Manuals & Books

Z80 CPU

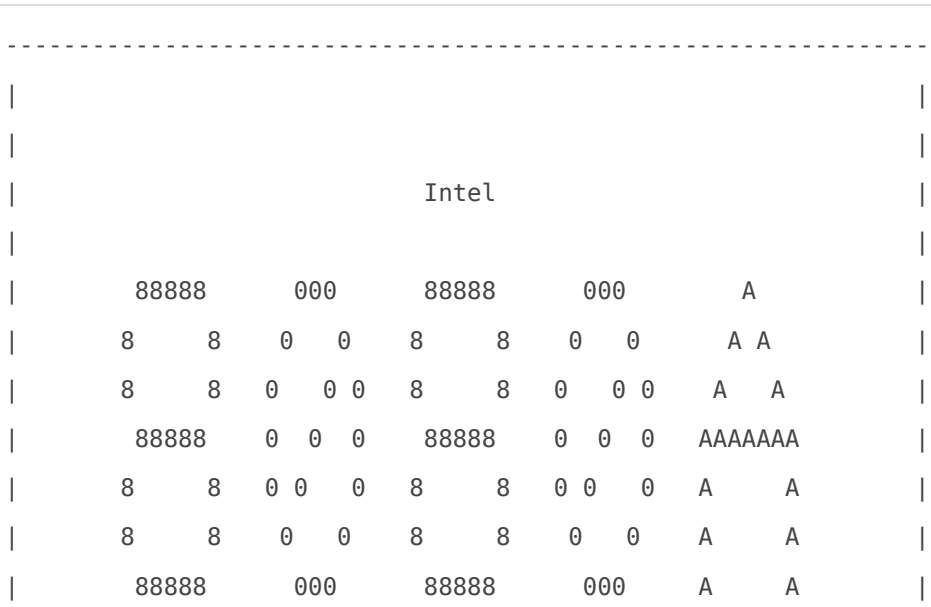
Hardware

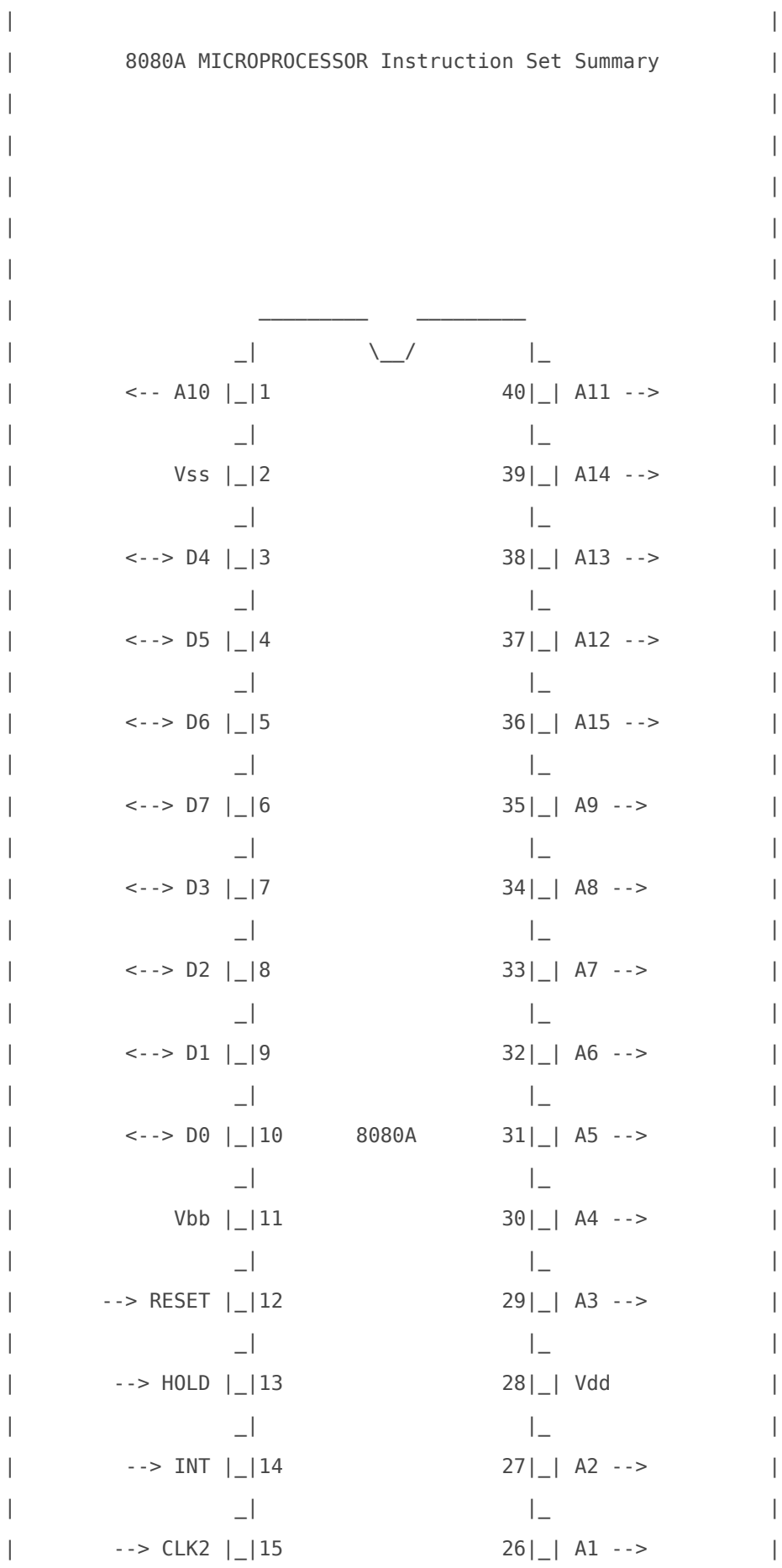
- User Manual
- CPU Peripherals

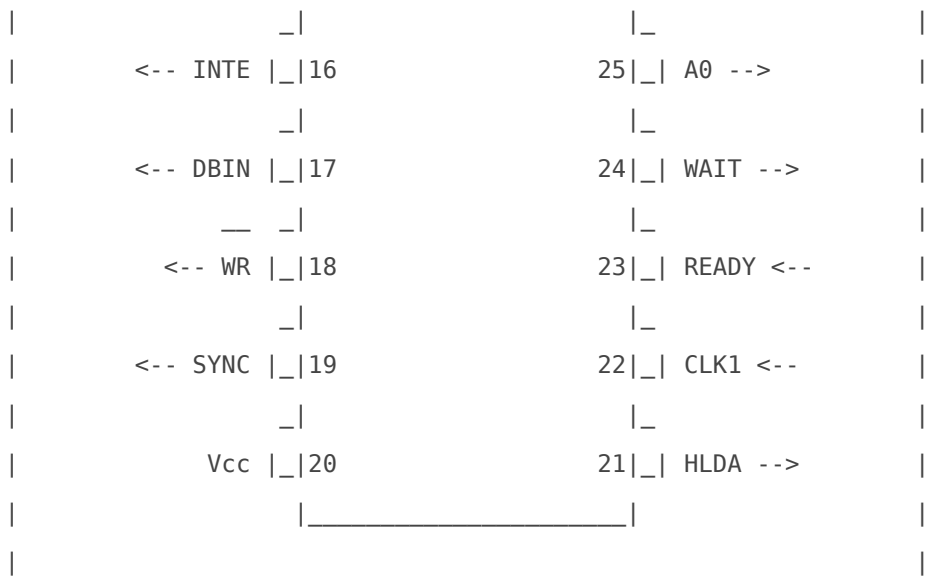
Software

- Z-80 Assembly Language Programming (and other books form archive.org)
- How to program the Z80
- Z80 CPU Microprocessor Instant Reference Card (alt)
- 8080 Programmers Manual
- 8085Ref
- 8080A MICROPROCESSOR Instruction Set Summary

8080A




8080A MICROPROCESSOR Instruction Set Summary
<-- A10
Vss
<--> D4
<--> D5
<--> D6
<--> D7
<--> D3
<--> D2
<--> D1
<--> D0
Vbb
--> RESET
--> HOLD
--> INT
--> CLK2
8080A
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
40
39
38
37
36
35
34
33
32
31
30
29
28
27
26
A11 -->
A14 -->
A13 -->
A12 -->
A15 -->
A9 -->
A8 -->
A7 -->
A6 -->
A5 -->
A4 -->
A3 -->
Vdd
A2 -->
A1 -->



