AAAA COMPUTER

GHULAM ISHAQ KHAN INSTITUTE OF ENGINEERING SCIENCES AND TECHNOLOGY

FACULTY OF COMPUTER SCIENCES AND ENGINEERING

CS 222 - ComputerOrganization and Assembly Language

Section: B

Submitted By

Muhammad Adil Shahid (2018224)
Muhammad Asim Khan (2018246)
Syed Amir Hussain (2018446)
Adnan Haider (2018035)

Table of Contents

Abstract

Literally the word computer means “a device that can compute (calculate)”. In today’s world the electronic computer has become an essential part of our life.

This report is about the design of a basic computer AAAA. It is the modified version of Basic computer. It includes all instruction set and architecture of computer. The major changes which are made in AAAA that distinguishes it from Morris Mano’s Basic computer are the length of the instruction which is 18 bits as compared to 16 bits. In the instruction of AAAA computer, the last two bits are used for the addressing mode. These last two bits decode into either one of the four addressing modes which are direct, indirect, indexed and immediate whereas Morris Mano’s Basic computer has two of them direct and indirect only.

There is also an addition of a register in the common bus i.e. Indexed Register (IX) and a flip-flop named Status. Indexed Register is utilized for Indexed addressing mode and Status Flip-flop is used to store the result of comparator.

Another significant change is of the introduction of Compare micro-operation in Memory Reference Instructions which compares a number with accumulator and stores either 1 or 0 bit in Status flip-flop.
Furthermore, there is Jump if True, Jump if Not True, multiply, swapping operand with AC, conversion from binary to gray code and gray code to binary number instructions introduced in Memory reference instructions as well.
Last but not the least modification is the option of Decrement in Register Reference Instructions which allow the contents of the register to be decreased by 1.

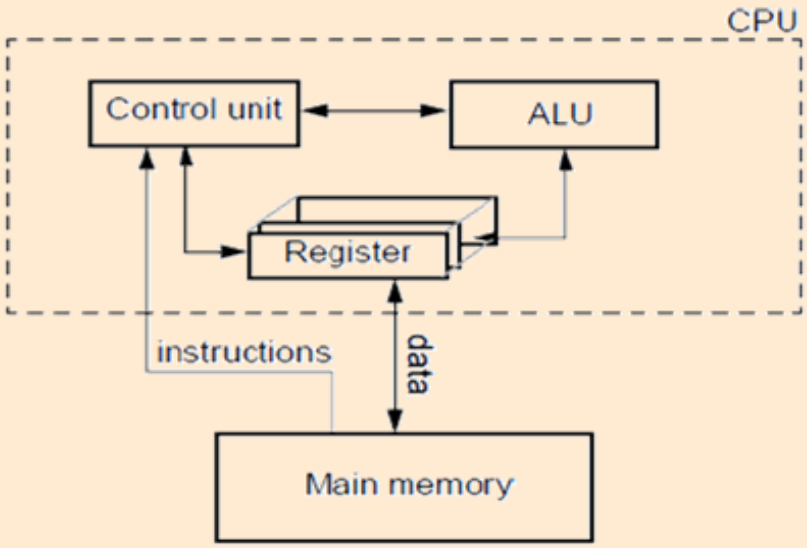
In a nutshell, this design has many new options and function which were absent in the previous designs and all of these changes combined makes the AAAA much more powerful and efficient in terms of memory and power consumption and allows more complex assembly programs to be executed easily.

Introduction

The literary meaning of the word computer is “a device that can calculate”. Modern computers can do more than calculations. Now a day’s computer is a programmable machine designed to perform logical and arithmetic operations on the input given by the user, and give desired output after processing.

The architecture of computer is basically of two types, Von Neumann and Harvard architecture. Modern computers follow Von Neumann.

The Neumann Architecture basically defines the simplest construction of an electronic computer which has four major parts, Arithmetic Logic Unit (ALU), Control Unit, Memory to store instructions and data both and Output and Input procedures. On a huge scale the capacity to regard instructions as data is the idea that makes assemblers and compilers possible. One can “Compose programs which compose programs”. This has permitted modern self-facilitating eco-system to prosper around Von Neumann design machines.



General-purpose (von Neumann) Architecture

The Morris Mano’s basic computer uses the same concept of the Neumann Architecture. The Neumann architecture has common memory to save data and programs. Our design follows the Von Neumann architecture.

Instruction Set Architecture

Instruction set which is also known as Instruction Set Architecture ISA, specify the order of arrangement by which processing has to take place. The set of instructions provide everything the processor needs to perform a specific task. A microprocessor's ISA include the information required to cooperate with the microprocessor. It serves as a boundary between hardware and software. ISA must specify the instructions, the registers on which the instructions depend as well as the sizes of these registers.

Instruction Format of AAAA

Most of the time the computer instruction is dived into two segments

- ✓ An op-code also known as the Operation Code that decodes the operation for a certain instruction. The instruction will specify the data they will process, in the form of operands. E.g. add, subtract, multiply, complement, etc.
- ✓ The Memory Address is the reference to a specific register and/or memory location to be utilized for that operation.

In AAAA's Computer, bit # 16 and 17 of the instruction identifies the addressing mode (00: Direct Address, 01: Indirect Address, 10: Indexed Address, 11: Immediate Address).Since the addressing mode utilizes 2 bits in instruction format and it needs 2 bits to address a word in a memory that leaves 4 bits for the instruction's op-code.

