Instruction Set - Simple Instruction Set

You

January 25, 2026

1 Introduction

Explaination of the instruction set for the simple CPU (8-bit "homemade" CPU), each instruction wil be explaind how the intecract with the varius components of the cpu, the memory and other types of I/O.

Contents

2 CPU FLAGs

The CPU has 5 flags used for comparing values for contitional code jumping og code executions. Examples are If statment, Switch statments and For loops and While loops:

Character	Type
N	Negative
Z	Zero
O	Overflow
U	Underflow
C	Carry

2.1 Negative Flag

If the result of the last ALU-operation had a '1' as the MSB (Most Segnificant Bit) The Negative flag will become '1', as we are working with 2's compliments.

Example:

Input 1	Input 2	Result
XXXXXXXX	**XXXXXXXX**	**1XXXXXXX**
XXXXXXXX	No input (00...)	**1XXXXXXX**

2.2 Zero Flag

If the result of the last ALU-operation had all bits equal to '0' then the Zero flag will become '1'.

Example:

Input 1	Input 2	Result
XXXXXXXX	**XXXXXXXX**	**00000000**
XXXXXXXX	No input (00...)	**00000000**

2.3 Overflow Flag

If the MSB in result is differant than the registers going inn the ALU, The Overflow flag will become '1'. AKA two **positive** numbers go inn but the result is **negative**

Input 1	Input 2	Result
0XXXXXXX	**0XXXXXXX**	**1XXXXXXX**
1XXXXXXX	**1XXXXXXX**	**0XXXXXXX**
0XXXXXXX	No input (00...)	**1XXXXXXX**

2.4 Underflow Flag

Can only occurre if the LSB (Least significant bit) is '1' and you **Right Shift** the value.

	Input 1	Result
Right Shift	**XXXXXXX1**	**0XXXXXXX**

2.5 Carry Flag

If inn the add result of the last ALU-operation got too big and overflows above the 8'th and MSB bit the Carry Flag will be set

3 Register

There are 4 x 8-bit general purpose registers Register A, B, C and D. These registers can be used in the ALU and for loading to and from memory and be able to move between each other.

PC (program counter) is a 2x8 bit register that holds the memory address

4 Instructions

The CPU reads 8-bit Op-Codes and can only decode 26 differant master instructions

nr	Assembly Instruction	OpCode
0	**Jump**	**0x02**
1	**BOP**	**0x03**
2	**BON**	**0x04**
3	**BOZ**	**0x05**
4	**BNZ**	**0x06**
5	**BOO**	**0x07**
6	**BNO**	**0x08**
7	**BOU**	**0x09**
8	**BNU**	**0x0a**
9	**BOC**	**0x0b**
10	**BNC**	**0x0c**
11	**RST**	**0x0d**
12-14	**Clear**	**0x0e - 0x13**
15	**Load**	**0x14 - 0x17**
16	**Store**	**0x18 - 0x1b**
17	**Inc**	**0x1c - 0x1f**
18	**Dec**	**0x20 - 0x23**
19	**Right Shift**	**0x24 - 0x27**
20	**Not**	**0x28 - 0x2b**
21	**And**	**0x30 - 0x3f**
22	**Or**	**0x40 - 0x4f**
23	**Xor**	**0x50 - 0x5f**
24	**Add**	**0x60 - 0x6f**
25	**Addc**	**0x70 - 0x7f**
26	**Move**	**0x80 - 0x8f**

4.1 Jump

Jump command is used to unconditionaly jump to spesific 16 bit (2 bytes) memory locations

Jump "00000010"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	
Jump 0x1234	0x02	3	3	-	-	-	-	-	Setts PC to 0x1234

4.2 BOP - Branch On Positive

This Comand Branches to a spesific 16 bit (2 bytes) memory location
only if flags : Z ='0' & N = '0'

BOP "00000011"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	
BOP 0x1234	0x03	3	3	-	-	-	-	-	Setts PC to 0x1234 if N & Z = '0'

4.3 BON - Branch On Negative

Branches to a spesific 16 bit (2 bytes) memory location
only if flag : N = '1'

BON "00000100"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	
BON 0x1234	0x04	3	3	-	-	-	-	-	Setts PC to 0x1234 if N = '1'

4.4 BOZ - Branch On Zero

Branches to a spesific 16 bit (2 bytes) memory location
only if flag : Z = '1'

BOZ "00000101"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
BOZ	0x1234	0x05	3	3	-	-	-	-	-	Setts PC to 0x1234 if Z = '1'