Written by Jonathan Bowen
Programming Research Group
Oxford University Computing Laboratory
8-11 Keble Road
Oxford OX1 3QD
England

Tel +44-865-273840

Created May 1983
Updated April 1985
Issue 1.1

Mnemonic	Op	SZAPC	~s	Description	Notes
ACI n	CE	*****	7	Add with Carry Immediate	A=A+n+CY
ADC r	8F	*****	4	Add with Carry	A=A+r+CY(21X)
ADC M	8E	*****	7	Add with Carry to Memory	A=A+[HL]+CY
ADD r	87	*****	4	Add	A=A+r (20X)
ADD M	86	*****	7	Add to Memory	A=A+[HL]

ADI n	C6	*****	7	Add Immediate	A=A+n
ANA r	A7	****0	4	AND Accumulator	A=A&r (24X)
ANA M	A6	****0	7	AND Accumulator and Memory	A=A&[HL]
ANI n	E6	**0*0	7	AND Immediate	A=A&n
CALL a	CD	-----	17	Call unconditional	-[SP]=PC,PC=a
CC a	DC	-----	11	Call on Carry	If CY=1(17~s)
CM a	FC	-----	11	Call on Minus	If S=1 (17~s)
CMA	2F	-----	4	Complement Accumulator	A=~A
CMC	3F	----*	4	Complement Carry	CY=~CY
CMP r	BF	*****	4	Compare	A-r (27X)
CMP M	BF	*****	7	Compare with Memory	A-[HL]
CNC a	D4	-----	11	Call on No Carry	If CY=0(17~s)
CNZ a	C4	-----	11	Call on No Zero	If Z=0 (17~s)
CP a	F4	-----	11	Call on Plus	If S=0 (17~s)
CPE a	EC	-----	11	Call on Parity Even	If P=1 (17~s)
CPI n	FE	*****	7	Compare Immediate	A-n
CPO a	E4	-----	11	Call on Parity Odd	If P=0 (17~s)
CZ a	CC	-----	11	Call on Zero	If Z=1 (17~s)
DAA	27	*****	4	Decimal Adjust Accumulator	A=BCD format
DAD B	09	----*	10	Double Add BC to HL	HL=HL+BC
DAD D	19	----*	10	Double Add DE to HL	HL=HL+DE
DAD H	29	----*	10	Double Add HL to HL	HL=HL+HL
DAD SP	39	----*	10	Double Add SP to HL	HL=HL+SP
DCR r	3D	****-	5	Decrement	r=r-1 (0X5)
DCR M	35	****-	10	Decrement Memory	[HL]=[HL]-1
DCX B	0B	-----	5	Decrement BC	BC=BC-1
DCX D	1B	-----	5	Decrement DE	DE=DE-1
DCX H	2B	-----	5	Decrement HL	HL=HL-1
DCX SP	3B	-----	5	Decrement Stack Pointer	SP=SP-1
DI	F3	-----	4	Disable Interrupts	
EI	FB	-----	4	Enable Interrupts	
HLT	76	-----	7	Halt	
IN p	DB	-----	10	Input	A=[p]
INR r	3C	****-	5	Increment	r=r+1 (0X4)
INR M	3C	****-	10	Increment Memory	[HL]=[HL]+1
INX B	03	-----	5	Increment BC	BC=BC+1
INX D	13	-----	5	Increment DE	DE=DE+1
INX H	23	-----	5	Increment HL	HL=HL+1

```
|INX SP   |33|-----| 5|Increment Stack Pointer   |SP=SP+1      |
|JMP a    |C3|-----|10|Jump unconditional        |PC=a         |
|JC a     |DA|-----|10|Jump on Carry             |If CY=1(10~s)|
|JM a     |FA|-----|10|Jump on Minus             |If S=1 (10~s)|
|JNC a    |D2|-----|10|Jump on No Carry          |If CY=0(10~s)|
|JNZ a    |C2|-----|10|Jump on No Zero           |If Z=0 (10~s)|
|JP a     |F2|-----|10|Jump on Plus              |If S=0 (10~s)|
|JPE a    |EA|-----|10|Jump on Parity Even       |If P=1 (10~s)|
|JPO a    |E2|-----|10|Jump on Parity Odd        |If P=0 (10~s)|
|JZ a     |CA|-----|10|Jump on Zero              |If Z=1 (10~s)|
|LDA a    |3A|-----|13|Load Accumulator direct   |A=[a]        |
|LDAX B   |0A|-----| 7|Load Accumulator indirect |A=[BC]       |
|LDAX D   |1A|-----| 7|Load Accumulator indirect |A=[DE]       |
|LHLD a   |2A|-----|16|Load HL Direct            |HL=[a]       |
|LXI B,nn |01|-----|10|Load Immediate BC         |BC=nn        |
|LXI D,nn |11|-----|10|Load Immediate DE         |DE=nn        |
|LXI H,nn |21|-----|10|Load Immediate HL         |HL=nn        |
|LXI SP,nn|31|-----|10|Load Immediate Stack Ptr  |SP=nn        |
