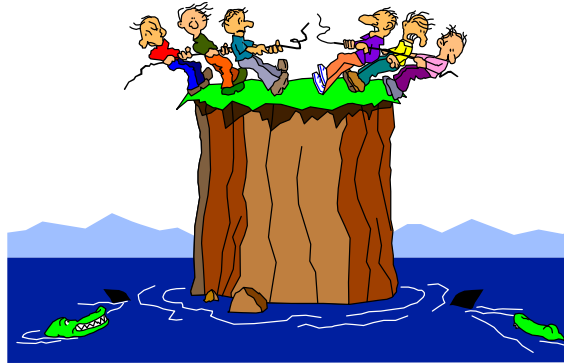


# Introduction to Firmware Theory

Software      Hardware



## Firmware

- The art and the technique of transforming hardware (logic systems) into software (programs) and vice versa
- Based on the central idea of equivalence between hardware and software
- The concept of firmware has evolved to mean almost any programmable content of a hardware

## Firmware

- From hardware engineer point of view: the sequential (programmed) behavior of hardware
- From software engineer point of view: the hardware interpretation of sequential program routine



Presented by Abramov B.

3

## Hardware System Representation

- Any combinational logic system with  $n$  input binary variables can be represented by various modes of representation:
  - a Truth table with  $2^n$  rows
  - a Karnaugh map of  $2^n$  cells
  - a Boolean cubes of  $n$  dimensions
  - a Binary Decision Tree with  $2^n$  branches

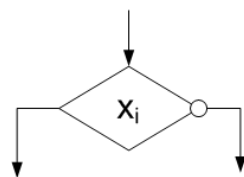
Presented by Abramov B.

4

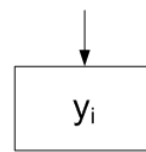
## BDT for system representation

- Any Boolean function can be represented as a rooted, directed, acyclic graph, which consists of decision nodes (test element) and terminal nodes (assignment or action element)

Test Element



Action Element

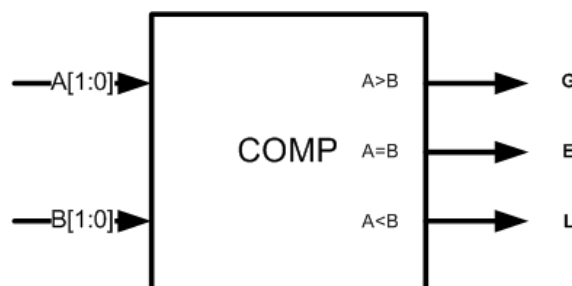


- The test and assignment elements constitute the two primitives of system specification

Presented by Abramov B.

5

## 2-bit comparator



Presented by Abramov B.

6

## 2-bit comparator: Truth Table

A <sub>1</sub>	B <sub>1</sub>	A <sub>0</sub>	B <sub>0</sub>	G	E	L
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	1	0	0
0	0	1	1	0	1	0
0	1	0	0	0	0	1
0	1	0	1	0	0	1
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	1	0	0
1	0	1	1	1	0	0
1	1	0	0	0	1	0
1	1	0	1	0	0	1
1	1	1	0	1	0	0
1	1	1	1	0	1	0

Presented by Abramov B.

7

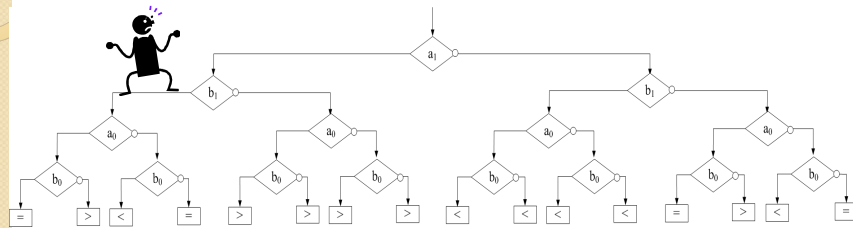
## 2-bit comparator: Truth Table

A <sub>1</sub>	B <sub>1</sub>	A <sub>0</sub>	B <sub>0</sub>	G	E	L
0	0	0	0	0	1	0
			1	0	0	1
		1	0	1	0	0
			1	1	0	1
	1	0	0	0	0	1
			1	0	0	1
1	0	0	0	1	0	0
			1	1	0	0
		1	0	1	0	0
			1	1	0	0
	1	0	0	0	1	0
			1	0	0	1
		1	0	1	0	0
			1	0	1	0

Presented by Abramov B.