Addressing Modes of AAAA

Data and instructions in memory can be fetched by numerous ways. This process is also known as "Memory Addressing Modes". Memory address modes decide the strategy utilized inside the program to get to data either from inside the CPU or outside RAM. The memory addressing modes in the AAAA Computer are:

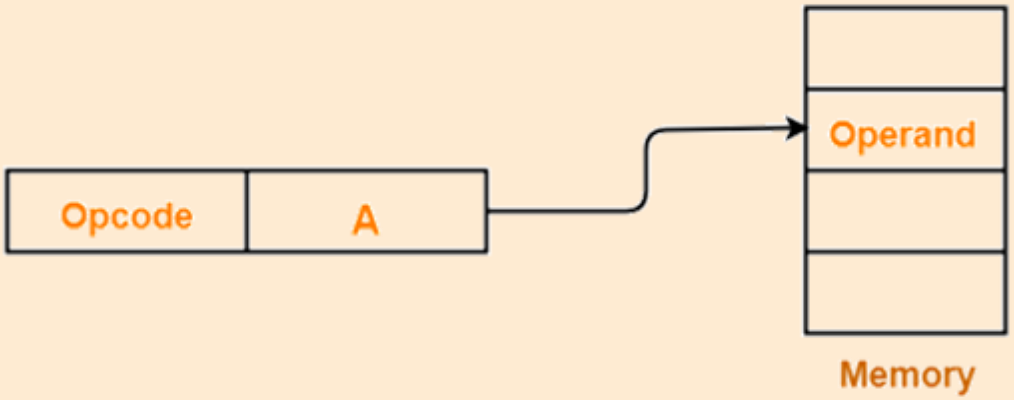
1. Direct
2. Indirect
3. Indexed
4. Immediate

Direct Addressing

This is an exceptionally straightforward method of memory addressing. Direct addressing simply points to the location in memory where the data is to be fetched from.

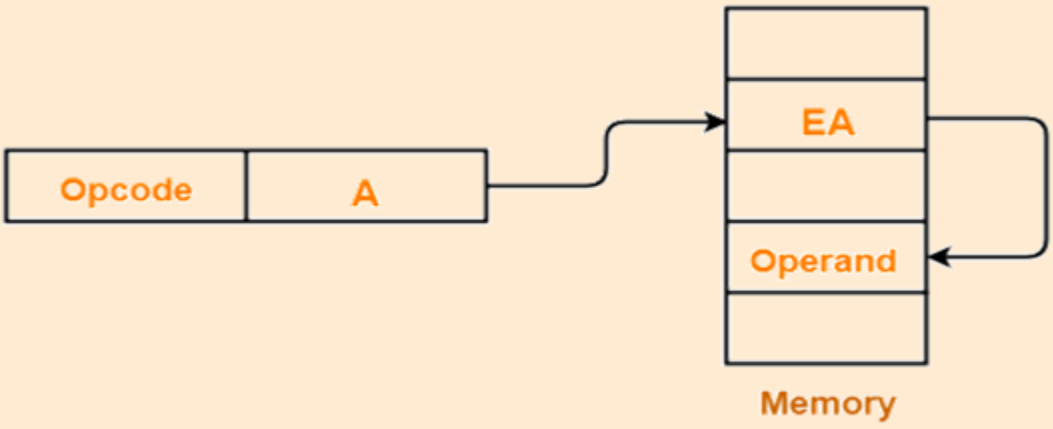
In the above stated example, data at memory location 457 gets loaded to the accumulator.

Fast is what makes the direct addressing preferable but on the other hand in direct addressing the code depends on the accurate data always being there at same location.



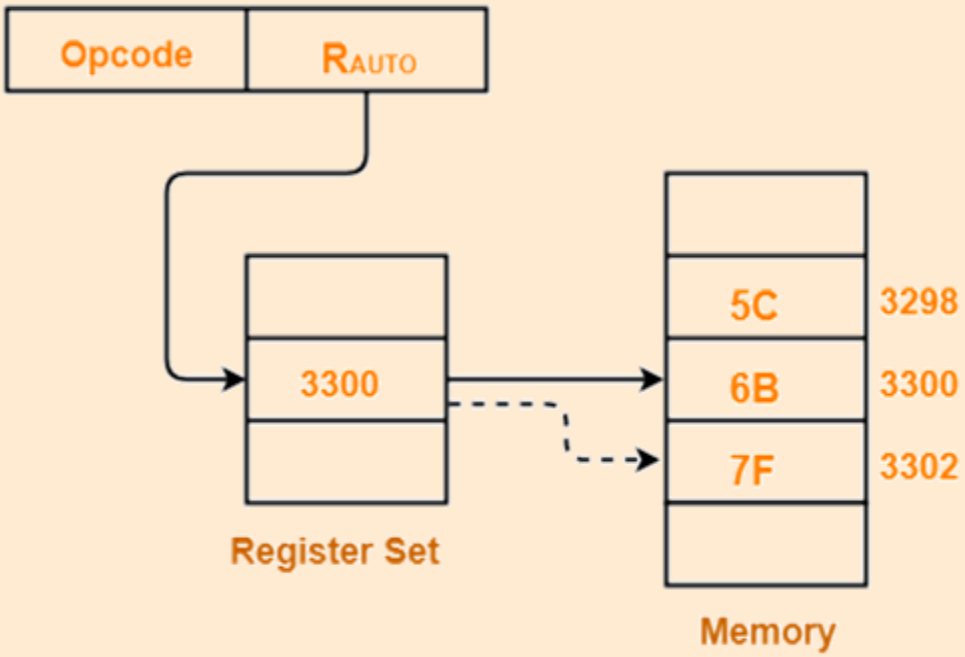
Indirect Addressing

In this addressing mode the address part does not contain the real address. In this mode the address part contain the address where the real address means the address of the operand is stored.



Auto Indexed (Increment mode)

In auto indexed mode of addressing, the address part of the instruction points to the register containing operand. After accessing the operand pointed by the address stored in the address part of the instruction the content of the register is incremented automatically to point to the next memory location. It is useful for traversing through arrays in a loop.



Immediate Addressing

Immediate tending implies that the data to be utilized is hard-coded into the instruction itself. This is the quickest technique of addressing as it doesn't include primary memory by any stretch of the imagination

Instruction Types

Three types of instructions are being used in this computer.

