

**Z80**  
**Instruction**  
**Set**

Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
LD A, r	1	78+rb	4	-----	A=r
LD A, N	2	3E XX	7	-----	A=N
LD A, (BC)	1	0A	7	-----	A=[BC]
LD A, (DE)	1	1A	7	-----	A=[DE]
LD A, (HL)	1	7E	7	-----	A=[HL]
LD A, (IX+n)	3	DD 7E XX	19	-----	A=[IX+n]
LD A, (IY+n)	3	FD 7E XX	19	-----	A=[IY+n]
LD A, (NN)	3	3A XX XX	13	-----	A=[NN]
LD B, N	2	06 XX	7	-----	B=N
LD B, r	1	40+rb	4	-----	B=r
LD B, (HL)	1	46	7	-----	B=[HL]
LD B, (IX+n)	3	DD 46 XX	19	-----	B=[IX+n]
LD B, (IY+n)	3	FD 46 XX	19	-----	B=[IY+n]
LD C, N	2	0E XX	7	-----	C=N
LD C, r	1	48+rb	4	-----	C=r
LD C, (HL)	1	4E	7	-----	C=[HL]
LD C, (IX+n)	3	DD 4E XX	19	-----	C=[IX+n]
LD C, (IY+n)	3	FD 4E XX	19	-----	C=[IY+n]
LD D, N	2	16 XX	7	-----	D=N
LD D, r	1	50+rb	4	-----	D=r
LD D, (HL)	1	56	7	-----	D=[HL]
LD D, (IX+n)	3	DD 56 XX	19	-----	D=[IX+n]
LD D, (IY+n)	3	FD 56 XX	19	-----	D=[IY+n]
LD E, N	2	1E XX	7	-----	E=N
LD E, r	1	58+rb	4	-----	E=r
LD E, (HL)	1	5E	7	-----	E=[HL]
LD E, (IX+n)	3	DD 5E XX	19	-----	E=[IX+n]
LD E, (IY+n)	3	FD 5E XX	19	-----	E=[IY+n]
LD H, N	2	26 XX	7	-----	H=N
LD H, r	1	60+rb	4	-----	H=r
LD H, (HL)	1	66	7	-----	H=[HL]
LD H, (IX+n)	3	DD 66 XX	19	-----	H=[IX+n]
LD H, (IY+n)	3	FD 66 XX	19	-----	H=[IY+n]
LD L, N	2	2E XX	7	-----	L=N
LD L, (HL)	1	6E	7	-----	L=[HL]
LD L, (IX+n)	3	DD 6E XX	19	-----	L=[IX+n]
LD L, (IY+n)	3	FD 6E XX	19	-----	L=[IY+n]
LD L, r	1	68+rb	4	-----	L=r

Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
LD (NN),A	3	32 XX XX	13	-----	[NN]=A
LD (BC),A	1	02	7	-----	[BC]=A
LD (DE),A	1	12	7	-----	[DE]=A
LD (HL),N	2	36 XX	10	-----	[HL]=N
LD (HL),r	1	70+rb	7	-----	[HL]=r
LD (IX+n),r	3	DD 70+rb XX	19	-----	[IX+n]=r
LD (IX+n),N	4	DD 36 XX XX	19	-----	[IX+n]=N
LD (IY+n),r	3	FD 70+rb XX	19	-----	[IY+n]=r
LD (IY+n),N	4	FD 36 XX XX	19	-----	[IY+n]=N
LD BC,NN	3	01 XX XX	10	-----	BC=NN
LD DE,NN	3	11 XX XX	10	-----	DE=NN
LD HL,NN	3	21 XX XX	10	-----	HL=NN
LD IX,NN	4	DD 21 XX XX	14	-----	IX=NN
LD IY,NN	4	FD 21 XX XX	14	-----	IY=NN
LD SP,HL	1	F9	6	-----	SP=HL
LD SP,IX	2	DD F9	10	-----	SP=IX
LD SP,IY	2	FD F9	10	-----	SP=IY
LD SP,NN	3	31 XX XX	10	-----	SP=NN
LD BC,(NN)	4	ED 4B XX XX	20	-----	C=[NN],B=[NN+1]
LD DE,(NN)	4	ED 5B XX XX	20	-----	E=[NN],D=[NN+1]
LD HL,(NN)	3	2A XX XX	16	-----	L=[NN],H=[NN+1]
LD HL,(NN)	4	ED 6B XX XX	20	-----	L=[NN],H=[NN+1]
LD IX,(NN)	4	DD 2A XX XX	20	-----	IX=[NN,NN+1]
LD IY,(NN)	4	FD 2A XX XX	20	-----	IY=[NN,NN+1]
LD SP,(NN)	4	ED 7B XX XX	20	-----	SP=[NN,NN+1]
LD (NN),BC	4	ED 43 XX XX	20	-----	[NN]=C, (NN+1)=B
LD (NN),DE	4	ED 53 XX XX	20	-----	[NN]=E, (NN+1)=D
LD (NN),HL	3	22 XX XX	16	-----	[NN]=L, (NN+1)=H
LD (NN),IX	4	DD 22 XX XX	20	-----	[NN,NN+1]=IX
LD (NN),IY	4	FD 22 XX XX	20	-----	[NN,NN+1]=IY
LD (NN),SP	4	ED 73 XX XX	20	-----	[NN,NN+1]=SP