8

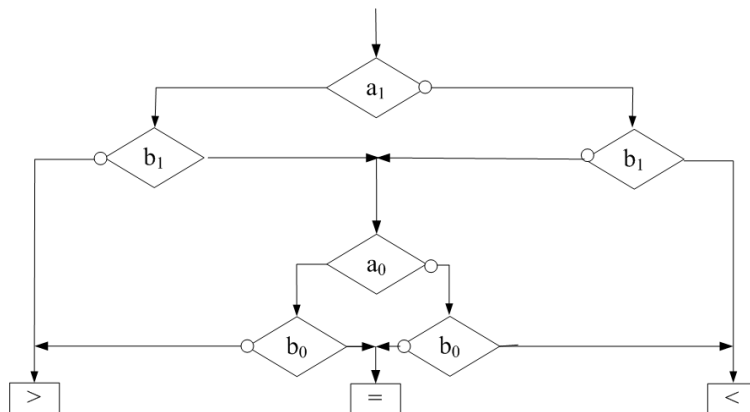
## 2-bit comparator: BDT



Presented by Abramov B.

9

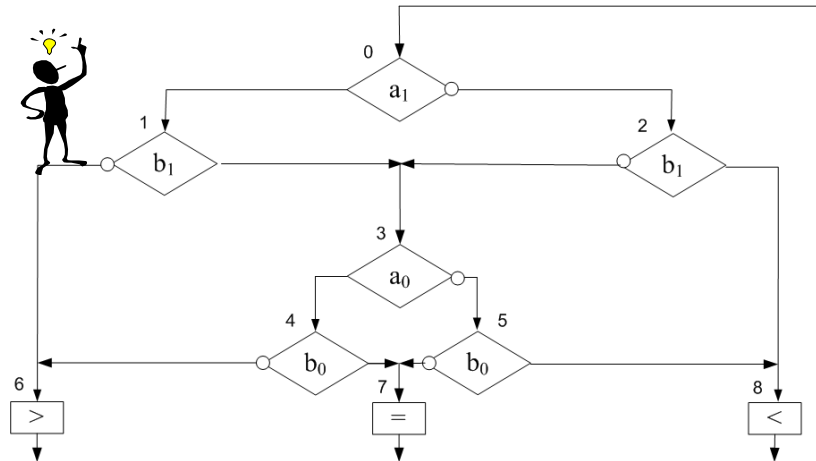
## 2-bit comparator: BDT-Optimized



Presented by Abramov B.

10

## Sequential execution hypothesis



Presented by Abramov B.

11

## Microprogrammed Logic System

- Any assemblage of logic elements, whether combinatorial or sequential constitutes a hardwired logic system
- If any memory is included it becomes a logic system with memory
- If the content of the memory is that of the primitives of specification- we have a microprogrammed logic system



Presented by Abramov B.

12

## Microprogrammed Logic System

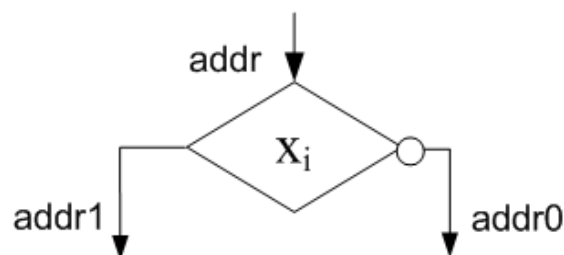
- Detailed definition of the primitives and realization of these in memory as microinstructions
- Representation of BDT by an assemblage of microinstructions
- Translation of these microinstructions into low level language mnemonic micro-program
- Realization in hardware of the language interpreter

Presented by Abramov B.

13

## Register Transfer Level Language: HL1

- A test microinstruction:  
*if  $x_i$  then addr1 else addr0*



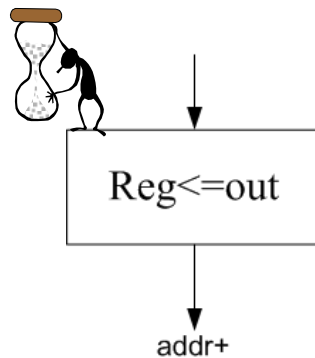
Presented by Abramov B.

14

## Register Transfer Level Language: HL1

- An assignment microinstruction:

*do reg ← out go to addr +*



Presented by Abramov B.

15

## Mnemonic micro-program

address	HL1 microinstruction
0	If $a_1$ then 1 else 2
1	If $b_1$ then 3 else 6
2	If $b_1$ then 8 else 3
3	If $a_0$ then 4 else 5
4	If $b_0$ then 7 else 6
5	If $b_0$ then 8 else 7
6	do reg ← [>] go to 0
7	do reg ← [=] go to 0
8	do reg ← [<] go to 0

Presented by Abramov B.

16

## Memory word format

- To store the HLI microinstruction in the memory we should have:
- A test variables coding
- An assignment (output) variables coding
- An opcode to distinguish between two types of microinstructions
- An instructions index (address) coding

Presented by Abramov B.