Memory Reference Instructions

In these instructions the memory of the computer is involved. They use the addressing part of the instruction which refers to the operand liable upon the kind of addressing mode used. They use the operand from memory.

Instruction Set For Memory Reference instructions					
Symbol	Hex Code				Description
	Direct	Indirect	Indexed	Immediate	
LDA	00xxx	10xxx	20xxx	30xxx	Load from memory to AC
STA	01xxx	11xxx	21xxx	31xxx	Store word from memory to AC
ADD	02xxx	12xxx	22xxx	32xxx	ADD memory word to AC
AND	03xxx	13xxx	23xxx	33xxx	AND memory word to AC
BSA	04xxx	14xxx	24xxx	34xxx	Branch and save return address
ISZ	05xxx	15xxx	25xxx	35xxx	Increment and skip if zero
JPN	06xxx	16xxx	26xxx	36xxx	Jump if not true
JPE	07xxx	17xxx	27xxx	37xxx	Jump if true
JMP	08xxx	18xxx	28xxx	38xxx	Jump
CMP	09xxx	19xxx	29xxx	39xxx	Compare with AC
SWP	0Axxx	1Axxx	2Axxx	3Axxx	Swap memory word with AC
MUL	0Bxxx	1Bxxx	2Bxxx	3Bxxx	Multiply memory word with AC
GTB	0Cxxx	1Cxxx	2Cxxx	3Cxxx	Convert Gray code to Binary
BTG	0Dxxx	1Dxxx	2Dxxx	3Dxxx	Convert Binary to Gray code

Register Reference Instructions

Register Reference instructions does not need memory that is why there is no requirement of the address for the operand. All the operations of the register reference instructions are completed in the registers. 00 is fixed as the address mode of the Register Reference Instructions

Instruction set for Register Reference Instructions		
Symbol	Hex Code	Description
CLA	0F001	Clear Accumulator
CLE	0F002	Clear Enable
CMA	0F003	Complement Accumulator
CME	0F004	Complement Enable
CIR	0F005	Circulate right Accumulator And Enable
CIL	0F006	Circulate left Accumulator And Enable
INC	0F007	Increment Accumulator
DECA	0F008	Decrement Accumulator
DECX	0F009	Decrement Any Register
SPA	0F00A	Avoid next instruction if Accumulator is positive
SNA	0F00B	Avoid next instruction if Accumulator is negative
SZA	0F00C	Avoid next instruction if Accumulator is zero
SZE	0F00D	Avoid next instruction if Enable is zero
IND	0F00E	Increment Data Register
HLT	0F00F	Close the Computer

Input/output Instructions

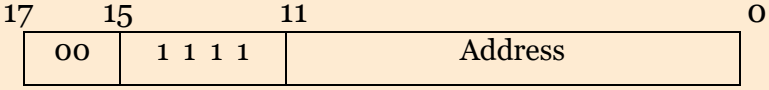
We interact using the Input/output instructions with the external devices of the microprocessor. The addressing part and the input / output operation varies for each instruction while the address mode is 11 and op-code is fixed to 1111.

Instruction Set For Input/output Instructions		
INP	1F800	Input character to Accumulator
OUT	1F400	Output character from Accumulator
SKI	1F200	Avoid on input flag
SKO	1F100	Avoid on output flag
ION	1F080	Interrupt ON
IOF	1F040	Interrupt OFF

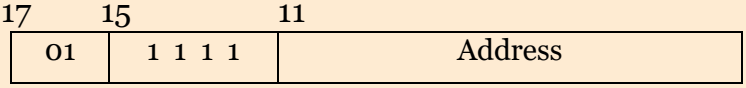
Instruction Set of AAAA

Memory Reference Instructions (Op-code = 0000 ~1001)

Register Reference Instructions (Op-code = 1111, Bit 16 & 17 = 00)



Register Reference Instructions (Op-code = 1111, Bit 16 & 17 = 01)



Instruction cycle

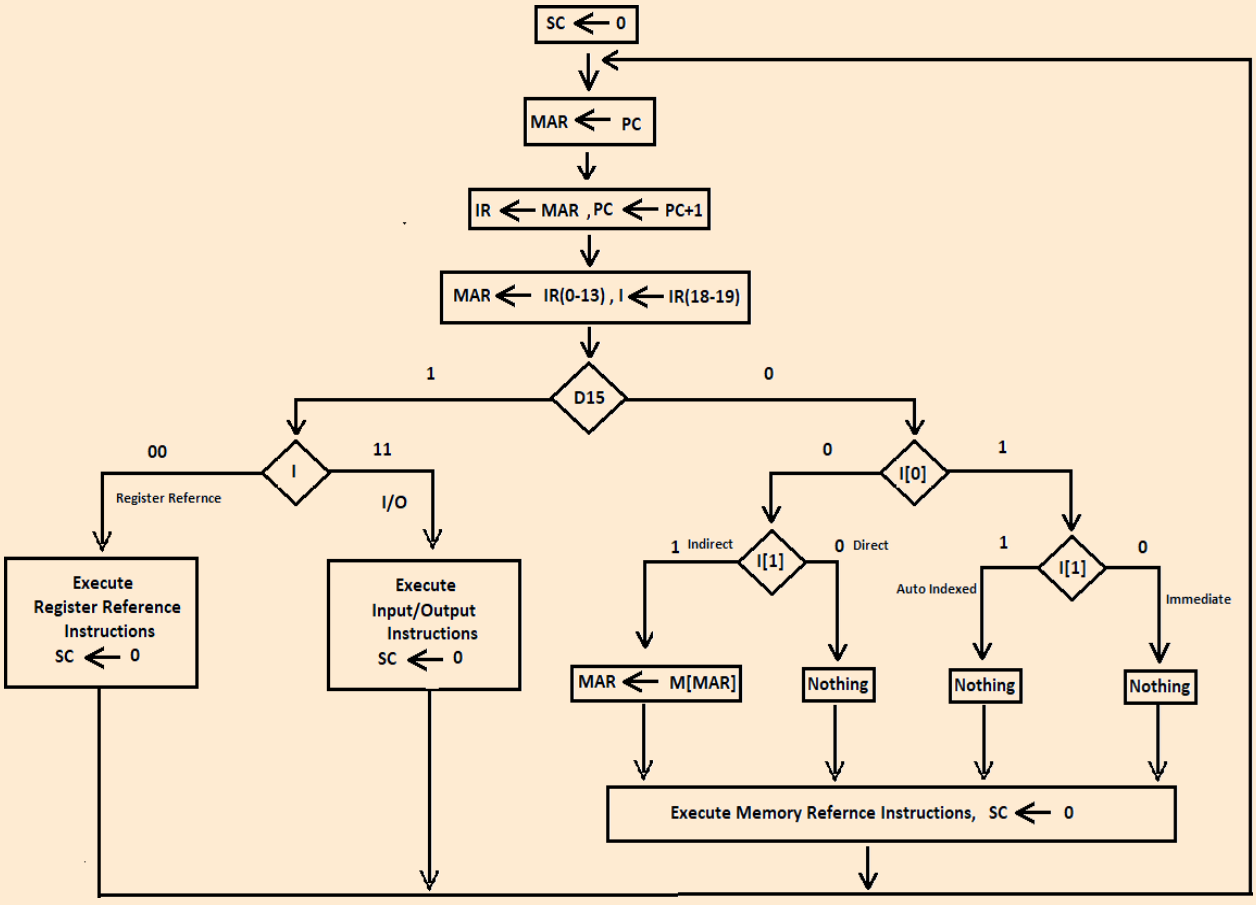
In the computer memory a program comprises of set of instructions.
The program is performed by experiencing a cycle for each instruction. The instruction cycle comprises of following stages:

- ✓ Fetch instruction from memory.
- ✓ Decrypt the instruction.
- ✓ Read the actual address from the memory if the instruction has an indirect address.
- ✓ Execute the instruction

Once the instruction is executed, the process or flow cycle begins back from the initial stage, for the next instruction. The cycle goes on until and unless the last instruction has been executed.
For every operation, there is a series of micro-operations expected to actualize that operation.

Micro-operations for AAAA Instructions

Control unit defines the type of instruction that is read from the memory. The following flow chart shows three types of instructions in basic computer.



Memory Reference Instructions

Symbol	Operation Decoder	Symbolic Description
LDA	D_0	AC ← Memory[MAR]
STA	D_1	MAR ← AC
ADD	D_2	AC ← AC + MAR, E ← Cout
AND	D_3	AC ← AC ^ MAR,
BSA	D_4	MAR ← PC, PC ← MAR + 1
ISZ	D_5	MAR ← MAR + 1, if MAR + 1 = 0 then PC ←PC + 1
JPN	D_6	If Status = 0 then PC ← MAR
JPE	D_7	If Status = 1 then PC ← MAR
JMP	D_8	PC ← MAR
CMP	D_9	Status ← AC # MAR
SWP	D_A	MAR← TR, SC ← 0
MUL	D_B	AC ← AC * DR, SC ← 0
GTB	D_C	DR ← MAR, AC ← DR*
BTG	D_D	DR ← MAR, AC ← DR*

Symbol	Binary			
	Direct	Indirect	Indexed	Immediate
LDA	00 0000 ×××× ×××× ××××	01 0000 ×××× ×××× ××××	10 0000 ×××× ×××× ××××	11 0000 ×××× ×××× ××××
STA	00 0001 ×××× ×××× ××××	01 0001 ×××× ×××× ××××	10 0001 ×××× ×××× ××××	11 0001 ×××× ×××× ××××
ADD	00 0010 ×××× ×××× ××××	01 0010 ×××× ×××× ××××	10 0010 ×××× ×××× ××××	11 0010 ×××× ×××× ××××
AND	00 0011 ×××× ×××× ××××	01 0011 ×××× ×××× ××××	10 0011 ×××× ×××× ××××	11 0011 ×××× ×××× ××××
BSA	00 0100 ×××× ×××× ××××	01 0100 ×××× ×××× ××××	10 0100 ×××× ×××× ××××	11 0100 ×××× ×××× ××××
ISZ	00 0101 ×××× ×××× ××××	01 0101 ×××× ×××× ××××	10 0101 ×××× ×××× ××××	11 0101 ×××× ×××× ××××
JPN	00 0110 ×××× ×××× ××××	01 0110 ×××× ×××× ××××	10 0110 ×××× ×××× ××××	11 0110 ×××× ×××× ××××
JPE	00 0111 ×××× ×××× ××××	01 0111 ×××× ×××× ××××	10 0111 ×××× ×××× ××××	11 0111 ×××× ×××× ××××
JMP	00 1000 ×××× ×××× ××××	01 1000 ×××× ×××× ××××	10 1000 ×××× ×××× ××××	11 1000 ×××× ×××× ××××
CMP	00 1001 ×××× ×××× ××××	01 1001 ×××× ×××× ××××	10 1001 ×××× ×××× ××××	11 1001 ×××× ×××× ××××
SWP	00 1010 ×××× ×××× ××××	01 1010 ×××× ×××× ××××	10 1010 ×××× ×××× ××××	11 1010 ×××× ×××× ××××
MUL	00 1011 ×××× ×××× ××××	01 1011 ×××× ×××× ××××	10 1011 ×××× ×××× ××××	11 1011 ×××× ×××× ××××
GTB	00 1100 ×××× ×××× ××××	01 1100 ×××× ×××× ××××	10 1100 ×××× ×××× ××××	11 1100 ×××× ×××× ××××
BTG	00 1101 ×××× ×××× ××××	01 1101 ×××× ×××× ××××	10 1101 ×××× ×××× ××××	11 1101 ×××× ×××× ××××