|MOV r1,r2|7F|-----| 5|Move register to register |r1=r2   (1XX)|
|MOV M,r  |77|-----| 7|Move register to Memory   |[HL]=r  (16X)|
|MOV r,M  |7E|-----| 7|Move Memory to register   |r=[HL]  (1X6)|
|MVI r,n  |3E|-----| 7|Move Immediate            |r=n     (0X6)|
|MVI M,n  |36|-----|10|Move Immediate to Memory  |[HL]=n       |
|NOP      |00|-----| 4|No Operation              |             |
|ORA r    |B7|**0*0| 4|Inclusive OR Accumulator  |A=Avr   (26X)|
|ORA M    |B6|**0*0| 7|Inclusive OR Accumulator  |A=Av[HL]     |
|ORI n    |F6|**0*0| 7|Inclusive OR Immediate    |A=Avn        |
|OUT p    |D3|-----|10|Output                    |[p]=A        |
|PCHL     |E9|-----| 5|Jump HL indirect          |PC=[HL]      |
|POP B    |C1|-----|10|Pop BC                    |BC=[SP]+     |
|POP D    |D1|-----|10|Pop DE                    |DE=[SP]+     |
|POP H    |E1|-----|10|Pop HL                    |HL=[SP]+     |
|POP PSW  |F1|-----|10|Pop Processor Status Word |{PSW,A}=[SP]+|
-------------------------------------------------------------
-------------------------------------------------------------
|Mnemonic |Op|SZAPC|~s|Description               |Notes        |
|---------+--+-----+--+--------------------------+-------------|
|PUSH B   |C5|-----|11|Push BC                   |-[SP]=BC     |
```

```
|PUSH D   |D5|-----|11|Push DE                   |-[SP]=DE     |
|PUSH H   |E5|-----|11|Push HL                   |-[SP]=HL     |
|PUSH PSW |F5|-----|11|Push Processor Status Word|-[SP]={PSW,A}|
|RAL      |17|----*| 4|Rotate Accumulator Left   |A={CY,A}<-   |
|RAR      |1F|----*| 4|Rotate Accumulator Righ   |A=->{CY,A}   |
|RET      |C9|-----|10|Return                    |PC=[SP]+     |
|RC       |D8|-----| 5|Return on Carry           |If CY=1(11~s)|
|RM       |F8|-----| 5|Return on Minus           |If S=1 (11~s)|
|RNC      |D0|-----| 5|Return on No Carry        |If CY=0(11~s)|
|RNZ      |C0|-----| 5|Return on No Zero         |If Z=0 (11~s)|
|RP       |F0|-----| 5|Return on Plus            |If S=0 (11~s)|
|RPE      |E8|-----| 5|Return on Parity Even     |If P=1 (11~s)|
|RPO      |E0|-----| 5|Return on Parity Odd      |If P=0 (11~s)|
|RZ       |C8|-----| 5|Return on Zero            |If Z=1 (11~s)|
|RLC      |07|----*| 4|Rotate Left Circular      |A=A<-        |
|RRC      |0F|----*| 4|Rotate Right Circular     |A=->A        |
|RST z    |C7|-----|11|Restart              (3X7)|-[SP]=PC,PC=z|
|SBB r    |9F|*****| 4|Subtract with Borrow      |A=A-r-CY(23X)|
|SBB M    |9E|*****| 7|Subtract with Borrow      |A=A-[HL]-CY  |
|SBI n    |DE|*****| 7|Subtract with Borrow Immed|A=A-n-CY     |
|SHLD a   |22|-----|16|Store HL Direct           |[a]=HL       |
|SPHL     |F9|-----| 5|Move HL to SP             |SP=HL        |
|STA a    |32|-----|13|Store Accumulator         |[a]=A        |
|STAX B   |02|-----| 7|Store Accumulator indirect|[BC]=A       |
|STAX D   |12|-----| 7|Store Accumulator indirect|[DE]=A       |
|STC      |37|----1| 4|Set Carry                 |CY=1         |
|SUB r    |97|*****| 4|Subtract                  |A=A-r   (22X)|
|SUB M    |96|*****| 7|Subtract Memory           |A=A-[HL]     |
|SUI n    |D6|*****| 7|Subtract Immediate        |A=A-n        |
