Instruction set architecture (RISC)

Instruction formats

1. RRR type

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Opcode					Reg A			Reg B			Reg C			Unused									

2. RI type

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Opcode					Reg A			Immediate															

3. I type

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Opcode					Immediate																		

Architecture

Data Memory (RAM) – 8M x 16 bits

Instruction Memory (ROM) – 4K x 24 bits

MAR – 24-bit Register

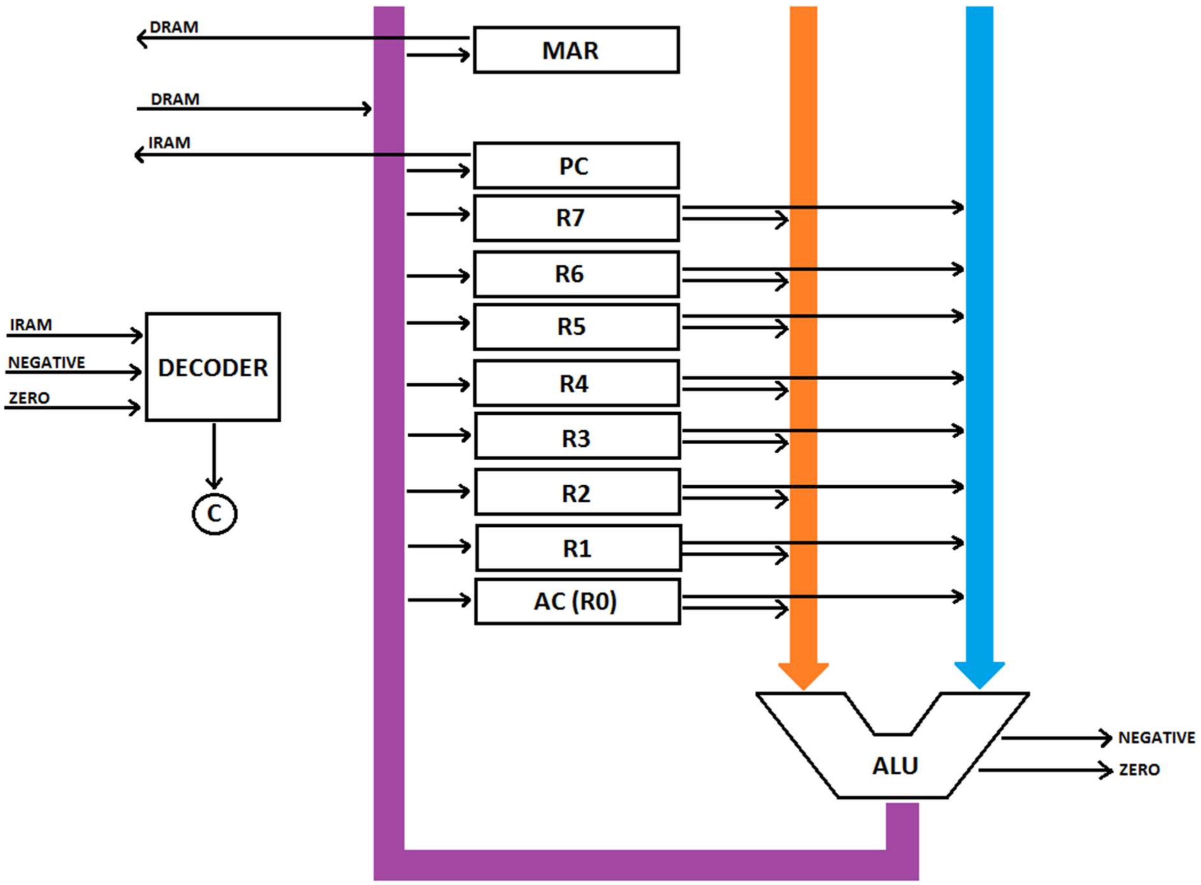
PC – 24-bit Register

Accumulator (AC, R0) – 24-bit Register

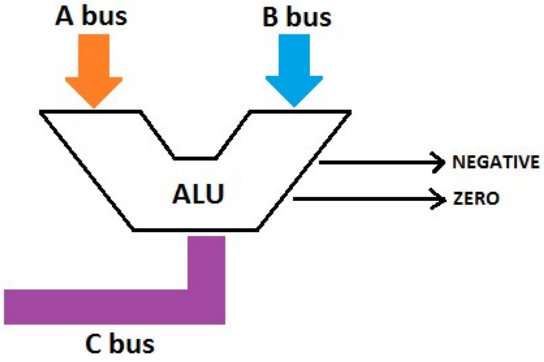
R1 – R7 General Purpose Registers – 24-bit

Instruction	Opcode	operation
NOP	00000	No operation
LOAD	00001	Reg A ← RAM[MAR]
STORE	00010	RAM[MAR] ← Reg A
MOVE	00011	Reg A ← Reg B
LDPC	00100	PC ← Reg A
LDMAR	00101	MAR ← Reg A
LOADI	00110	Reg A ← signed immediate (16-bit)
LDACI	00111	AC ← signed immediate (19-bit)
ADD	01000	Reg A ← Reg B + Reg C
SUB	01001	Reg A ← Reg B - Reg C
MUL	01010	Reg A ← Reg B << Reg C
DIV	01011	Reg A ← Reg B >> Reg C
INC	01100	Reg A ← Reg A + 1
DEC	01101	Reg A ← Reg A - 1
NEG	01110	Reg A ← -Reg B
NOT	01111	Reg A ← Reg B (NOT) Reg C
AND	10000	Reg A ← Reg B (AND) Reg C
OR	10001	Reg A ← Reg B (OR) Reg C
XOR	10010	Reg A ← Reg B (XOR) Reg C
JGT	10011	If ALU out > 0 then PC ← Reg A else PC ← PC + 1
JEQ	10100	If ALU out = 0 then PC ← Reg A else PC ← PC + 1
JGE	10101	If ALU out >= 0 then PC ← Reg A else PC ← PC + 1
JLT	10110	If ALU out < 0 then PC ← Reg A else PC ← PC + 1
JNE	10111	If ALU out != 0 then PC ← Reg A else PC ← PC + 1
JLE	11000	If ALU out <= 0 then PC ← Reg A else PC ← PC + 1
JMP	11001	PC ← Reg A (Unconditional Jump)