r means register : B,C,D,E,H,L,A. Add to last byte of OP-Code:

Reg	regbits
B	0
C	1
D	2
E	3
H	4
L	5
A	7

On LD (IX+n),r and LD (IY+n),r add it to the byte before the last.

Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
PUSH AF	1	F5	11	-----	-SP, [SP]=A, -SP, [SP]=F
PUSH BC	1	C5	11	-----	-SP, [SP]=B, -SP, [SP]=C
PUSH DE	1	D5	11	-----	-SP, [SP]=D, -SP, [SP]=E
PUSH HL	1	E5	11	-----	-SP, [SP]=H, -SP, [SP]=L
PUSH IX	2	DD E5	15	-----	-SP, -SP, [SP, SP+1]=IX
PUSH IY	2	FD E5	15	-----	-SP, -SP, [SP, SP+1]=IY
POP AF	1	F1	10	*****	F=[SP], SP+, A=[SP], SP+
POP BC	1	C1	10	-----	C=[SP], SP+, B=[SP], SP+
POP DE	1	D1	10	-----	E=[SP], SP+, D=[SP], SP+
POP HL	1	E1	10	-----	L=[SP], SP+, H=[SP], SP+
POP IX	2	DD E1	14	-----	IX=[SP, SP+1], SP+, SP+
POP IY	2	FD E1	14	-----	IY=[SP, SP+1], SP+, SP+
EX AF, AF'	1	08	4	*****	AF<->AF'
EX DE, HL	1	EB	4	-----	DE<->HL
EXX	1	D9	4	-----	BC<->BC', DE<->DE', HL<->HL'
EX (SP), HL	1	E3	19	-----	[SP]<->HL
EX (SP), IX	2	DD E3	23	-----	[SP]<->IX
EX (SP), IY	2	FD E3	23	-----	[SP]<->IY
LDD	2	ED A8	16	--0*0-	[DE]=[HL], HL--=1, DE--=1, BC--=1
LDDR	2	ED B8	21/16	--000-	LDD until BC=0
LDI	2	ED A0	16	--0*0-	[DE]=[HL], HL+=1, DE+=1, BC=-1
LDIR	2	ED B0	21/16	--000-	LDI until BC=0

In memory, the low byte of a 16-bit value comes first, then the high byte.  
e.g. after LD (addr),BC in memory, [addr] = C and [addr+1] = B

EX instructions cannot work on IX, IY registers.

