Department of Computer Science, Mathematics & Physics

COMP2220 Computer Systems Architecture

Laboratory #7 – IAS

Introduction

The focus of the laboratory session this week is on programming the IAS Computer. This asynchronous machine, also referred to as the von Neumann machine, was built from 1945 to 1951 and was (one of) the first electronic computers to make use of the stored program concept where both data and instructions resided in main memory. This was an advancement over the program-controlled computers such as the ENIAC.

Lab Objectives:

By the end of this session students will be able to:

- Trace a simple low-level language program.
- Create a level program using the ISA Simulator Tool.

Getting Started

A bit of history†:

- The IAS machine was a binary computer with a 40-bit word, storing two 20-bit instructions in each word.
- The memory was 1024 words (5.1 kilobytes).
- Negative numbers were represented in "two's complement" format.
- The IAS had two general-purpose registers available: the Accumulator (AC) and Multiplier/Quotient (MQ).
- Other registers include: Memory Buffer Register, Memory Address Register, Instruction Register, Instruction Buffer Register and Program Counter.
- The structure of the IAS is as illustrated in Figure 1. The instruction and data formats are provided in Figure 2.

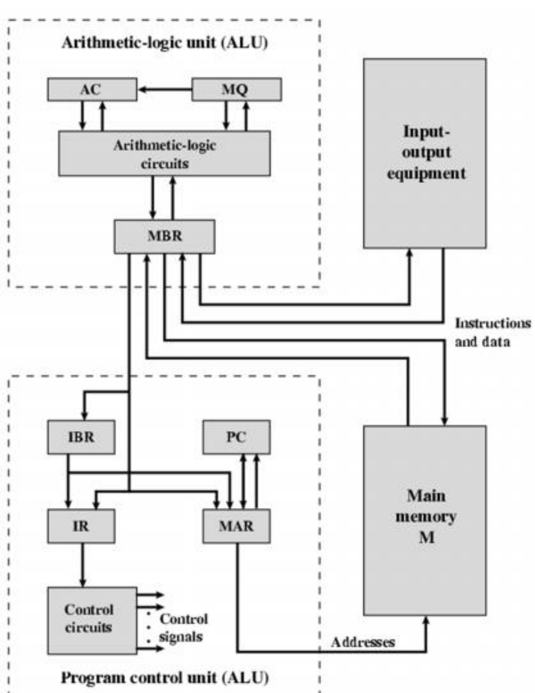

Figure 1 - ISA Structure

† W. Ware, "The history and development of the electronic computer project at the Institute for Advanced Study". March 1953

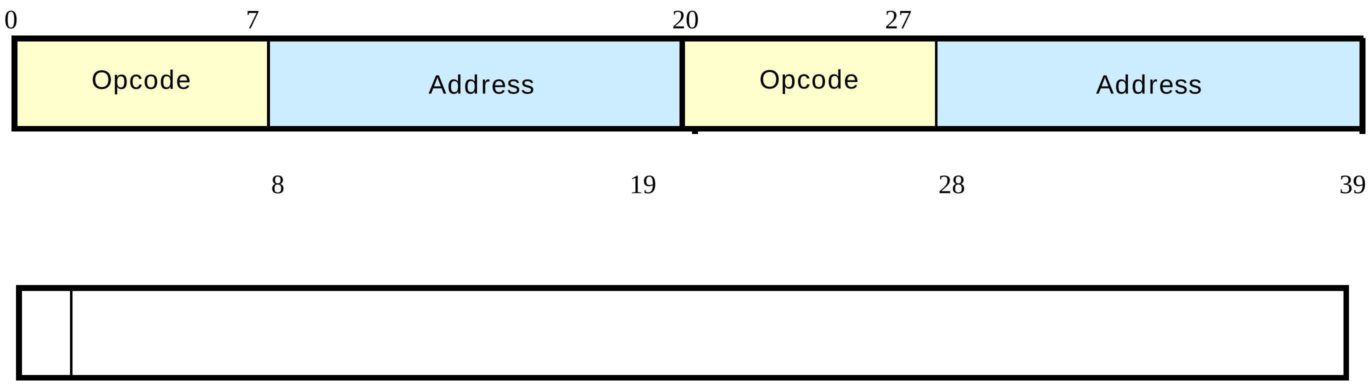

Figure 2 IAS Instruction Format (top) and Data Format (bottom)

With regards to the instruction format, observe that:

- Bits 0-19, and bits 20-39 represent the left hand side (LHS) and right hand side (RHS) instructions, respectively. Consequently, the instruction pair at memory address location 5 is referred to as 5L and 5R. Note that the IAS simulator tool refers to memory addresses as “Selectrons” and are denoted by S(x); where x is the address location. In the original instruction set, S(x) is referred to as M(x). Both instructions sets are provided at the end of this lab.