4.5 BNZ - Branch Not Zero

Branches to a spesific 16 bit (2 bytes) memory location
only if flag : Z = '0'

BNZ "00000110"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
BNZ	0x1234	0x06	3	3	-	-	-	-	-	Setts PC to 0x1234 if Z = '0'

4.6 BOO - Branch On Overflow

Branches to a spesific 16 bit (2 bytes) memory location
only if flag : O = '1'

BOO "00000111"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
BOO	0x1234	0x07	3	3	-	-	-	-	-	Setts PC to 0x1234 if O = '1'

4.7 BNO - Branch Not Overflow

Branches to a spesific 16 bit (2 bytes) memory location
only if flag : O = '0'

BNO "00001000"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
BNO	0x1234	0x08	3	3	-	-	-	-	-	Setts PC to 0x1234 if O = '0'

4.8 BOU - Branch On Underflow

Branches to a spesific 16 bit (2 bytes) memory location
only if flag : U = '1'

BOU "00001001"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
BOU	0x1234	0x09	3	3	-	-	-	-	-	Setts PC to 0x1234 if U = '1'

4.9 BNU - Branch Not Underflow

Branches to a spesific 16 bit (2 bytes) memory location
only if flag : U = '0'

BNU "00001010"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
BNU	0x1234	0x0a	3	3	-	-	-	-	-	Setts PC to 0x1234 if U = '0'

4.10 BOC - Branch On Carry

Branches to a spesific 16 bit (2 bytes) memory location
only if flag : C = '1'

BOC "00001011"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
BOC	0x1234	0x0b	3	3	-	-	-	-	-	Setts PC to 0x1234 if C = '1'

4.11 BNC - Branch Not Carry

Branches to a spesific 16 bit (2 bytes) memory location
only if flag : C = '0'

BNC "00001100"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
BNC	0x1234	0x0C	3	3	-	-	-	-	-	Setts PC to 0x1234 if C = '0'

4.12 RST - Reset

Resets everything about the cpu. (Dont use it; not synced)

RST "00001101"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
RST	0x1234	0x0d	1	1	-	-	-	-	-	Resets everything

4.13 Clear

Clear can clear the flag Register, individual or all registers

Clear				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	
Clear f	0x0e	1	1	0	0	0	0	0	Clears Flags
Clear r	0x0f	1	1	-	-	-	-	-	Clears all registers
Clear a	0x10	1	1	-	-	-	-	-	Clears register A
Clear b	0x11	1	1	-	-	-	-	-	Clears register B
Clear c	0x12	1	1	-	-	-	-	-	Clears register C
Clear d	0x13	1	1	-	-	-	-	-	Clears register D

4.14 Load

This command loads 1 byte from memory into one of the registers

Load "000101XX"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
Load a	0x1234	0x14	3	4	-	-	-	-	-	Loads from memory 0x1234 to register A
Load b	0x1234	0x15	3	4	-	-	-	-	-	Loads from memory 0x1234 to register B
Load c	0x1234	0x16	3	4	-	-	-	-	-	Loads from memory 0x1234 to register C
Load d	0x1234	0x17	3	4	-	-	-	-	-	Loads from memory 0x1234 to register D

4.15 Store

This command stores / saves the select registers into memory

Store "000110XX"					Set Flags					Comment
Ex Instruction		OpCode	Bytes	Clocks	N	Z	O	U	C	
Store a	0x1234	0x18	3	4	-	-	-	-	-	Stores Register A to memory 0x1234
Store b	0x1234	0x19	3	4	-	-	-	-	-	Stores Register B to memory 0x1234
Store c	0x1234	0x1a	3	4	-	-	-	-	-	Stores Register C to memory 0x1234
Store d	0x1234	0x1b	3	4	-	-	-	-	-	Stores Register D to memory 0x1234

4.16 INC - Increment

This command incriments one of the registers

Inc "000111XX"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	
Inc a	0x1c	1	1	+	+	+	0	+	Increments register A
Inc b	0x1d	1	1	+	+	+	0	+	Incriments register B
Inc c	0x1e	1	1	+	+	+	0	+	Incriments register C
Inc d	0x1f	1	1	+	+	+	0	+	Incriments register D

4.17 DEC - Decriment

This command decriments one of the registers

Dec "001000XX"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	
Dec a	0x20	1	1	+	+	+	0	+	Decriments register A
Dec b	0x21	1	1	+	+	+	0	+	Decriments register B
Dec c	0x22	1	1	+	+	+	0	+	Decriments register C
Dec d	0x23	1	1	+	+	+	0	+	Decriments register D