Use XOR A as a faster alternative to LD A,0, and also clear flags

Note the faster forms of LD HL, (NN) and LD (NN), HL

Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
INC (HL)	1	34	11	***V0-	[HL]=[HL]+1
INC (IX+n)	3	DD 34 XX	23	***V0-	[IX+n]=[IX+n]+1
INC (IY+n)	3	FD 34 XX	23	***V0-	[IY+n]=[IY+n]+1
INC A	1	3C	4	***V0-	A=A+1
INC B	1	04	4	***V0-	B=B+1
INC BC	1	03	6	-----	BC=BC+1
INC C	1	0C	4	***V0-	C=C+1
INC D	1	14	4	***V0-	D=D+1
INC DE	1	13	6	-----	DE=DE+1
INC E	1	1C	4	***V0-	E=E+1
INC H	1	24	4	***V0-	H=H+1
INC HL	1	23	6	-----	HL=HL+1
INC IX	2	DD 23	10	-----	IX=IX+1
INC IY	2	FD 23	10	-----	IY=IY+1
INC L	1	2C	4	***V0-	L=L+1
INC SP	1	33	6	-----	SP=SP+1
DEC (HL)	1	35	11	***V1-	[HL]=[HL] - 1
DEC (IX+n)	3	DD 35 XX	23	***V1-	[IX+n]=[IX+n] - 1
DEC (IY+n)	3	FD 35 XX	23	***V1-	[IY+n]=[IY+n] - 1
DEC A	1	3D	4	***V1-	A=A - 1
DEC B	1	05	4	***V1-	B=B - 1
DEC BC	1	0B	6	-----	BC=BC - 1
DEC C	1	0D	4	***V1-	C=C - 1
DEC D	1	15	4	***V1-	D=D - 1
DEC DE	1	1B	6	-----	DE=DE - 1
DEC E	1	1D	4	***V1-	E=E - 1
DEC H	1	25	4	***V1-	H=H - 1
DEC HL	1	2B	6	-----	HL=HL - 1
DEC IX	2	DD 2B	10	-----	IX=IX - 1
DEC IY	2	FD 2B	10	-----	IY=IY - 1
DEC L	2	2D	4	***V1-	L=L - 1
DEC SP	1	3B	6	-----	SP=SP - 1
DJNZ n	2	10 XX	13/8	-----	B=B-1, if B != 0 then PC+=n

There are undocumented instructions that operate on the high and low bytes of IX and IY, equivalent to operations on H and L.

e.g. LD A,IXL LD IXH,A LD IYL,n LD B,IYH

Also INC, DEC, ADD, ADC, SUB, SBC, AND, OR, XOR and CP wherever H or L could be used.

Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
NEG	2	ED 44	8	***V1*	A=-A
CPL	1	2F	4	--1-1-	A=~A
SCF	1	37	4	--0-01	CY=1
CCF	1	3F	4	--*-0*	CY=~CY
DAA	1	27	4	***P-*	A=adjust result to BCD-format
ADD A, (HL)	1	86	7	***V0*	A=A+[HL]
ADD A, (IX+n)	3	DD 86 XX	19	***V0*	A=A+[IX+n]
ADD A, (IY+n)	3	FD 86 XX	19	***V0*	A=A+[IY+n]
ADD A, r	1	80+rb	4	***V0*	A=A+r
ADD A, N	2	C6 XX	7	***V0*	A=A+N
ADD HL, BC	1	09	11	--*-0*	HL=HL+BC
ADD HL, DE	1	19	11	--*-0*	HL=HL+DE
ADD HL, HL	1	29	11	--*-0*	HL=HL+HL
ADD HL, SP	1	39	11	--*-0*	HL=HL+SP
ADD IX, BC	2	DD 09	15	--*-0*	IX=IX+BC
ADD IX, DE	2	DD 19	15	--*-0*	IX=IX+DE
ADD IX, IX	2	DD 29	15	--*-0*	IX=IX+IX
ADD IX, SP	2	DD 39	15	--*-0*	IX=IX+SP
ADD IY, BC	2	FD 09	15	--*-0*	IY=IY+BC
ADD IY, DE	2	FD 19	15	--*-0*	IY=IY+DE
ADD IY, IY	2	FD 29	15	--*-0*	IY=IY+IY
ADD IY, SP	2	FD 39	15	--*-0*	IY=IY+SP
ADC A, (HL)	1	8E	7	***V0*	A=A+[HL]+CY
ADC A, (IX+n)	3	DD 8E XX	19	***V0*	A=A+[IX+n]+CY
ADC A, (IY+n)	3	FD 8E XX	19	***V0*	A=A+[IY+n]+CY
ADC A, r	1	88+rb	4	***V0*	A=A+r+CY
ADC A, N	2	CE XX	7	***V0*	A=A+N+CY
ADC HL, BC	2	ED 4A	15	***V0*	HL=HL+BC+CY
ADC HL, DE	2	ED 5A	15	***V0*	HL=HL+DE+CY
ADC HL, HL	2	ED 6A	15	***V0*	HL=HL+HL+CY
ADC HL, SP	2	ED 7A	15	***V0*	HL=HL+SP+CY

