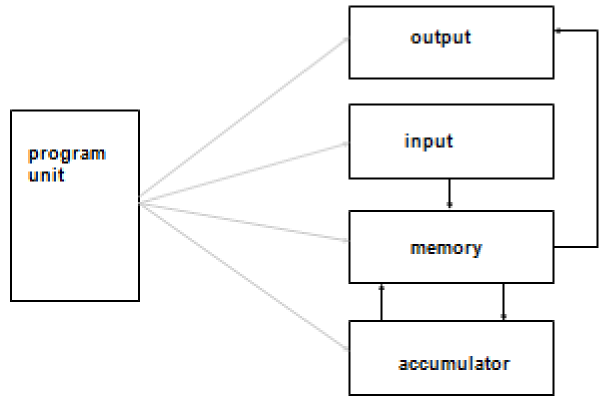
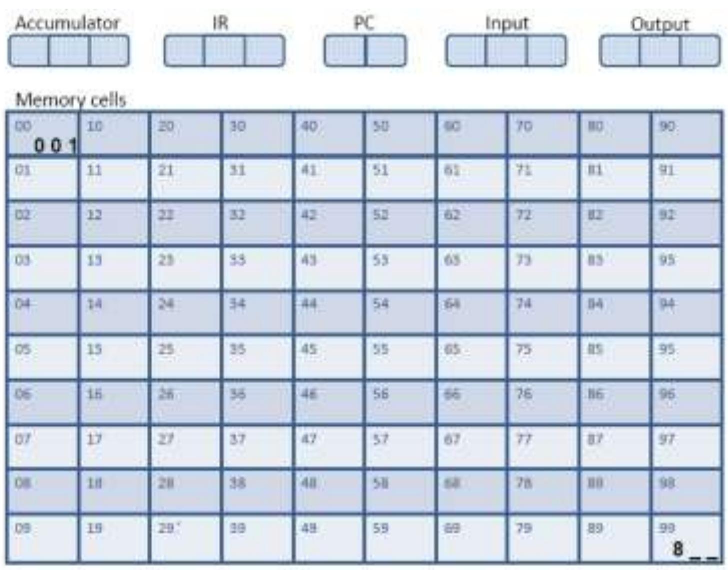
CARDIAC (Cardboard Illustrative Aid to Computing)

CARDIAC is a very simple computer, consisting of a small CPU, 100 memory cells, an input device and an output device. The CPU consists of just a 4-digit accumulator, and instruction decoder and a program counter. Operations are performed between memory and accumulator.

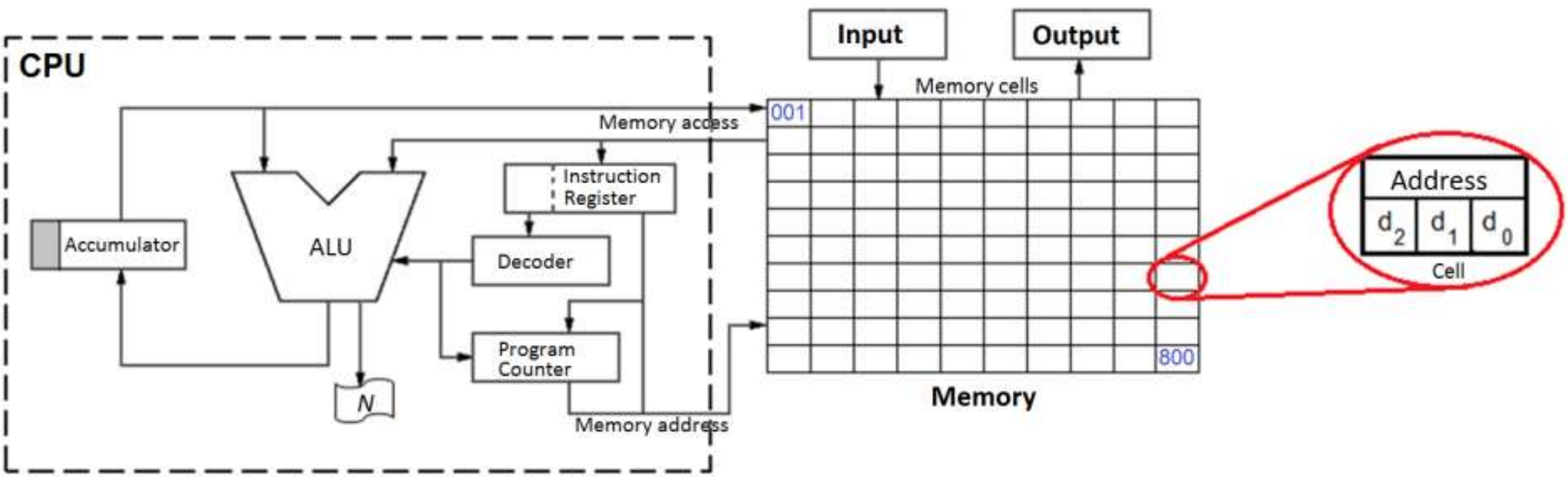


Instructions are 3-digit long, the first one indicating the operation to be executed, and the other to the memory cell to obtain the data. Instructions and operands are mixed in the same memory, and they are distinguished only by the number pointed by the program counter.