Register Reference Instructions

Symbol	Binary	Control Function	Micro-operations
CLA	00 1111 0000 0000 0001	rB_1	AC ← 0
CLE	00 1111 0000 0000 0010	rB_2	E ← 0
CMA	00 1111 0000 0000 0011	rB_3	AC ← AC'
CME	00 1111 0000 0000 0100	rB_4	E ← E'
CIR	00 1111 0000 0000 0101	rB_5	AC ← shr AC, AC(17) ← E, E ← AC(0)
CIL	00 1111 0000 0000 0110	rB_6	AC ← shl AC, AC(0) ← E, E ← AC(17)
INC	00 1111 0000 0000 0111	rB_7	AC ← AC + 1
DECA	00 1111 0000 0000 1000	rB_8	AC ← AC – 1
DECX	00 1111 0000 0000 1001	rB_9	IX ← IX – 1
SPA	00 1111 0000 0000 1010	rB_{10}	If (AC (17) = 0) then (PC ← PC + 1)
SNA	00 1111 0000 0000 1011	rB_{11}	If (AC (17) = 1) then (PC ← PC + 1)
SZA	00 1111 0000 0000 1100	rB_{12}	If (AC = 0) then (PC ← PC + 1)
SZE	00 1111 0000 0000 1101	rB_{13}	If (E = 0) then (PC ← PC + 1)
IND	00 1111 0000 0000 1110	rB_{14}	DR ← DR + 1
HLT	00 1111 0000 0000 1111	rB_{15}	S ← 0 (S = start-stop flip-flop)

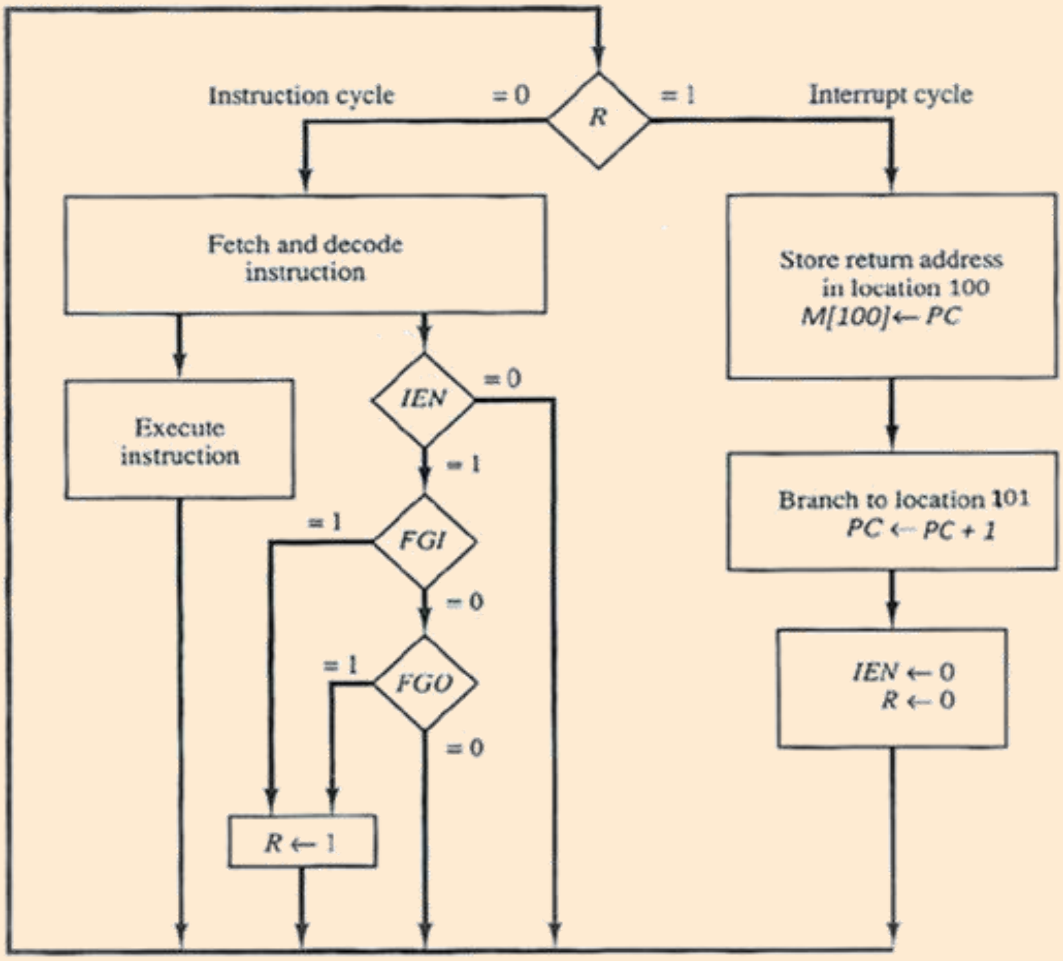
Input/output Instructions

Symbol	Binary	Control Function	Micro-operations
INP	00 1111 1000 0000 0000	pB_{11}	AC (0-7) ← INPR, FGI ← 0
OUT	00 1111 0100 0000 0000	pB_{10}	OUTR ← AC (0-7), FGO ← 0
SKI	00 1111 0010 0000 0000	pB_9	If (FGI = 1) then (PC ← PC + 1)
SKO	00 1111 0001 0100 0000	pB_8	If (FGO = 1) then (PC ← PC + 1)
ION	00 1111 0000 1000 0000	pB_7	IEN ← 1
IOF	00 1111 0000 0100 0000	pB_6	IEN ← 0

Interrupt Cycle

Interrupt is used to deal with input/output devices. Whenever input is available, the input flag is active and current instruction is interrupted. Computer will first perform the input task then instruction will be executed. Same goes for output flag. These interrupts are essential for computer to perform input/output tasks. IEN flip-flop is set to ON state when input/output interrupt is called that is, when FGI and FGO is set to 1.