Use NEG A to Mathematically Negate A register

Use CPL to Binary Invert A register, equivalent to XOR \$FF

Use CPL to calculate -A-1 as it is faster than NEG A

Use SCF to Set Carry Flag

Use AND A to Clear Carry Flag e.g. before SBC HL,BC

Use CCF to Invert Carry Flag

Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
SUB (HL)	1	96	7	***V1*	A=A- [HL]
SUB (IX+n)	3	DD 96 XX	19	***V1*	A=A- [IX+n]
SUB (IY+n)	3	FD 96 XX	19	***V1*	A=A- [IY+n]
SUB r	1	90+rb	4	***V1*	A=A- r
SUB N	2	D6 XX	7	***V1*	A=A-N
SBC (HL)	1	9E	7	***V1*	A=A- [HL] -CY
SBC A, (IX+n)	3	DD 9E XX	19	***V1*	A=A- [IX+n] -CY
SBC A, (IY+n)	3	FD 9E XX	19	***V1*	A=A- [IY+n] -CY
SBC A,N	2	DE XX	7	***V1*	A=A-N-CY
SBC r	1	98+rb	4	***V1*	A=A- r -CY
SBC HL,BC	2	ED 42	15	***V1*	HL=HL -BC -CY
SBC HL,DE	2	ED 52	15	***V1*	HL=HL -DE -CY
SBC HL,HL	2	ED 62	15	***V1*	HL=HL -HL -CY
SBC HL,SP	2	ED 72	15	***V1*	HL=HL -SP -CY
AND (HL)	1	A6	7	***P00	A=A&[HL]
AND (IX+n)	3	DD A6 XX	19	***P00	A=A&[IX+n]
AND (IY+n)	3	FD A6 XX	19	***P00	A=A&[IY+n]
AND r	1	A0+rb	4	***P00	A=A&r
AND N	2	E6 XX	7	***P00	A=A&N
OR (HL)	1	B6	7	***P00	A=Av[HL]
OR (IX+n)	3	DD B6 XX	19	***P00	A=Av[IX+n]
OR (IY+n)	3	FD B6 XX	19	***P00	A=Av[IY+n]
OR r	1	B0+rb	4	***P00	A=Avr
OR N	2	F6 XX	7	***P00	A=AvN
XOR (HL)	1	AE	7	***P00	A=Ax[HL]
XOR (IX+n)	3	DD AE XX	19	***P00	A=Ax[IX+n]
XOR (IY+n)	3	FD AE XX	19	***P00	A=Ax[IY+n]
XOR r	1	A8+rb	4	***P00	A=Axr
XOR N	2	EE XX	7	***P00	A=AxN

Use ADD HL,HL to double HL or shift it one bit left.

Use CPL instead of XOR \$FF

Use SBC A,0 to sign-extend Carry Flag into A, giving either \$00 or \$FF

Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
CP (HL)	1	BE	7	***V1*	A-[HL]
CP (IX+n)	3	DD BE XX	19	***V1*	A-[IX+n]
CP (IY+n)	3	FD BE XX	19	***V1*	A-[IY+n]
CP r	1	B8+rb	4	***V1*	A-r
CP N	2	FE XX	7	***V1*	A-N
CPD	2	ED A9	16	****1-	A-[HL],HL=HL-1,BC=BC-1
CPDR	2	ED B9	21/16	****1-	CPD until A=[HL] or BC=0
CPI	2	ED A1	16	****1-	A-[HL],HL=HL+1,BC=BC-1
CPIR	2	ED B1	21/16	****1-	CPI until A=[HL] or BC=0
JR n	2	18 XX	12	-----	PC=PC+n
JR C,n	2	38 XX	12/7	-----	If CY then PC=PC+n
JR NC,n	2	30 XX	12/7	-----	If !CY then PC=PC+n
JR NZ,n	2	20 XX	12/7	-----	If !Z then PC=PC+n
JR Z,n	2	28 XX	12/7	-----	If Z then PC=PC+n
JP NN	3	C3 XX XX	10	-----	PC=NN
JP (HL)	1	E9	4	-----	PC=HL
JP (IX)	2	DD E9	8	-----	PC=IX
JP (IY)	2	FD E9	8	-----	PC=IY
JP C,NN	3	DA XX XX	10/10	-----	If CY then PC=NN
JP M,NN	3	FA XX XX	10/10	-----	If S then PC=NN
JP NC,NN	3	D2 XX XX	10/10	-----	If !CY then PC=NN
JP NZ,NN	3	C2 XX XX	10/10	-----	If !Z then PC=NN
JP P,NN	3	F2 XX XX	10/10	-----	If !S then PC=NN
JP PE,NN	3	EA XX XX	10/10	-----	If P then PC=NN
JP PO,NN	3	E2 XX XX	10/10	-----	If !P then PC=NN
JP Z,NN	3	CA XX XX	10/10	-----	If Z then PC=NN