Op Code	Instruction	Example	Meaning
0	Input	012	memory[12] = input
1	Load	1 23	acc = memory[23]
2	Add	205	acc = acc + memory[5]
3	Branch if Less than Zero	312	If acc < 0 Then pc = 12
4	Shift	421	acc = shift left (acc , 2) + shift right(acc , 1)
5	Output	512	output = memory[12]
6	Store	623	memory[23] = acc
7	Subtraction	705	acc = acc – memory[5]
8	Jump	812	pc = 12
9	Stop	900	pc = 00, stop

CARDIAC Computational Model Architecture



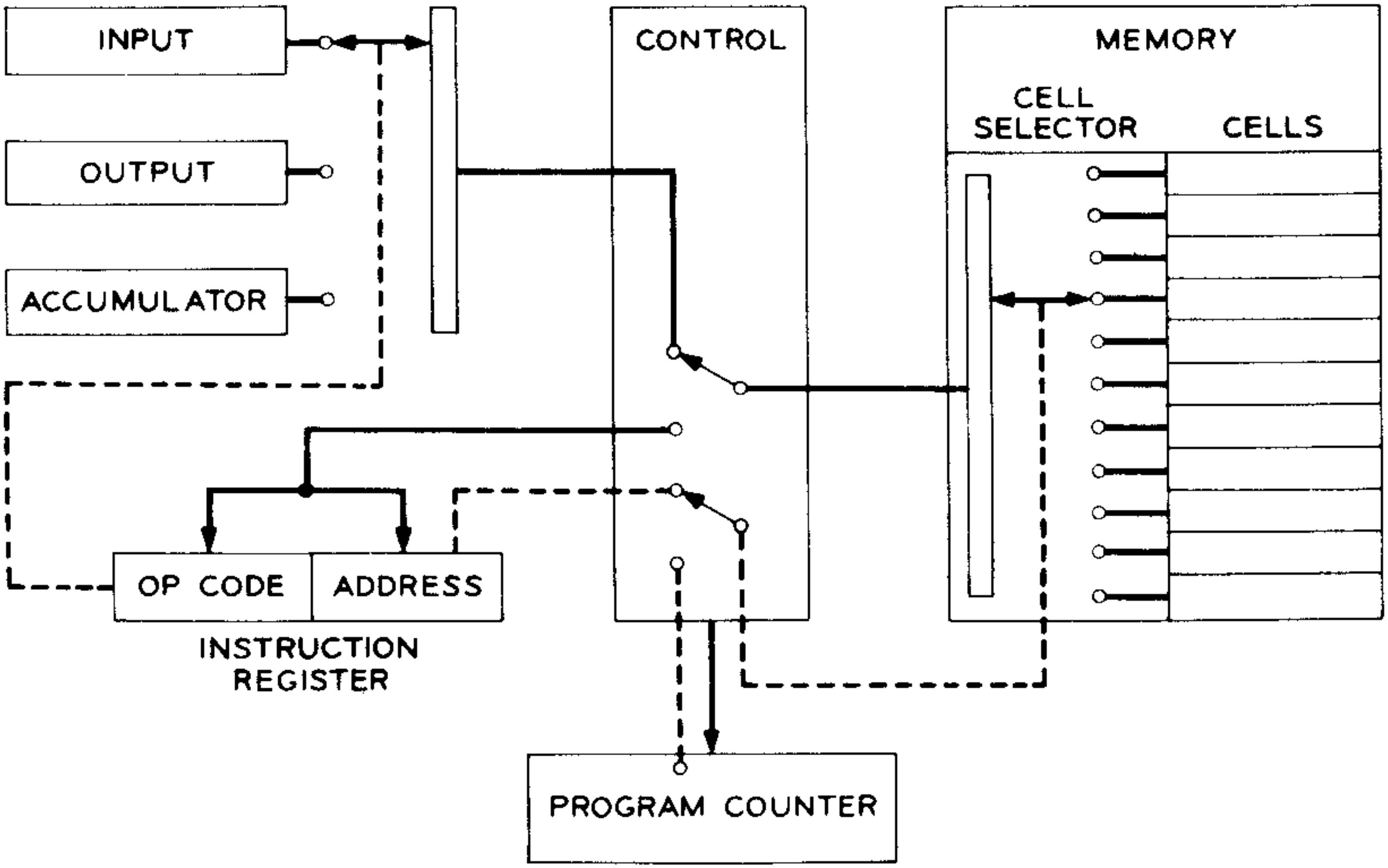
CARDIAC Assembly Language

Op Code	Mnemonic	Instruction	Example		Meaning	Comment
			Machine code	Assembly code		
0	INP	Input	**0**12	INP 12	memory[12] = input	Get any input and place it in memory address #12). Discard value at the input section.
1	LDA	Load	**1**23	LDA 23	acc = memory[23]	Replace the value in the accumulator with the number inside memory address #23.
2	ADD	Add	**2**05	ADD 5	acc = acc + memory[5]	Increase the content of the accumulator by the amount inside memory address #5.
3	BLZ	Branch if Less than Zero	**3**12	BLZ 12	If acc < 0 Then pc = 12	If the number in the accumulator is negative jump to the instruction in memory address #12, otherwise continue with the next instruction.
4	SHF	Shift	**4**21	SHT 21	acc = shift left (acc , 2), then acc = shift right(acc , 1)	Move the accumulator's content 2 positions to the left (filling with zeros the empty places and dropping the digits beyond the fourth position), then shift it one position to the right.
5	OUT	Output	**5**12	OUT 12	output = memory[12]	Copy content of memory location #12 and placed it into output section.
6	STO	Store	**6**23	STO 23	memory[23] = acc	Copy accumulator's content and place into memory location #23.
