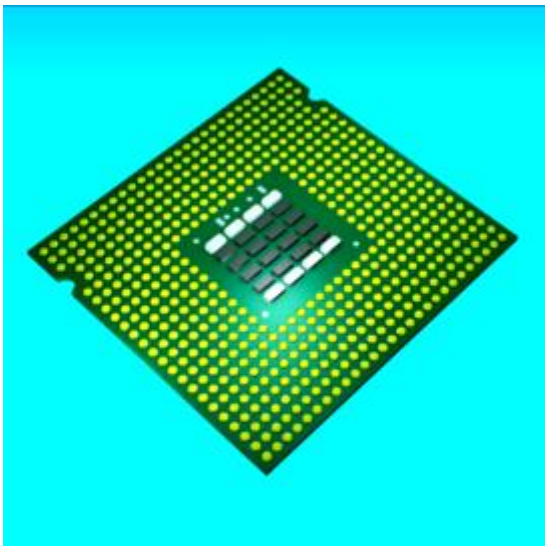

10 BIT MICROPROCESSOR DESIGN

A understandable CPU DESIGN

CONTENT

Here Your get a full overview of this project. How it work. How many operation it support. What is Its assembly code format etc.

Nabendu Bikash Binda

01. How many operands-
Answer: 3 operands [e.g.: z = x + y]

02. Types of operands (register/memory/mixed)-
Answer: Mixed

03. How many Operations? Why-
Answer: 8 Opcodes.
Explanation: We are designing ISA for 10 bit CPU where we want to run especially some given program. That's why we need implement total 8 instructions.

04. Types of operations? (Arithmetic, logical, branch type?? How many from each category? List the opcodes and respective binary values)-
Answer: 5 types

Sl. No	Operation Types	Operations	Register Type
01	Arithmetic	ADD	R Type
02	Logical	AND, OR	R Type
03	Data Transfer	ADDi, LW, SW	I-Type
04	Conditional Branch	BEQ	I-Type
05	Unconditional	J	J-Type

05. No. of format of instruction (how many different formats?)
Answer: 3 Types of formats. R-Type, I-Type and J-Type

06. Describe each of the format (fields and field length)
Answer:
R-Type Format: -

- OP Code field is 3 bits, RS, RT & RD field are 2 bits and SA field is 1 in length
- OP Code is an operation code that selects a specific operation
- RS and RT are the first and second source registers
- RD is the destination register
- SA is Shifting bit(how many bit I want to shift)

For example: ADD $0, $1, $2

000	01	10	00	0
OP Code (3 bits)	RS (2 bits)	RT (2 bits)	RD (2 bits)	SA (1 bit)

I-Type Format: -

- OP Code field is 3 bits and RS & RT field are 2 bits and Address/Immediate filed is 3 bits in length
- Load word, store word, branch type, & immediate type are I-type
- RS is a source register. It keeps an address for loads and stores or an operand for branch and immediate arithmetic instructions
- RT is a source register for branches but a destination register for the other I-type instructions

NABENDU BIKASH BINDA

For Example: LW $1, 3($2)

100	01	10	011
OP Code (3 bits)	RS (2 bits)	RT (2 bits)	Address/Immediate (3 bits)

J-Type Format: -

- OP Code field is 3 bits and Target field is 7 bits in length
- Unconditional jump is J-type instruction
- Target is label number in instruction memory where we want to jump

For Example: J 12

111	0001100
OP Code (3 bits)	Target (7 bits)

Format (fields and field length)

Operations	Opcode Reg (OP) 4 bits			Source Reg (RS) 2 bits		Secondary Reg (RT) 2 bits		Destination Reg (RD) 2 bits		Shift Bit (SA) 1 bit
	9	8	7	6	5	4	3	2	1	0
ADD	0	0	0	RS		RT		RD		SA
AND	0	0	1	RS		RT		RD		SA
OR	0	1	0	RS		RT		RD		SA
ADDi	0	1	1	RS		RT		Immediate		
LW	1	0	0	RS		RT		Immediate		
SW	1	0	1	RS		RT		Immediate		
BEQ	1	1	0	RS		RT		Immediate		
J	1	1	1	Target						

07. List of Registers?

Answer:

Name of the Registers	Register Number	Value Assigned (2 Bits)
$t0	0	00
$t1	1	01
$t2	2	10
$zero	3	11

Instruction Description:

ADD: It adds two registers and stores the result in destination register.

Operation: $d = $s + $t

Syntax: ADD $d, $s, $t

NABENDU BIKASH BINDA

AND: It AND's two register values and stores the result in destination register. Basically, it sets some bits to 0.
Operation: $d = $s && $t
Syntax: AND $d, $s, $t

OR: It OR's two register values and stores the result in destination register. Basically, it sets some bits to 1.
Operation: $d=$s || $t
Syntax: OR $d, $s, $t

ADDi: It ADD's a value from register with an integer value from immediate and stores the result in destination register.
Operation: $d = $s + immediate
Syntax: ADDi $d, $s, immediate

LW: It loads required value from the memory and write it back into the register.
Operation: $d = MEM[$s + offset]
Syntax: LW $d, offset($s)

SW: It stores specific value from register to memory.
Operation: MEM[$d + offset] = $s
Syntax: SW $s, offset($d)

BEQ: It checks whether the values of two registers are same or not. If it's same it performs the operation located in the address at offset value.
Operation: if ($s==$t) jump to offset
else go to next line
Syntax: BEQ $s, $t, offset

J: It Jump to target address

Operation: JUMP label

Syntax: j labelNo

Summary:

• This is a 10-bit RISC Type CPU. As an ISA designer, we have chosen 3 mixed type operands, 8 opcodes, 5 types of operations, 3 types of instruction format and finally we have given names and values of 4 different registers.

• This is a 10-bit CPU and 4 registers in it. For the design, we believe simplicity favors regularity. So, we have assigned 3 bits each to the opcodes and 2 bits to the RS, RT, RD and also 1 bit to shifting for R-type register and we also have assigned 3 bits each to the opcodes and 2 bits to the RS, RT & 3 bits for immediate in I-type register. In J-type we have assigned 3 bits for opcode and 7 bits for label.

NABENDU BIKASH BINDA

• Smaller is faster. Though it is only a 10-bit CPU, it will handle very common operations. In this CPU, we tried to use as many as operations that can handle, also keeping register's values in our mind.

NABENDU BIKASH BINDA