The flag field contains one of the following:

- Flag unaffected
- \* Flag affected
- 0 Flag reset
- 1 Flag set
- ? Unknown
- P Parity-Flag used as Parity
- V Parity-Flag used as Overflow-Flag

After a CP N instruction (equivalent flags to a SUB N instruction):

- Z Zero Flag means  $A == N$
- C Carry Flag means  $A < N$  or  $N > A$
- NC !Carry means  $A \geq N$  or  $N \leq A$

Where there are two numbers given for Clock, the highest is when the jump is taken.



Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
CALL C,NN	3	DC XX XX	17/10	-----	If CY then [SP-=2]=PC,PC=NN
CALL M,NN	3	FC XX XX	17/10	-----	If S then [SP-=2]=PC,PC=NN
CALL NC,NN	3	D4 XX XX	17/10	-----	If !CY then [SP-=2]=PC,PC=NN
CALL NN	3	CD XX XX	17	-----	SP-=2, [SP+1, SP]=PC,PC=NN
CALL NZ,NN	3	C4 XX XX	17/10	-----	If !Z then [SP-=2]=PC,PC=NN
CALL P,NN	3	F4 XX XX	17/10	-----	If !S then [SP-=2]=PC,PC=NN
CALL PE,NN	3	EC XX XX	17/10	-----	If P then [SP-=2]=PC,PC=NN
CALL PO,NN	3	E4 XX XX	17/10	-----	If !P then [SP-=2]=PC,PC=NN
CALL Z,NN	3	CC XX XX	17/10	-----	If Z then [SP-=2]=PC,PC=NN
RET	1	C9	10	-----	PC=[SP, SP+1], SP+, SP+
RET C	1	D8	11/5	-----	If CY then PC=[SP, SP+1], SP+=2
RET M	1	F8	11/5	-----	If S then PC=[SP, SP+1], SP+=2
RET NC	1	D0	11/5	-----	If !CY then PC=[SP, SP+1], SP+=2
RET NZ	1	C0	11/5	-----	If !Z then PC=[SP, SP+1], SP+=2
RET P	1	F0	11/5	-----	If !S then PC=[SP, SP+1], SP+=2
RET PE	1	E8	11/5	-----	If P then PC=[SP, SP+1], SP+=2
RET PO	1	E0	11/5	-----	If !P then PC=[SP, SP+1], SP+=2
RET Z	1	C8	11/5	-----	If Z then PC=[SP, SP+1], SP+=2
RETI	2	ED 4D	14	-----	PC=[SP, SP+1], SP+, SP+
RETN	2	ED 45	14	-----	PC=[SP, SP+1], SP+, SP+
RST 0	1	C7	11	-----	-SP, -SP, [SP+1, SP]=PC, PC=00
RST 8H	1	CF	11	-----	-SP, -SP, [SP+1, SP]=PC, PC=08
RST 10H	1	D7	11	-----	-SP, -SP, [SP+1, SP]=PC, PC=10
RST 18H	1	DF	11	-----	-SP, -SP, [SP+1, SP]=PC, PC=18
RST 20H	1	E7	11	-----	-SP, -SP, [SP+1, SP]=PC, PC=20
RST 28H	1	EF	11	-----	-SP, -SP, [SP+1, SP]=PC, PC=28
RST 30H	1	F7	11	-----	-SP, -SP, [SP+1, SP]=PC, PC=30
RST 38H	1	FF	11	-----	-SP, -SP, [SP+1, SP]=PC, PC=38

