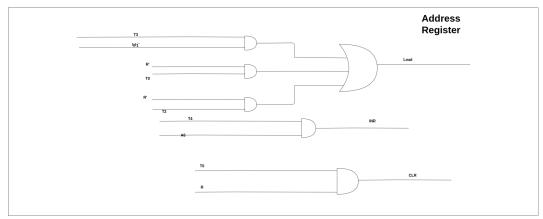


Fig 3.1.1: Address Register

```
\begin{split} &CLR = RT_0\\ &INCR = A_6T_4\\ &LOAD = R'T_0 + R'T_2 + T_3.I_0.I_1' \end{split}
```

2)Data Register

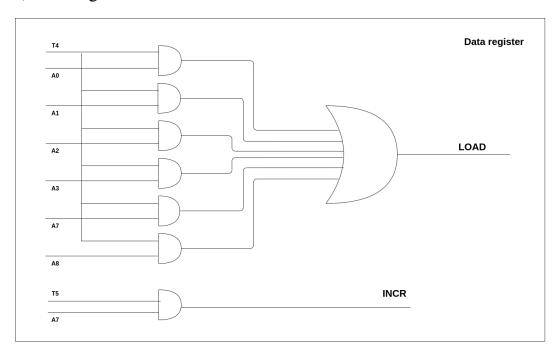


Fig 3.1.2: Data Register

$$\begin{aligned} &LOAD = (A_0 + A_1 + A_2 + A_3 + A_7 + A_8).T_4 \\ &INCR = T_5.A_7 \end{aligned}$$