4.18 Right shift

This command right shifts the selected registers

Rs "001001XX"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	
Rs a	0x24	1	1	0	+	+	+	0	Right Shifts A
Rs b	0x25	1	1	0	+	+	+	0	Right Shifts B
Rs c	0x26	1	1	0	+	+	+	0	Right Shifts C
Rs d	0x27	1	1	0	+	+	+	0	Right Shifts D

4.19 Not

This command not one of the registers

Not "001010XX"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	
Not A	0x28	1	1	+	+	+	0	0	Nots Regiser A
Not B	0x29	1	1	+	+	+	0	0	Nots Regiser B
Not C	0x2a	1	1	+	+	+	0	0	Nots Regiser C
Not D	0x2b	1	1	+	+	+	0	0	Nots Regiser D

4.20 And

This command and's 2 registers

And "0011XXXX"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	(Registers)
And A A	0x30	1	1	+	+	0	0	0	A <= A and A
And B A	0x31	1	1	+	+	0	0	0	B <= B and A
And C A	0x32	1	1	+	+	0	0	0	C <= C and A
And D A	0x33	1	1	+	+	0	0	0	D <= D and A
And A B	0x34	1	1	+	+	0	0	0	A <= A and B
And B B	0x35	1	1	+	+	0	0	0	B <= B and B
And C B	0x36	1	1	+	+	0	0	0	C <= C and B
And D B	0x37	1	1	+	+	0	0	0	D <= D and B
And A C	0x38	1	1	+	+	0	0	0	A <= A and C
And B C	0x39	1	1	+	+	0	0	0	B <= B and C
And C C	0x3a	1	1	+	+	0	0	0	C <= C and C
And D C	0x3b	1	1	+	+	0	0	0	D <= D and C
And A D	0x3c	1	1	+	+	0	0	0	A <= A and D
And B D	0x3d	1	1	+	+	0	0	0	B <= B and D
And C D	0x3e	1	1	+	+	0	0	0	C <= C and D
And D D	0x3f	1	1	+	+	0	0	0	D <= D and D

4.21 Or

This command or's 2 registers.

Or "0100XXXX"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	(Registers)
Or A A	0x40	1	1	+	+	0	0	0	A <= A or A
Or A B	0x41	1	1	+	+	0	0	0	B <= B or A
Or A C	0x42	1	1	+	+	0	0	0	C <= C or A
Or A D	0x43	1	1	+	+	0	0	0	D <= D or A
Or B A	0x44	1	1	+	+	0	0	0	A <= A or B
Or B B	0x45	1	1	+	+	0	0	0	B <= B or B
Or B C	0x46	1	1	+	+	0	0	0	C <= C or B
Or B D	0x47	1	1	+	+	0	0	0	D <= D or B
Or C A	0x48	1	1	+	+	0	0	0	A <= A or C
Or C B	0x49	1	1	+	+	0	0	0	B <= B or C
Or C C	0x4a	1	1	+	+	0	0	0	C <= C or C
Or C D	0x4b	1	1	+	+	0	0	0	D <= D or C
Or D A	0x4c	1	1	+	+	0	0	0	A <= A or D
Or D B	0x4d	1	1	+	+	0	0	0	B <= B or D
Or D C	0x4e	1	1	+	+	0	0	0	C <= C or D
Or D D	0x4f	1	1	+	+	0	0	0	D <= D or D

4.22 Xor

This command xor's 2 registers

Xor "0101XXXX"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	(Registers)
Xor A A	0x50	1	1	+	+	+	0	0	A <= A xor A
Xor A B	0x51	1	1	+	+	+	0	0	B <= B xor A
Xor A C	0x52	1	1	+	+	+	0	0	C <= C xor A
Xor A D	0x53	1	1	+	+	+	0	0	D <= D xor A
Xor B A	0x54	1	1	+	+	+	0	0	A <= A xor B
Xor B B	0x55	1	1	+	+	+	0	0	B <= B xor B
Xor B C	0x56	1	1	+	+	+	0	0	C <= C xor B
Xor B D	0x57	1	1	+	+	+	0	0	D <= D xor B
Xor C A	0x58	1	1	+	+	+	0	0	A <= A xor C
Xor C B	0x59	1	1	+	+	+	0	0	B <= B xor C
Xor C C	0x5a	1	1	+	+	+	0	0	C <= C xor C
Xor C D	0x5b	1	1	+	+	+	0	0	D <= D xor C
Xor D A	0x5c	1	1	+	+	+	0	0	A <= A xor D
Xor D B	0x5d	1	1	+	+	+	0	0	B <= B xor D
Xor D C	0x5e	1	1	+	+	+	0	0	C <= C xor D
Xor D D	0x5f	1	1	+	+	+	0	0	D <= D xor D

