Instructions			Byte 1		Byte 2			Instruction Explanation
Type	Full Name	Mnemonic	Opcode	Operands				
System	No Operation	NOP	□□□□□					Does Nothing
	Halt CPU	HLT	□□□□■					Halts CPU
	Jump	JMP	□□□■□	Page	Address			Unconditional jump to page:address
	Branch on Condition	BRH	□□□■■	Condition	Address			Branches inside page if Condition is met
	Push Call Stack	CALL	□□■□□	Page	Address			Push PC to Call Stack and JMP
	Pop Call Stack	RET	□□■□■					Pop Call Stack and JMP to top+1
Registers	Load Immediate	LDI	□□■■□	Reg A	Immediate			A = IMM
	Move	MOV	□□■■■	Reg A			Reg B	A = B
ALU	Add	ADD	□■□□□	Reg A	Reg C		Reg B	C = A + B
	Subtract	SUB	□■□□■	Reg A	Reg C		Reg B	C = A - B
	Add Immediate	ADDI	□■□■□	Reg A	Immediate			A += IMM
	Add Sign. Ext. IMM	ADSI	□■□■■	Reg A	Reg C	Sign Ext. IMM		C = A + Sign Ext. IMM
	Bitwise XOR	XOR	□■■□□	Reg A	Reg C		Reg B	C = A ^ B
	Bitwise AND	AND	□■■□■	Reg A	Reg C		Reg B	C = A & B
	Bitwise OR	OR	□■■■□	Reg A	Reg C		Reg B	C = A | B
	Compare	CMP	□■■■■	Reg A			Reg B	A - B, Store Flags; No WB
	Bitwise XOR Immediate	XORI	■□□□□	Reg A	Immediate			A ^= IMM
	Bitwise AND Immediate	ANDI	■□□□■	Reg A	Immediate			A &= IMM
	Bitwise OR Immediate	ORI	■□□■□	Reg A	Immediate			A |= IMM
	Compare Immediate	CMPI	■□□■■	Reg A	Immediate			A - IMM, Store Flags; No WB
Barrel Shifter	Barrel Right Shift [9]	RSH	■□■□□	Reg A	Reg C		Reg B	C = A >> B
	Barrel Left Shift	LSH	■□■□■	Reg A	Reg C		Reg B	C = A << B
	Barrel Rotate Left	RTL	■□■■□	Reg A	Reg C		Reg B	C = A rotl. B
	Arithmetic Right Shift	ARS	■□■■■	Reg A	Reg C		Reg B	C = A >> B
	BRS Immediate [10]	RSHI	■■□□□	Reg A	Reg C		IMM	C = A >> IMM
	BLS Immediate	LSHI	■■□□■	Reg A	Reg C		IMM	C = A << IMM
	BRL Immediate	RTLI	■■□■□	Reg A	Reg C		IMM	C = A rotl. IMM
	ARS Immediate	ARSI	■■□■■	Reg A	Reg C		IMM	C = A >> IMM
Memory	Memory Store	MST	■■■□□	Reg A			Reg B	Mem[*B] = A
	Memory Load	MLD	■■■□■	Reg A			Reg B	A = Mem[*B]
Ports	Port Store	PST	■■■■□	Reg A			Port	Ports[Port] = A
	Port Load	PLD	■■■■■	Reg A		R [11	Port	A = Ports[Port]

Specifications			
Type	Description	Mnemonic	Opcode
Branch Condition	Equal to Zero [1]	BEQ	□□□
	Not Equal to Zero [2]	BNE	□□■
	Greater than Zero [3]	POS	□■□
	Lesser than Zero [4]	NEG	□■■
	Pos or Equal to Zero [5]	PEQ	■□□
	Neg or Equal to Zero [6]	NEQ	■□■
	Even [7]	EVN	■■□
	Signed Overflow [8]	SOF	■■■

[1] Zero
if set to BEQ R0 R0, it always jumps

[2] !Zero

[3] !MSB && !Zero

[4] MSB

[5] !MSB

[6] MSB || Zero

[7] !LSB

[8] MSB ^ Bit7

[9] Note: When shifting right, the amount to shift by is interpreted as a negative number, that is:

1. 1000 0000 >> 001 = 0000 0001;
2. 1000 0000 >> 111 = 0100 0000.

001 is interpreted as 'shift by 7';
111 is interpreted as 'shift by 1'.

[10] Read the note for BRS

[11] If R, read from the random Port