Data path


DRAM
MAR
DRAM
IRAM
PC
R7
R6
R5
R4
R3
R2
R1
AC (R0)
IRAM
NEGATIVE
ZERO
DECODER
C
ALU
NEGATIVE
ZERO

Arithmetic and Logic Unit (ALU)



Opcode	ALU control bits	Operation
01000	0000	ADD (A + B)
01001	0001	SUB (A – B)
01010	0010	MUL (A << B)
01011	0011	DIV (A >> B)
01100	0100	INC (A + 1)
01101	0101	DEC (A – 1)
01110	0110	NEG (–A)
01111	0111	NOT (!A)
10000	1000	AND (A & B)
10001	1001	OR (A | B)
10010	1010	XOR (A ^ B)

If ALU output = 0, then Z = 1 else Z = 0

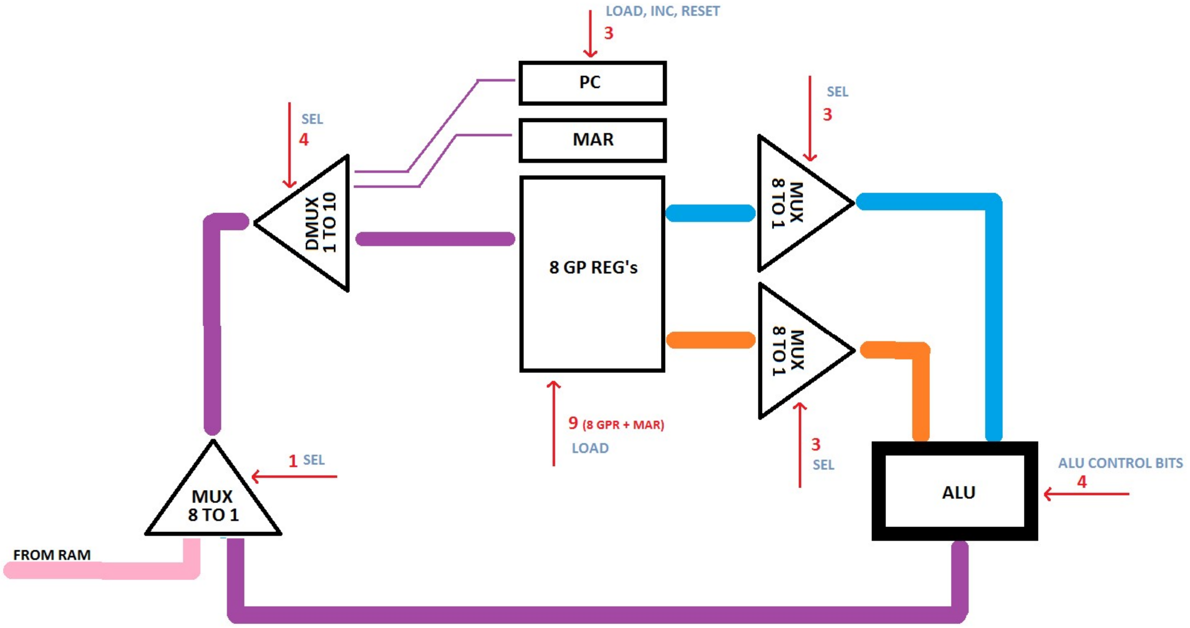
If ALU output < 0, the N = 1 else N = 0

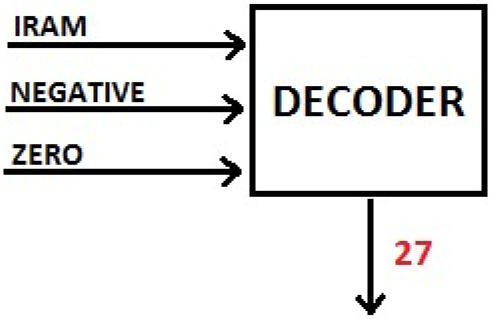
Jump Logic

Opcode	Operation	Z flag	N flag	Jump
JGT	If ALU out > 0	0	0	PC ← Reg A
JEQ	If ALU out = 0	1	x	PC ← Reg A
JGE	If ALU out >= 0	x	0	PC ← Reg A
JLT	If ALU out < 0	0	1	PC ← Reg A
JNE	If ALU out != 0	0	x	PC ← Reg A
JLE	If ALU out <= 0	x	1	PC ← Reg A
JMP	Unconditional Jump	x	x	PC ← Reg A

Control bits

Depending on the opcode and ALU flags the decoder will generate the control bits


LOAD, INC, RESET
3
PC
MAR
SEL
4
DMUX
1 TO 10
8 GP REG's
SEL
3
MUX
8 TO 1
MUX
8 TO 1
3
SEL
9 (8 GPR + MAR)
LOAD
1 SEL
MUX
8 TO 1
FROM RAM
ALU
ALU CONTROL BITS
4


IRAM
NEGATIVE
ZERO
DECODER
27