Every time when instruction is performed computer check for input/output flag. If one these two flags are active IEN is set to 1, and interrupt subroutine is executed. Otherwise computer carry on with instruction.
Computer will store the address of the next instruction before the interrupt subroutine is encountered. After the execution of this subroutine computer will jump back to the return address and instruction cycle will be executed.



Hardware

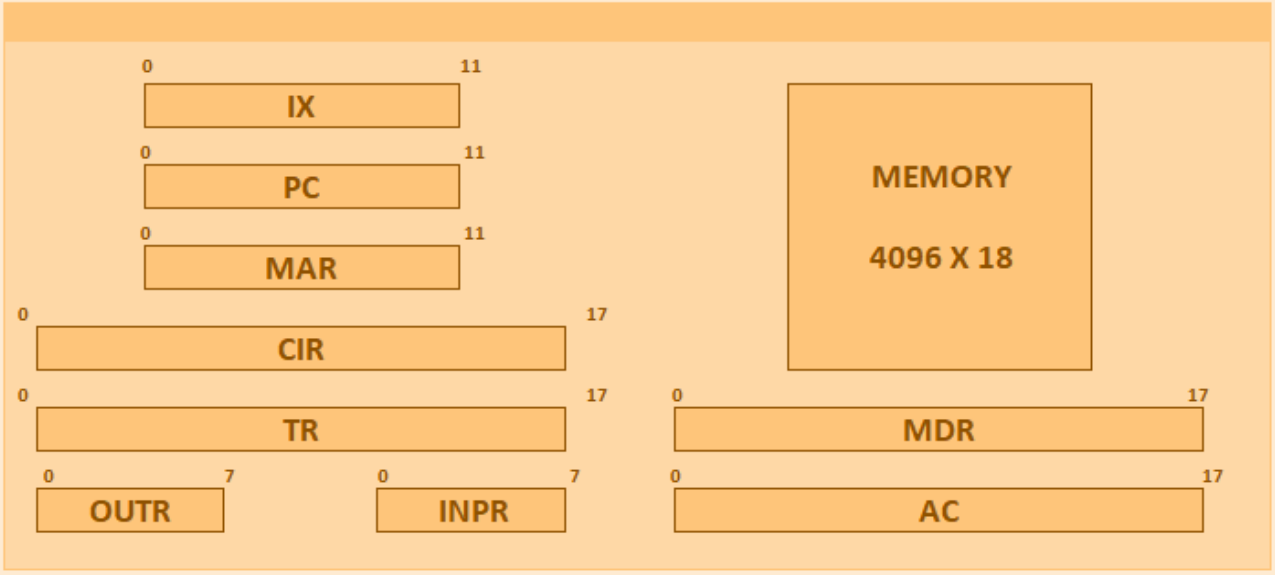
The hardware of the AAAA basic computer consists of registers, memory, circuits and buses. All of these are explained below

Memory

The memory is the main part of the computer hardware. The instructions or the operations which are to be performed by the computer are stored in the memory. The data or the operand which the user wants to store is also stored in the memory for future use. So, memory has the important role in the computer.
The memory of this computer has capacity of 2^12 words which it will contain 4,096 words. And each word stored in memory will consist of 18 bits.

Register

Register also have an important part they are used by computer for utilizing data and holding the memory address for the instruction being executed.



Registers in AAAA Computer		
IX	12	Indexed Register
MDR	18	Memory Data Register
MAR	12	Memory Address Register
AC	18	Accumulator
CIR	18	Current Instruction Register
PC	12	Program Counter
TR	18	Temporary Register
INPR	8	Input Register
OUTR	8	Output Register

Purpose of Registers in RBA Computer

Program Counter (PC):

Program Counter is used to store address of instructions which are stored in memory. The address of next instruction is stored in program counter to be executed. The size of the PC is 14-bits.

Current Instruction Register (CIR):

CIR holds the present instruction to be accomplished.

Memory Address Register (MAR):

MAR holds the address for memory.

Memory Data Register (MDR):

Holds data transferred from the memory through Bus ready for RBA to process.

Indexed Register (IX):

IX stores the offset which together with the base address allow the operand to be fetched. It is mostly used to perform array operations.

Temporary Register (TR):

TR holds Temporary data.

Accumulator (AC):

AC holds immediate arithmetic and logic results.

Input Register (INPR):