- Bits 0-7 and bits 20-27 represent the opcodes of the LHS and RHS instructions, respectively.

- Bits 8-19 and 28-39 represent the operands of the LHS and RHS instructions, respectively.

1. Describe the Purpose of the AC and MQ‡ registers.

[2 marks]

Task 1 – Understanding Sequence

A. Examine the instruction set provided on page 6 of the lab and note the descriptions of the instructions. A keen eye will recognize that the arrow “→” in the instruction name indicates the direction of the data flow. E.g. S(x)→Ac copies the number in Selectron location x into the AC register; conversely At→S(x) copied the number in AC to Selectron location x. Hence, these two commands function as load and store, respectively.

B. Based on the instruction set provided on page 6, examine the code in the table below and write appropriate comments describing each instruction. Note that the program is expressed in hexadecimal format.

Memory Address	Instruction	Comment
0L	01 004	
0R	05 005	
1L	11 006	
1R	00 000	
2L		
2R		
. . .	. . .	. . .
4	7	Data variable #1
5	8	Data variable #2
6		
. . .	. . .	. . .

Table 1 IAS Program

[2 Marks]

‡ Note that the MQ register is referred to as R in the IAS Simulator Tool.

2. What is the result stored in memory location 6H at the end of this program?

[1 mark]

Note that the program may be also expressed in the following equivalent ways (see page 7):

Memory Address	**IAS SIMTool Instruction**	**Original IAS Instruction**
0L	S(4) → Ac	LOAD M(4)
0R	S(5) → Ah+	ADD M(5)
1L	At → S(6)	STORE M(6)
1R	Halt	§

Task 2 – Understanding Selection

A. Based on the instruction set on page 6, examine the code in the table below and write appropriate comments for the describing the 1R and 2L. Note that the program is expressed in hexadecimal format.

Memory Address	Instruction	Comment
0L	01 004	
0R	05 005	
1L	11 006	
1R	10 002	
2L	11 005	
2R	00 000	
. . .	. . .	. . .
4	-7	Data variable #1
5	8	Data variable #2
6		
. . .	. . .	. . .

Table 2 IAS Program

[1 mark]

3. Write a suitable pseudocode snippet for the above program.

[2 mark]

Task 3 – Understanding Repetition

A. Examine the code snippet below and confirm that it calculates result = num^2.

```
// this code snippet is a ludicrous way to calculate the value of num*num
// in this example num is set to be 3.

num = 3;
result = 0;
count = num-1;

while (count >=0)
{
        result = result + num;
        count = count -1;
}
```

§ There was no halt instruction; programs ended by creating an infinite loop back to a known address.

B. The table below gives the code for this example. Trace the code and confirm its accuracy. Note that the final result is stored in memory location 9.

Memory Address	Instruction	Comment
0L	01 007	AC 🡐 **num**
0R	06 008	AC 🡐 AC - 1
1L	11 006	**count** 🡐AC
1R	10 002	if AC > 0 go to instruction 2R
2L	00 000	Halt
2R	01 009	AC 🡐 **result**
3L	05 007	AC 🡐 **result** + **num**
3R	11 009	**result** 🡐 AC
4L	01 006	AC 🡐 **count**
4R	06 008	AC 🡐 AC - 1
5L	11 006	**count** 🡐 AC
5R	0E 001	Jump to 1R
6	-	**count** data variable
7	3	**num** data variable
8	1	Data variable to represent ‘1’
9	-	**result** data variable

C. Launch the IAS SIM Tool. And configure as follows:

  A. Click Edit 🡒 Preferences 🡒 Selectron tab. Set the Selectron size to 1024; click Apply; click Close.
  B. In the **RAM Selectrons window** ensure that the Address type and Data type are set to hexadecimal; Cell size is set to 20.
  C. In the **Registers window** ensure that the Base is set to Hexadecimal.

D. Enter the program manually by clicking on the data fields and typing the hexadecimal values. Note that the memory addresses are in pairs. The first and second numbers represent the left address and right addresses, respectively. Figure 3 shows a screen shot of the program as entered into the simulator.