3)Accumulator

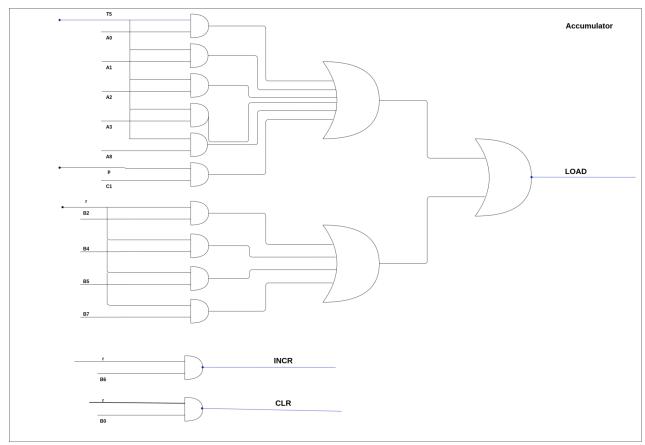


Fig 3.1..3: Accumulator

LOAD: $(A_0 + A_1 + A_2 + A_3 + A_8).T_5 + rB_2 + rB_4 + rB_5 + rB_7 + pC_1$

INCR: rB₆

 $CLR : rB_0$

4)Program Counter

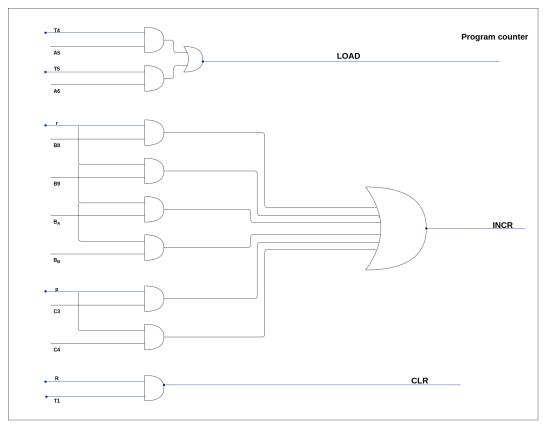


Fig 3.1.4: Program Counter

Load: A_5T , + A_6T_5

Increment = $rB_8 + rB_9$

5)Temporary Register

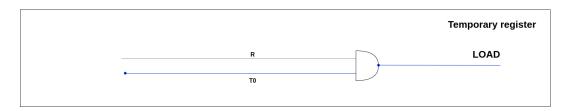


Fig 3.1.5: Temporary Register

 $LOAD = R.T_0$

6) Instruction Register

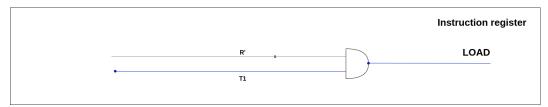


Fig 3.1.6: Instruction Register

 $LOAD = R'.T_1$

7)Output Register



Fig 3.1.7: Output Register

 $LOAD = T_3.C_1$

3.2 Flip Flop

1)E

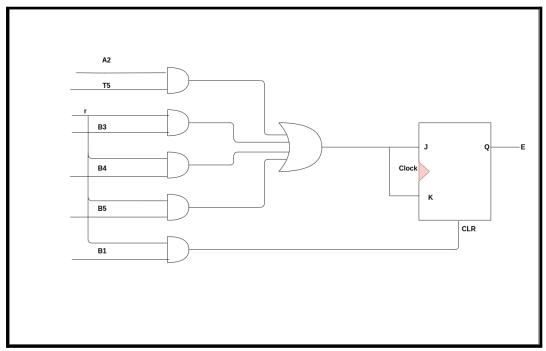


Fig 3.2.1: E flipflop

LOAD: rB3 + rB4 + rB5 +A2T5 CLEAR :rB1

2)R

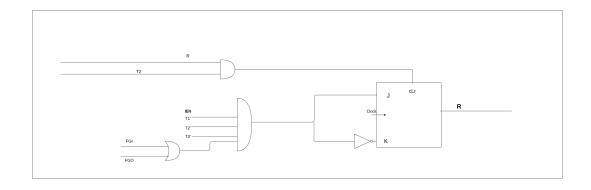


Fig 3.2.2: R flip flop

 $\begin{array}{l} SET: T_0'T_1'T_2'.IEN.(FGO+FGI) \\ RESET: RT_2 \end{array}$

3)S

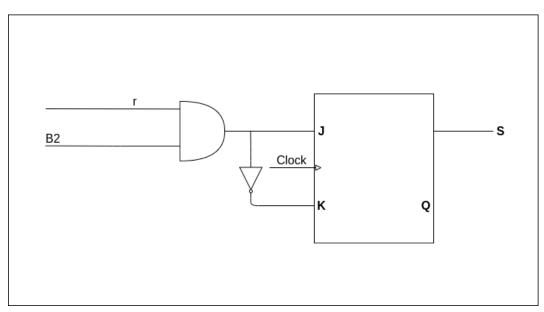


Fig 3.2.3: S flip flop

SET: rB₂ RESET: SET'

4)SC

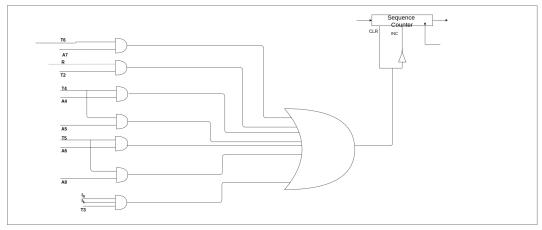


Fig 3.2.4: Sequence counter

CLR: $T_6A_7 + RT_2 + T_4A_4 + T_4A_5 + T_5A_6 + T_5A_8 + I_0I_1T_3$ INC: CLR'