|XCHG     |EB|-----| 4|Exchange HL with DE       |HL<->DE      |
|XRA r    |AF|**0*0| 4|Exclusive OR Accumulator  |A=Axr   (25X)|
|XRA M    |AE|**0*0| 7|Exclusive OR Accumulator  |A=Ax[HL]     |
|XRI n    |EE|**0*0| 7|Exclusive OR Immediate    |A=Axn        |
|XTHL     |E3|-----|18|Exchange stack Top with HL|[SP]<->HL    |
|-----------+-----+--+----------------------------------------|
| PSW       |-*01 |  |Flag unaffected/affected/reset/set      |
| S         |S    |  |Sign (Bit 7)                            |
| Z         | Z   |  |Zero (Bit 6)                            |
```

```
| AC         |  A  |  |Auxilary Carry (Bit 4)                  |
| P          |   P |  |Parity (Bit 2)                          |
| CY         |    C|  |Carry (Bit 0)                           |
|--------------------+----------------------------------------|
| a p                |Direct addressing                       |
| M z                |Register indirect addressing            |
| n nn               |Immediate addressing                    |
| r                  |Register addressing                     |
|--------------------+----------------------------------------|
|DB n(,n)            |Define Byte(s)                          |
|DB 'string'         |Define Byte ASCII character string      |
|DS nn               |Define Storage Block                    |
|DW nn(,nn)          |Define Word(s)                          |
|--------------------+----------------------------------------|
| A B C D E H L      |Registers (8-bit)                       |
| BC DE HL           |Register pairs (16-bit)                 |
| PC                 |Program Counter register (16-bit)       |
| PSW                |Processor Status Word (8-bit)           |
| SP                 |Stack Pointer register (16-bit)         |
|--------------------+----------------------------------------|
| a                  |16-bit address quantity (0 to 65535)    |
| n                  |8-bit data quantity (0 to 255)          |
| nn                 |16-bit data quantity (0 to 65535)       |
| p                  |8-bit I/O port number (0 to 255)        |
| r                  |Register (X=B,C,D,E,H,L,M,A)            |
| z                  |Vector (X=0H,8H,10H,18H,20H,28H,30H,38H)|
|--------------------+----------------------------------------|
| +  -               |Arithmetic addition/subtraction         |
| &  ~               |Logical AND/NOT                         |
| v  x               |Logical inclusive/exclusive OR          |
| <-  ->             |Rotate left/right                       |
| <->                |Exchange                                |
| [ ]                |Indirect addressing                     |
| [ ]+  -[ ]         |Indirect addr. auto-increment/decrement |
| { }                |Combination of operands                 |
| ( X )              |Octal op code where X is a 3-bit code   |
| If ( ~s)           |Number of cycles if condition true      |
-------------------------------------------------------------
```

CP/M

- CP/M Internals
- CPM 2.0 Interface Guide

Kits

- RC2014: https://rc2014.co.uk/
- Z80-MBC2: https://github.com/SuperFabius/Z80-MBC2

Revision #16
Created 2023-12-09 12:17:26 GMT by hxd
Updated 2024-11-20 19:59:07 GMT by hxd