17

## Variables Coding

i	Test variable	$i_1$	$i_0$	Decision	G E L
0	$A_1$	0	0	$A > B$	1 0 0
1	$B_1$	0	1	$A = B$	0 1 0
2	$A_0$	1	0	$A < B$	0 0 1
3	$B_0$	1	1		

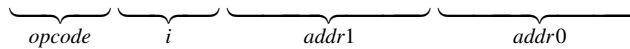
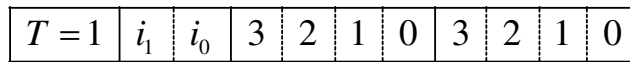


Presented by Abramov B.

18

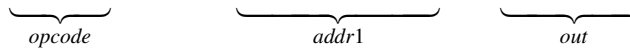
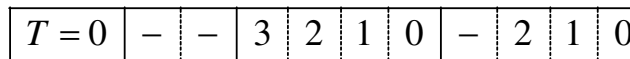
## Command format

11bit



*if xi then addr1 else addr0*

11bit



*do reg ← out go to addr +*

Presented by Abramov B.

19

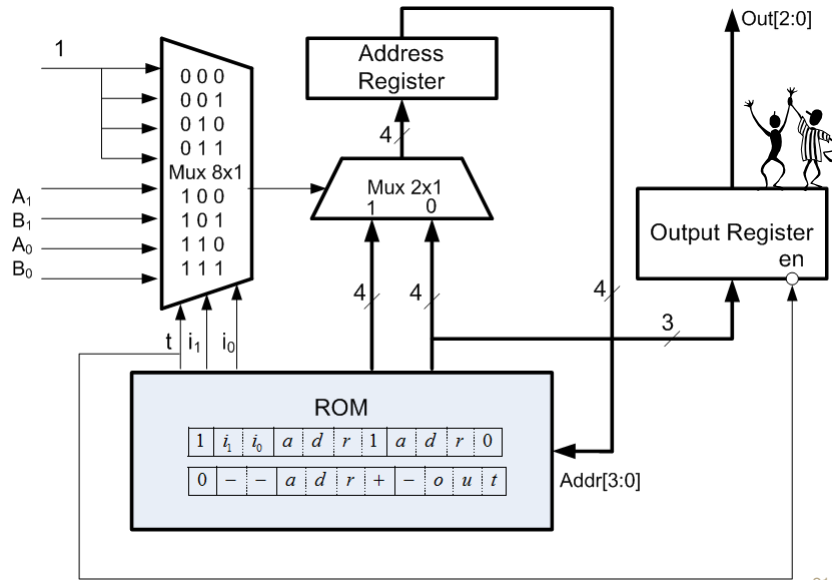
## Binary micro-program

Addr	T	$i_1 i_0$	Addr I/Addr+	Addr0/Out
0000	1	00	0001	0010
0001	1	01	0011	0110
0010	1	01	1000	0011
0011	1	10	0100	0101
0100	1	11	0111	0110
0101	1	11	1000	0111
0110	0	--	0000	-100
0111	0	--	0000	-010
1000	0	--	0000	-001

Presented by Abramov B.

20

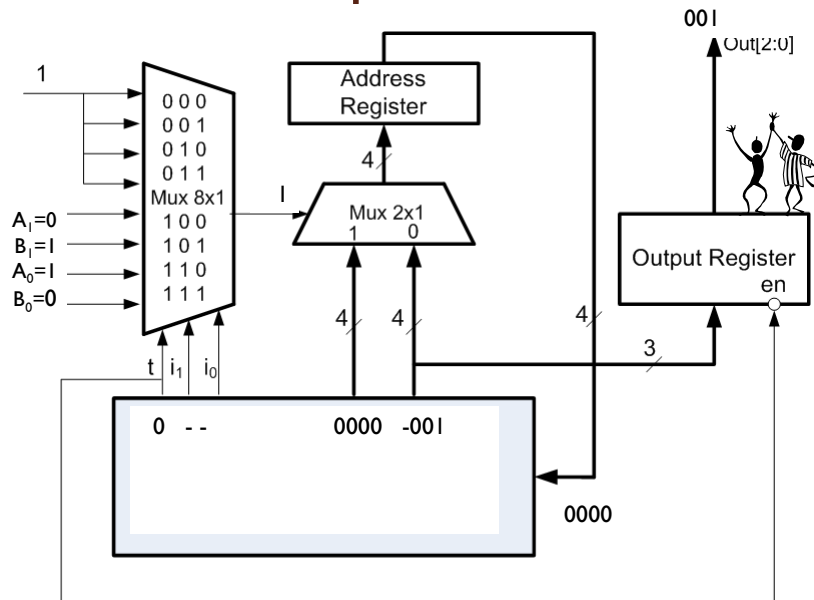
# Hardware Interpreter



Presented by Abramov B.

21

# Hardware Interpreter in Action



Presented by Abramov B.

22