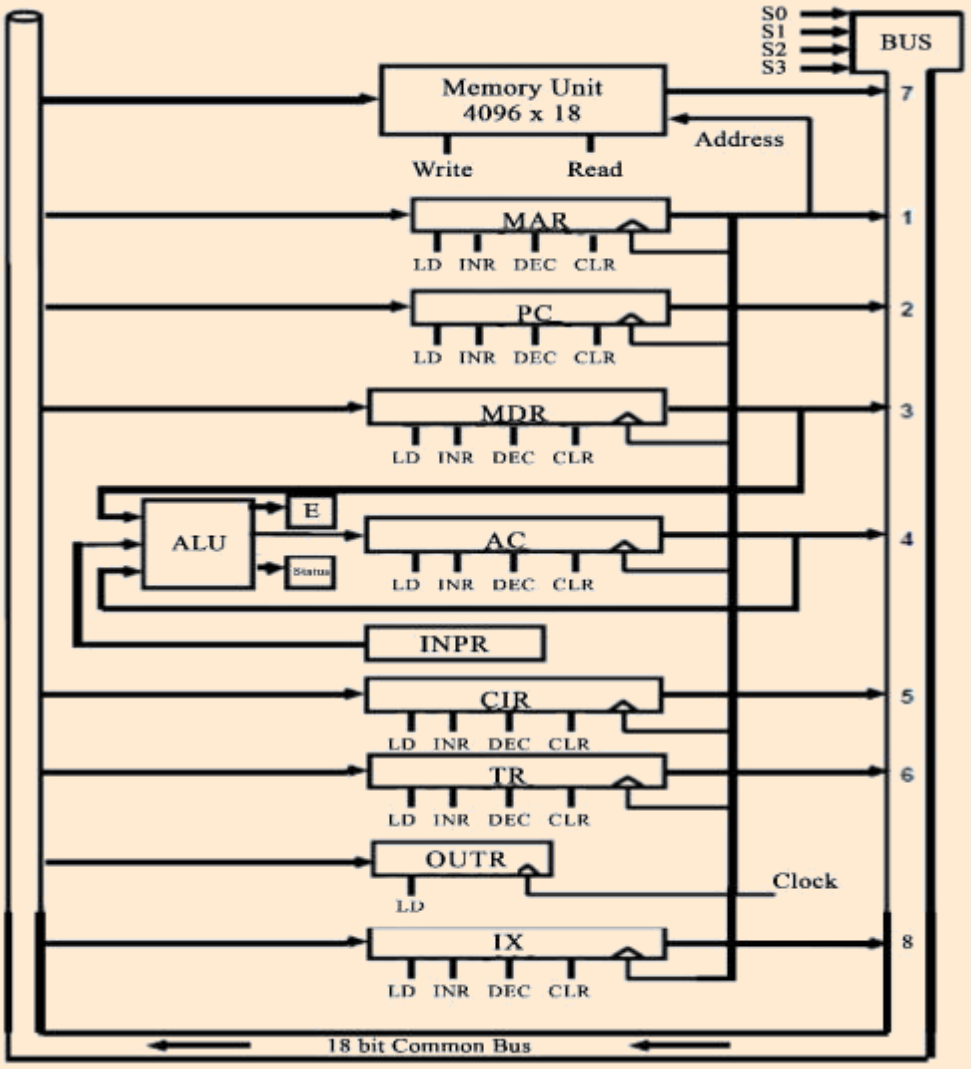
INPR holds the word that is taken as an input character.

Output Register (OUTR):

OUTR holds the word after processing the input instruction.

COMMON BUS SYSTEM

To transfer information between registers and between memory and registers, paths must be provided. There will be too many numbers of wires, if the direct links are made among the inputs of each register and the outputs of the other registers. A more suitable method for transferring information in a system with multiple registers is to use a common bus. The AAAA Computer consists of nine registers, a memory unit, and a control unit. AAAA's common bus houses the major block of a computer system. Data is transferred from register to register and the same goes on between the registers and the memory. Output of the registers and the memory are connected with each other through the AAAA's bus. The 19 lines of the common bus of AAAA get data from seven registers and the memory unit.



Control Unit

Control unit (CU) of a processor translates from machine instructions to the control signals for the micro-operations that implement them. The control unit of a computer is made up combinational & sequential circuits.

Control units can be set up in either in one of ways mentioned below:

Hardwired Control

In the Hardwired Control implementation, the control logic is implemented with flip flops, decoders, gates and other digital circuits.

Micro-programmed Control

A control memory on processor contains micro programs which activates the necessary control signals.

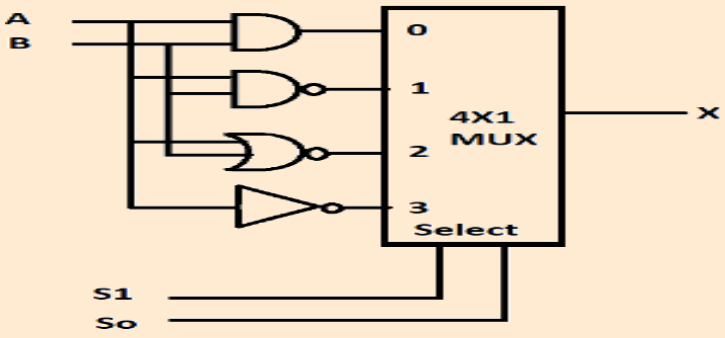
We have considered a hardwired implementation of the control unit for the AAAA Computer.

Arithmetic Logic Unit

The arithmetic logic unit (ALU) includes the circuits that are involved to perform the operation in the computer. The ALU includes the arithmetic and logical circuits. These arithmetic circuits perform the arithmetic operations and logical circuits perform logical operations. The op code of the instruction tells the ALU what operation to perform on which operands and which operands are to be used in the decoded operation. The ALU is responsible for all the mathematical calculations and the logical answers in the AAAA computer.

Logical circuit

Logical circuits are involved in performing logical operations such as AND, NOR, NAND, COMPLEMENT, etc.

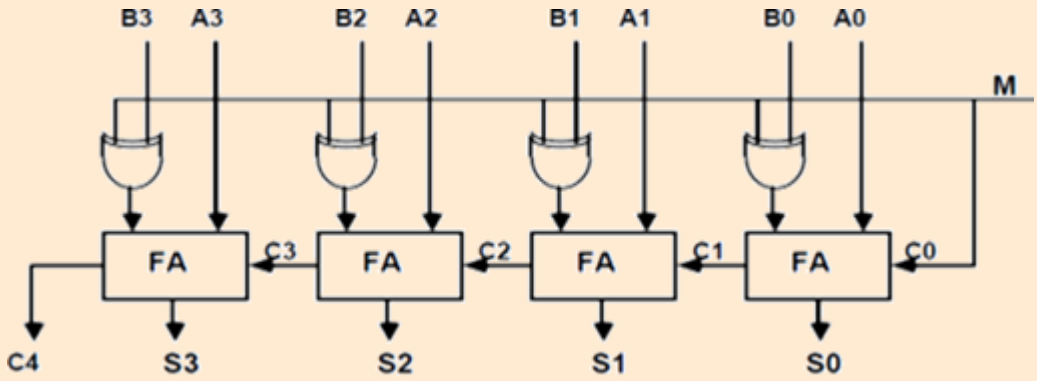


Arithmetic Circuits

The arithmetic circuits are involved in performing arithmetic operations such as ADD, SUBTRACT, MULTIPLICATION, circular shift etc.

