Instruction Name	OP Code	Description
STI	1 0 X X X X X R	Store to memory immediately
LDI	1 1 X X X X X R	Load to register immediately
LRI	0 1 0 X X X X R	Load register immediately from opcode
BRNI	0 1 1 X X X X X	Branch to instruction immediately (always checks zero flag)
SRL	0 0 1 1 0 0 0 R	Shift Right Logical
SRA	0 0 1 1 0 1 0 R	Shift Right Arithmatical
SRRL	0 0 1 1 1 0 0 R	Shift Right with Rotate Logical
SRRA	0 0 1 1 1 1 0 R	Shift Right with Rotate Arithmatical
SLL	0 0 1 0 0 0 0 R	Shift Left Logical
SLA	0 0 1 0 0 1 0 R	Shift Left Arithmetical
SLRL	0 0 1 0 1 0 0 R	Shift Left with Rotate Logical
SLRA	0 0 1 0 1 1 0 R	Shift Left with Rotate Arithmetical
PUSH	0 0 0 0 1 0 0 R	PUSH register to stack
POP	0 0 0 0 1 1 0 R	POP from stack to register
BRNR	0 0 0 0 0 0 0 R	Branch to instruction memory which register points to (always checks ZeroFlag)
MOV	0 0 0 0 0 0 1 R	Move data from one register to the other
CLE	0 0 0 0 0 1 0 0	Clear Extended flag
HLT	0 0 0 0 0 1 0 1	Halt
CLZ	0 0 0 0 0 1 1 0	Clear ZeroFlag for unconditional jump
NOT	0 0 0 1 0 0 0 R	NOT specified register
INC	0 0 0 1 0 0 1 R	Increase specified register
DECR	0 0 0 1 0 1 0 R	Decrease specified register
CLR	0 0 0 1 0 1 1 R	Clear specified register
AND	0 0 0 1 1 0 0 R	AND two registers then move result to R
OR	0 0 0 1 1 0 1 R	OR two registers then move result to R
ADD	0 0 0 1 1 1 0 R	ADD two registers then move result to R
SUB	0 0 0 1 1 1 1 R	SUB register R from the other register then move result to R

In this ISA we need instruction memory , data memory , a separate stack memory , register files with 2 registers and 1 stack pointer .

Instruction Memory : 256 Bytes
Stack Memory : 256 Bytes
Data Memory : 32 Bytes
RF : 2 Registers with 8 bits possible data
Stack Pointer : 1 stack pointer to point to stack memory .
NOTE : I combined all stack requirements including SP into one big module named “Stack” . This is not a good architecture and should be changed ASAP .