4.23 Add - add without carry

This command adds 2 registers

Add "0110XXXX"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	(Registers)
Add A A	0x60	1	1	+	+	+	0	+	A <= A + A
Add A B	0x61	1	1	+	+	+	0	+	B <= B + A
Add A C	0x62	1	1	+	+	+	0	+	C <= C + A
Add A D	0x63	1	1	+	+	+	0	+	D <= D + A
Add B A	0x64	1	1	+	+	+	0	+	A <= A + B
Add B B	0x65	1	1	+	+	+	0	+	B <= B + B
Add B C	0x66	1	1	+	+	+	0	+	C <= C + B
Add B D	0x67	1	1	+	+	+	0	+	D <= D + B
Add C A	0x68	1	1	+	+	+	0	+	A <= A + C
Add C B	0x69	1	1	+	+	+	0	+	B <= B + C
Add C C	0x6a	1	1	+	+	+	0	+	C <= C + C
Add C D	0x6b	1	1	+	+	+	0	+	D <= D + C
Add D A	0x6c	1	1	+	+	+	0	+	A <= A + D
Add D B	0x6d	1	1	+	+	+	0	+	B <= B + D
Add D C	0x6e	1	1	+	+	+	0	+	C <= C + D
Add D D	0x6f	1	1	+	+	+	0	+	D <= D + D

4.24 Addc - add with carry

This command adds 2 registers + Carry

Addc "0111XXXX"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	(Registers)
Addc A A	0x70	1	1	+	+	+	0	+	A <= A + A + Carry
Addc A B	0x71	1	1	+	+	+	0	+	B <= B + A + Carry
Addc A C	0x72	1	1	+	+	+	0	+	C <= C + A + Carry
Addc A D	0x73	1	1	+	+	+	0	+	D <= D + A + Carry
Addc B A	0x74	1	1	+	+	+	0	+	A <= A + B + Carry
Addc B B	0x75	1	1	+	+	+	0	+	B <= B + B + Carry
Addc B C	0x76	1	1	+	+	+	0	+	C <= C + B + Carry
Addc B D	0x77	1	1	+	+	+	0	+	D <= D + B+ Carry
Addc C A	0x78	1	1	+	+	+	0	+	A <= A + C + Carry
Addc C B	0x79	1	1	+	+	+	0	+	B <= B + C + Carry
Addc C C	0x7a	1	1	+	+	+	0	+	C <= C + C + Carry
Addc C D	0x7b	1	1	+	+	+	0	+	D <= D + C + Carry
Addc D A	0x7c	1	1	+	+	+	0	+	A <= A + D + Carry
Addc D B	0x7d	1	1	+	+	+	0	+	B <= B + D + Carry
Addc D C	0x7e	1	1	+	+	+	0	+	C <= C + D + Carry
Addc D D	0x7f	1	1	+	+	+	0	+	D <= D + D + Carry

4.25 Move

This command moves the value of one register into another register

Move "1000XXXX"				Set Flags					Comment
Ex Instruction	OpCode	Bytes	Clocks	N	Z	O	U	C	(Registers)
Move A A	0x80	1	1	-	-	-	-	-	A <= A
Move A B	0x81	1	1	-	-	-	-	-	B <= A
Move A C	0x82	1	1	-	-	-	-	-	C <= A
Move A D	0x83	1	1	-	-	-	-	-	D <= A
Move B A	0x84	1	1	-	-	-	-	-	A <= B
Move B B	0x85	1	1	-	-	-	-	-	B <= B
Move B C	0x86	1	1	-	-	-	-	-	C <= B
Move B D	0x87	1	1	-	-	-	-	-	D <= B
Move C A	0x88	1	1	-	-	-	-	-	A <= C
Move C B	0x89	1	1	-	-	-	-	-	B <= C
Move C C	0x8a	1	1	-	-	-	-	-	C <= C
Move C D	0x8b	1	1	-	-	-	-	-	D <= C
Move D A	0x8c	1	1	-	-	-	-	-	A <= D
Move D B	0x8d	1	1	-	-	-	-	-	B <= D
Move D C	0x8e	1	1	-	-	-	-	-	C <= D
Move D D	0x8f	1	1	-	-	-	-	-	D <= D

References