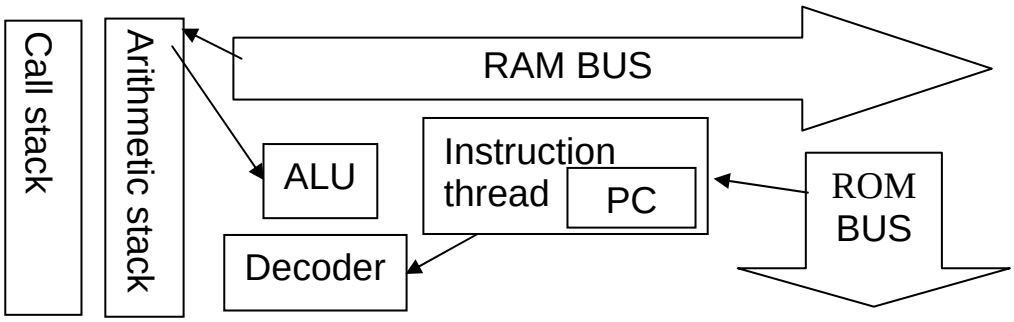
Pepelatz MISC (zero edition)

About

Pepelatz MISC is a small education processor written in Verilog. It can be used for learning HDLs or computer low-level structure.

Programmers architecture



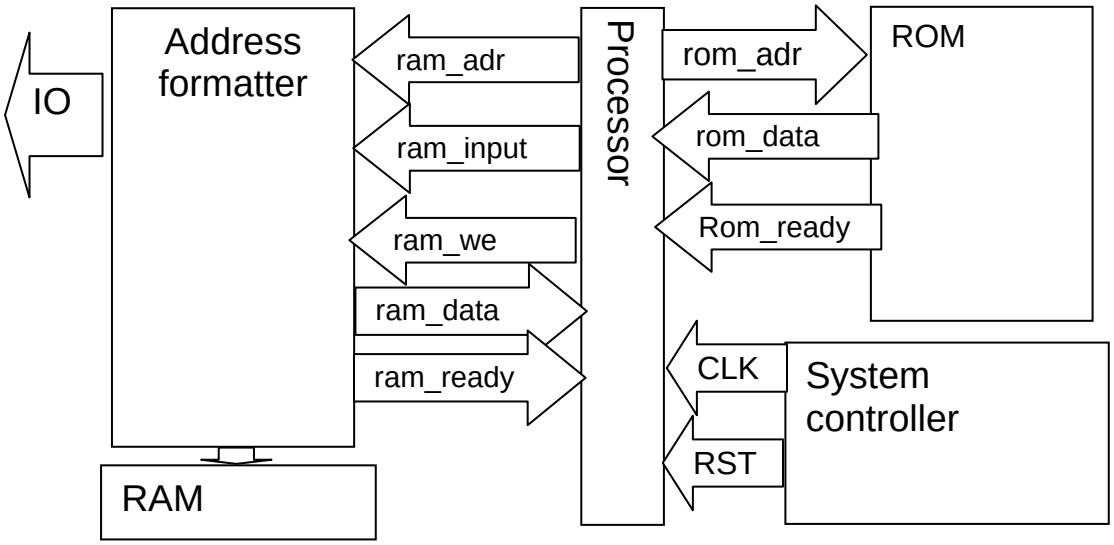
Pepelatz is a processor with stack-based architecture. It has two stacks: one for arithmetic operations and parameters and one for organization of loops and procedures.

In current version both stacks are stored inside processor. Arithmetic stack has 64 16-bite words, and call stack has 16.

Pepelatz has two buses: one for data and one for program. There is no IO bus: IO ports (if they are needed) can be placed in data area.

Instruction thread (ITh) is a very important path of architecture. It is a FIFO queue of commands. It is used as command cache.

Hardware architecture



Instructions

Each word (16 bites) in ITh is 3 5-bite instructions. All instructions have the same size. Operands (if it is needed) are placed in the next words of ITh. In this next I will name sets of 5-bite instructions commands.

The first bite in each command is operand flag. If it is on, an operand will be loaded into stack without using special instruction.

Num	Mnemonic	Comment
00	add	
01	sub	
02	and	
03	or	
04	xor	
05	shr	
06	shl	
07	drop	SP--;
08	inc	
09	dec	
0A	not	
0B	svap	S0<=S1;S1<=S0
0C	rot	S0<S2;S2<=0
0D	store	Save value to memory. *Does not SP--!!!
0E	setcall	Push value from arithmetic stack to call stack.*
0F	dropcall	Drops call stack.*
10	const	Loads a constant from ITh.
11	load	Loads [S0] from memory.*
12	getcall	Gets value from call stack. Does not drop call stack!
13	dup	Duplicates S0.
14	dupd	Duplicates S1 (to stack top).
15	isover	Overflow control.
16	isneg	Negate control.
17	nop	
18	loop	Starts current command from beginning.
19	loop1	Loads an operand and starts current command from beginning.
1A	skip	Goes to the next command.
1B	call	Push PC into call stack and go to address
1C	if	if S0=0 then jmp
1D	jump	PC=S0
1E	loopz	If C[0]!=0, dec C[0] and starts current command from beginning.
1F	ifloop	If C[0]!=0, dec C[0] and jmp.

Warring I. PC is a pointer to the next command, not instruction. Jump and call is not jumping. Jumping will happen after the end of current instruction or after skip command.

Warring II. After changing PC, operands will be read from the new address!

Note I. After changing PC, ITh will be overwritten.