7	SUB	Subtraction	**7**05	SUB 5	acc = acc – memory[5]	Decrease the content of the accumulator by the amount inside memory address #5.
8	JMP	Jump	**8**12	JMP 12	pc = 12	Continue the program from memory location #12.
9	HLT	Halt	**9**00	HLT 00	pc = 00, stop	Stop program execution.

Note: The first digit of the machine code instruction it is the opcode, the last two digits indicate the memory cell to be used. (Except for instruction 4.)

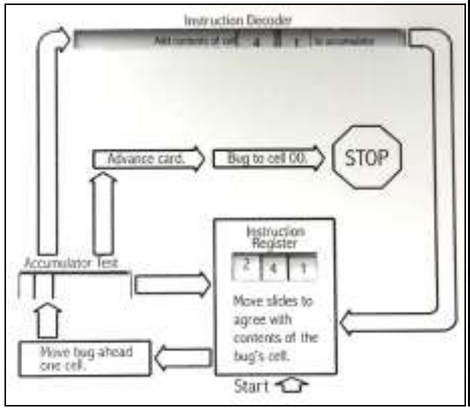
CARDIAC Assembly Language

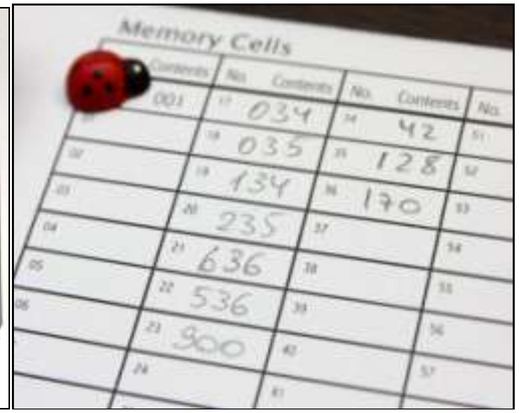
Main Functional Blocks of CARDIAC's Hardware



Note: The first digit of the machine code instruction it is the opcode, the last two digits indicate the memory cell to be used. (Except for instruction 4.)

CARDIAC machine code exercises





The following programs were written in the CARDIAC machine language. Each one is presented in the form *memory address*: *instruction*. For each one, write the equivalent assembly code and then, handtrace the program to figure its purpose out.

Program 1	**Program 2**	**Program 3**	**Program 4**	**Program 5**	**Program 6**
17 : 034	20 : 100	21 : 100	(variables)	07 : 068	15 : 039
18 : 035	21 : 603	22 : 603	00 : 001	08 : 404	16 : 139
19 : 134	22 : 503	23 : 503	19 : -004	09 : 669	17 : 431
20 : 235	23 : 200	24 : 200		10 : 070	18 : 640
21 : 636	24 : 603	25 : 822	(program)	11 : 170	19 : 139
22 : 536	25 : 503	26 : 900	20 : 119	12 : 700	20 : 413
23 : 900	26 : 200		21 : 200	13 : 670	21 : 240
	27 : 603		22 : 618	14 : 319	22 : 640
	28 : 503		23 : 518	15 : 169	23 : 139
	29 : 200		24 : 321	16 : 268	24 : 423
	30 : 603		25 : 900	17 : 669	25 : 410
	31 : 503			18 : 811	26 : 240
	32 : 200			19 : 569	27 : 640
	33 : 603			20 : 900	28 : 540
	34 : 503				29 : 900