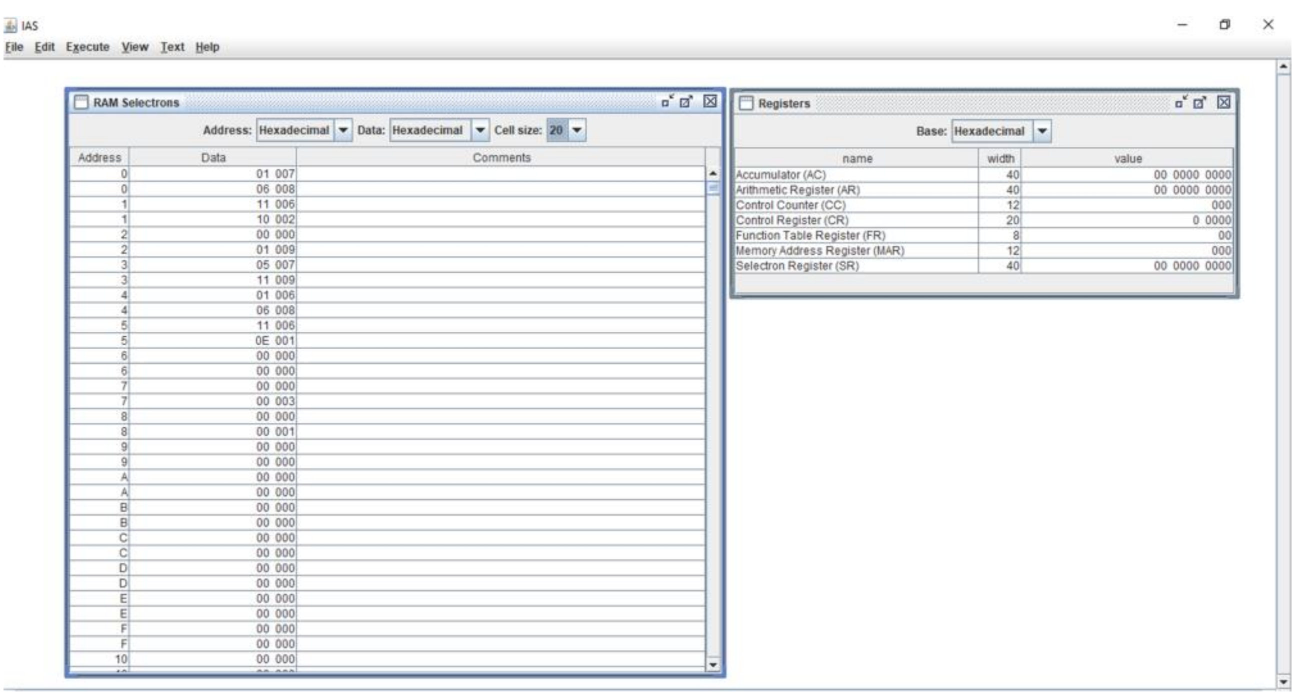

Figure 3 IAS SIM Screenshot

E. In the **RAM Selectrons window** ensure that the change the Cell size to 40. Note that this is a more efficient way to view the memory and is aligned with the instruction format presented in Figure 2.

F. Click on Execute → Run.

4. What is the result in memory location 9H?

[2 marks]

The program is repeated here with Original IAS instruction format included.

Memory Address	Instruction	Original IAS instruction	Comments
0L	01 007	LOAD M(7)	Load num to the AC
0R	06 008	SUB M(8)	AC ← AC - 1
1L	11 006	STOR M(6)	count ←AC
1R	10 002	JUMP +M(2, 20:39)	if AC > 0 go to instruction 2R
2L	00 000	HALT	Halt[**]
2R	01 009	LOAD M(9)	Load result to AC
3L	05 007	ADD M(7)	AC ← result + num
3R	11 009	STOR (M9)	Update result
4L	01 006	LOAD M(6)	AC ← count
4R	06 008	SUB M(8)	AC ← AC - 1
5L	11 006	STOR M(6)	Update count
5R	0E 001	JUMP M(1,20:39)	Jump to 1R
6	-		count data variable
7	3		num data variable
8	1		Data variable to represent '1'
9	-		result data variable

Task 4 – Exercise

Using the original instruction set for the IAS computer provided on the page 6 to write a program which computes $Y\ \sum_{1}^{100} x_i$. The numbers, X_i, are stored in consecutive memory locations beginning at 100. Store the result Y at memory location 300[††].

[5 marks]

Memory Address	Original IAS instruction	Comments

[**] NB: Halt is not part of the original IAS instruction set but is included here for compliance with the IAS SIM.

[††] If you wish to use IAS SIM to test your code it is recommended that you change 100 to a smaller number like 5 or 10. Ask the instructor if you wish to learn more advanced uses of IAS SIM.