5)IEN

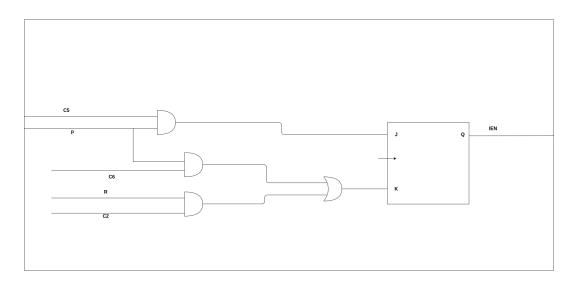


Fig 3.2.5: IEN flip flop

SET: pC_5 RESET: $pC_6 + RC_2$

6)FGI

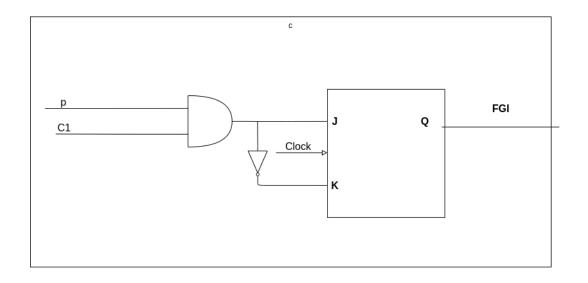


Fig 3.2.6 FGI flip flop

SET: pC₁

7)FGO

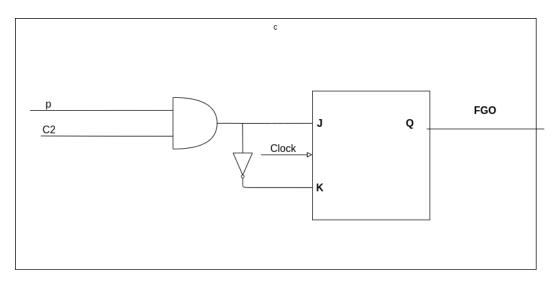


Fig 3.2.7 FGO flip flop

SET: pC₂

3.3 Memory

1)Memory write

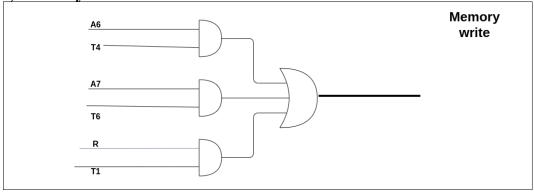


Fig 3.3.1: Memory write

Write: $A_6T_4 + A_7T_6 + RT_1$

2) Memory read

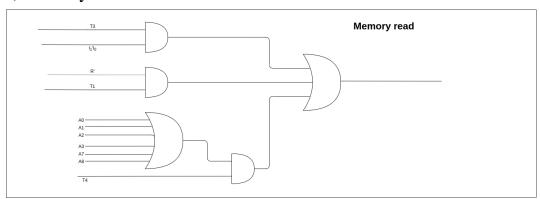


Fig 3.3.1: Memory read

Read: $T_3I_1'I_0 + R'T_1 + (A_0 + A_1 + A_2 + A_3 + A_7 + A_8).T_4$

3.4 ALU

As the name suggests, this unit performs all the arithmetic and logical calculations in the computer. Its output is directly connected to the input of the accumulator and therefore is of 15 bits in the computer design. Its inputs are connected to the outputs of the Data Register, Accumulator and the Input Register. The solution obtained from the calculation in the ALU is sent to the Accumulator, so it is from here that the output goes to the bus to its destination (i.e, Memory or the OUTR). A detailed design of a bit of the ALU is given below.

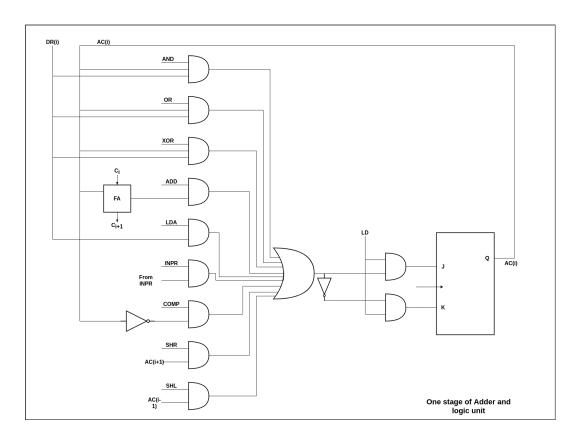


Fig 3.4.1:ALU

Control of common bus

Inputs					Outputs			Register		
X1	X2	Х3	X4	X5	X6	X7	S2	S1	S0	selected
0	0	0	0	0	0	0	0	0	0	None
1	0	0	0	0	0	0	0	0	1	AR
0	1	0	0	0	0	0	0	1	0	DR
0	0	1	0	0	0	0	0	1	1	PC
0	0	0	1	0	0	0	1	0	0	AC
0	0	0	0	1	0	0	1	0	1	IR
0	0	0	0	0	1	0	1	1	0	TR
0	0	0	0	0	0	1	1	1	1	Memory

Table 3.1: Decoder for control of common bus

$$S0 = X1 + X3 + X5 + X7$$

 $S1 = X2 + X3 + X6 + X7$
 $S2 = X4 + X5 + X6 + X7$

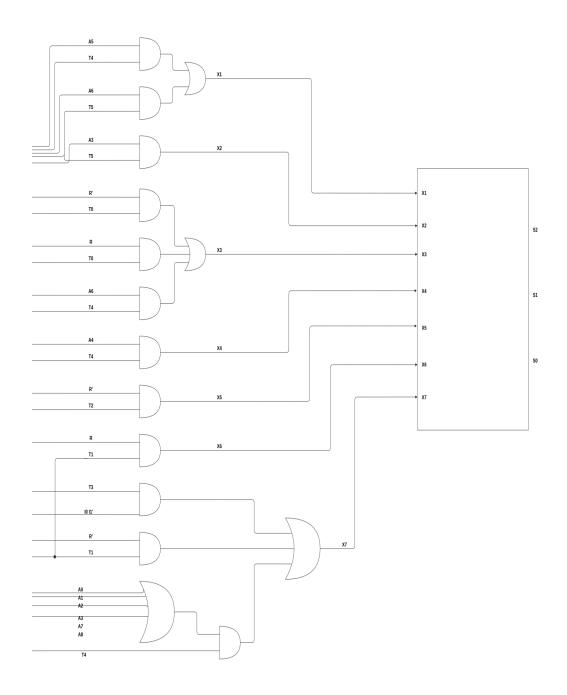


Fig 3.4.2:Control for Common Bus System

Chapter 4: Conclusion

We learned the architecture of basic computers from this project. By the end of the project we got the insight of how registers, flip flops are used in basic computer and various machine level instruction. Although the computer we design is very minimal and simple it's not for commercial use. However we learned the basics of how computers are designed which is the basis for all computers.