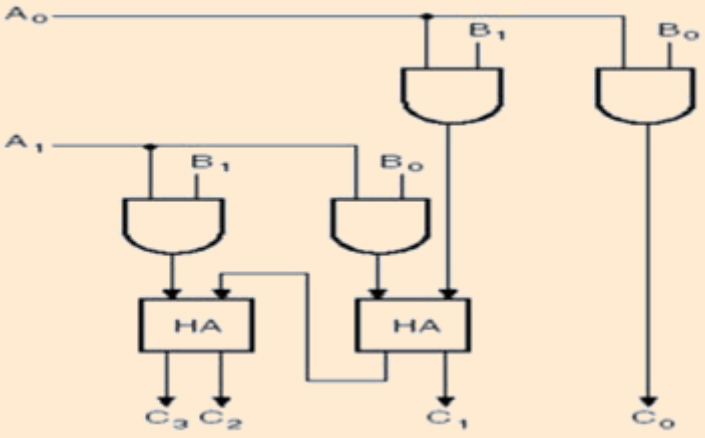
Adder-Subtractor

The circuit given below is for Adder and subtractor. This adds or subtracts only 4 bits. In this computer the 18 bit word is used so similarly this circuit will work for 18 bits word.



Multiplier

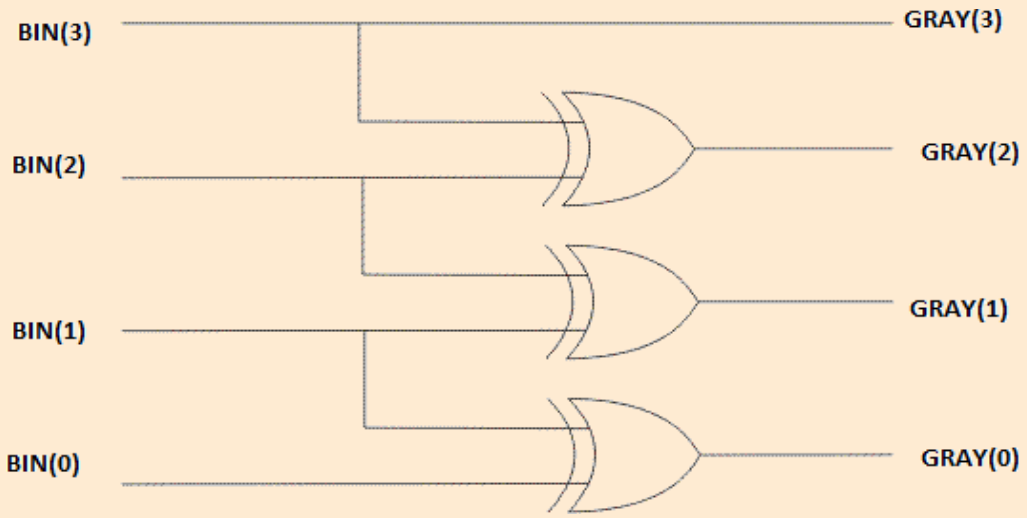
The circuit given below is for multiplying two 2-bit numbers. This circuit is designed for only 2 bits. In this computer the 18-bit word is used so similarly this circuit will work for 18 bits word.



Here A is the first number and B is the second number which is to be multiplied. And C is product of two numbers.

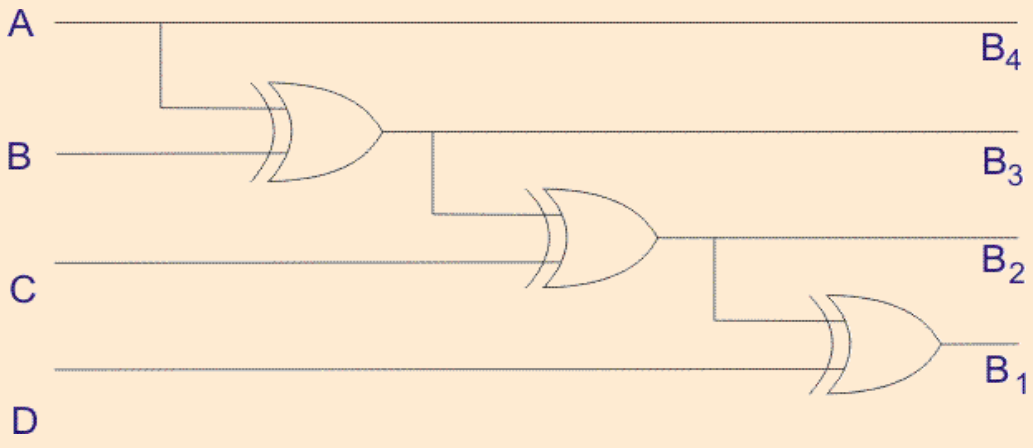
Binary to Gray Code Circuit

This circuit is designed for only 4 bits. 18 bits word is used in our case so than a somewhat similar circuit will work for 18 bits word.



Gray Code to Binary Circuit

This circuit is designed for only 4 bits. 18 bits word is used in our case so than a somewhat similar circuit will work for 18 bits word.



.

Conclusion

In the end we could conclude by saying that the AAAA Computer is without doubt more powerful than the Basic Computer as the increased and enhanced Instruction Set Architecture allows it to solve more complex programs in less time and much more efficiently. The hardware implementation of the AAAA computer can be seen more complex than the Basic but that complex hardware makes the AAAA available with more operations than the Basic Computer.

Bibliography

Computer System and Architecture, 3rd Edition, M. Morris Mano
Computer Organization & Architecture, 10th Edition by W. Stallings
https://www.geeksforgeeks.org/addressing-modes/
http://www.padakuu.com/article/517-common-bus-system