IASSIM – IAS Instruction Set

Instruction name	Opcode (bin)	Opcode (dec)	Opcode (hex)	Description
S(x)->Ac+	00000001	1	01	Copy number in Selectron location x into AC
S(x)->Ac-	00000010	2	02	Copy the negative of the number in Selectron location x into AC
S(x)->AcM	00000011	3	03	Copy the absolute value of the number in Selectron location x into AC.
S(x)->Ac-M	00000100	4	04	Copy the negative of the absolute value of the number in Selectron location x into AC
S(x)->Ah+	00000101	5	05	Add number in Selectron location x into the accumulator.
S(x)->Ah-	00000110	6	06	Subtract number in Selectron location x from the accumulator.
S(x)->AhM	00000111	7	07	Add the absolute value of the number in Selectron location x into the accumulator.
S(x)->Ah-M	00001000	8	08	Subtract absolute value of number in Selectron location x from the accumulator.
S(x)->R	00001001	9	09	Copy number in Selectron location x into AR
R->A	00001010	10	0A	Copy number in AR to AC
S(x)*R->A	00001011	11	0B	Multiply number in Selectron location x by the number in AR. Place the left half of the result in AC and the right half in AR.
A/S(x)->R	00001100	12	0C	Divide the number in AC by the number in Selectron location x. Place the quotient in AR and the remainder in AC.
Cu->S(x)	00001101	13	0D	Continue execution at the left-hand instruction of the pair at Selectron location x.
Cu'->S(x)	00001110	14	0E	Continue execution at the right-hand instruction of the pair at Selectron location x.
Cc->S(x)	00001111	15	0F	If the number in AC is >= 0, continue execution at the left-hand instruction of the pair at Selectron location x. Otherwise continue normally.
Cc'->S(x)	00010000	16	10	If the number in AC is >= 0, continue execution at the right-hand instruction of the pair at Selectron location x. Otherwise continue normally.
At->S(x)	00010001	17	11	Copy the number in AC to Selectron location x
Ap->S(x)	00010010	18	12	Replace the right-hand 12 bits of the left-hand instruction at Selectron location x by the right-hand 12 bits of AC.
Ap'->S(x)	00010011	19	13	Replace the right-hand 12 bits of the right-hand instruction at Selectron location x by the right-hand 12 bits of AC.
L	00010100	20	14	Shift the number in AC to the left 1 bit (new bit on the right is 0)
R	00010101	21	15	Shift the number in AC to the right 1 bit (new bit on the left is whatever the original bit on the left was).
halt	00000000	0	00	Halt the program (causing either a bell to ring or a dialog box to appear)

IASSIM – IAS Instruction Set (Original Instructions)

Instruction Type	Opcode	Symbolic Representation	Description
Data transfer	00001010	LOAD MQ	Transfer contents of register MQ to the accumulator AC
	00001001	LOAD MQ,M(X)	Transfer contents of memory location X to MQ
	00100001	STOR M(X)	Transfer contents of accumulator to memory location X
	00000001	LOAD M(X)	Transfer M(X) to the accumulator
	00000010	LOAD –M(X)	Transfer –M(X) to the accumulator
	00000011	LOAD |M(X)|	Transfer absolute value of M(X) to the accumulator
	00000100	LOAD -|M(X)|	Transfer -|M(X)| to the accumulator
Unconditional branch	00001101	JUMP M(X,0:19)	Take next instruction from left half of M(X)
	00001110	JUMP M(X,20:39)	Take next instruction from right half of M(X)
Conditional branch	00001111	JUMP+M(X,0:19)	If number in the accumulator is nonnegative, take next instruction from left half of M(X)
	00010000	JUMP+M(X,20:39)	If number in the accumulator is nonnegative , take next instruction from right half of M(X)
Arithmetic	00000101	ADD M(X)	Add M(X) to AC; put the result in AC
	00000111	ADD |M(X)|	Add |M(X)| to AC; put the result in AC
	00000110	SUB M(X)	Subtract M(X) from AC; put the result in AC
	00001000	SUB |M(X)|	Subtract |M(X)} from AC; put the remainder in AC
	00001011	MUL M(X)	Multiply M(X) by M(Q); put most significant bits of result in AC, put less significant bits in M(Q)
	00001100	DIV M(X)	Divide AC by M(X); put the quotient in MQ and the remainder in AC
	00010100	LSH	Multiply accumulator by 2 (i.e., shift left one bit position)
	00010101	RSH	Divide accumulator by 2 (i.e., shift right one bit position)
Address modify	00010010	STOR M(X,8:19)	Replace left address field at M(X) by 12 right-most bits of AC
	00010011	STOR M(X,28:39)	Replace right address field at M(X) by 12 right-most bits of AC

End of Laboratory Exercise

Fun fact ☺ - The text book reports that data is represented sign and magnitude however the tool uses sign and magnitude.