The flag register contains the following bits:

7	Sign Flag, 1=Minus, 0=Plus	JP M , JP P
6	Zero Flag, 1 if result was zero	JP Z , JP NZ
-	Bit 5 Not Used	
4	Half-Carry Flag, 1 if Bit 3 carried	
-	Bit 3 Not Used	
2	Parity 1=Even, 0=Odd, or Overflow Flag	JP PE , JP PO
1	Add / Subtract Flag	
0	Carry Flag	JP C , JP NC , RR, RL

Note that JP (IX) and JP (IY) do not include a '+n' offset.

RST is equivalent to a CALL, but to fixed addresses.

Use OR A instead of CP 0

Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
RLCA	1	07	4	--0-0*	A={A}<<
RLC r	2	CB 00+rb	8	**0P0*	r={r}<<
RLC (HL)	2	CB 06	15	**0P0*	[HL]={ [HL] }<<
RLC (IX+n)	4	DD CB XX 06	23	**0P0*	[IX+n]={ [IX+n] }<<
RLC (IY+n)	4	FD CB XX 06	23	**0P0*	[IY+n]={ [IY+n] }<<
RRCA	1	0F	4	--0-0*	A=>>{A}
RRC r	2	CB 08+rb	8	**0P0*	r=>>{r}
RRC (HL)	2	CB 0E	15	**0P0*	[HL]=>>{ [HL] }
RRC (IX+n)	4	DD CB XX 0E	23	**0P0*	[IX+n]=>>{ [IX+n] }
RRC (IY+n)	4	FD CB XX 0E	23	**0P0*	[IY+n]=>>{ [IY+n] }
RLA	1	17	4	--0-0*	A={CY, A}<<CY
RL r	2	CB 10+rb	8	**0P0*	r={CY, r}<<CY
RL (HL)	2	CB 16	15	**0P0*	[HL]={CY, [HL]}<<CY
RL (IX+n)	4	DD CB XX 16	23	**0P0*	[IX+n]={CY, [IX+n]}<<CY
RL (IY+n)	4	FD CB XX 16	23	**0P0*	[IY+n]={CY, [IY+n]}<<CY
RRA	1	1F	4	--0-0*	A=CY>>{CY, A}
RR r	2	CB 18+rb	8	**0P0*	r=CY>>{CY, r}
RR (HL)	2	CB 1E	15	**0P0*	[HL]=CY>>{CY, [HL]}
RR (IX+n)	4	DD CB XX 1E	23	**0P0*	[IX+n]=CY>>{CY, [IX+n]}
RR (IY+n)	4	FD CB XX 1E	23	**0P0*	[IY+n]=CY>>{CY, [IY+n]}
SLA r	2	CB 20+rb	8	**0P0*	r=r*2
SLA (HL)	2	CB 26	15	**0P0*	[HL]=[HL]*2
SLA (IX+n)	4	DD CB XX 26	23	**0P0*	[IX+n]=[IX+n]*2
SLA (IY+n)	4	FD CB XX 26	23	**0P0*	[IY+n]=[IY+n]*2
SLL r	2	CB 30+rb	8	**0P0*	r=r*2+1
SLL (HL)	2	CB 36	15	**0P0*	[HL]=[HL]*2+1
SLL (IX+n)	4	DD CB XX 36	23	**0P0*	[IX+n]=[IX+n]*2+1
SLL (IY+n)	4	FD CB XX 36	23	**0P0*	[IY+n]=[IY+n]*2+1
SRA r	2	CB 28+rb	8	**0P0*	r=(signed) r/2
SRA (HL)	2	CB 2E	15	**0P0*	[HL]=(signed) [HL]/2
SRA (IX+n)	4	DD CB XX 2E	23	**0P0*	[IX+n]=(signed) [IX+n]/2
SRA (IY+n)	4	FD CB XX 2E	23	**0P0*	[IY+n]=(signed) [IY+n]/2
SRL r	2	CB 38+rb	8	**0P0*	r=(unsigned) r/2
SRL (HL)	2	CB 3E	15	**0P0*	[HL]=(unsigned) [HL]/2
SRL (IX+n)	4	DD CB XX 3E	23	**0P0*	[IX+n]=(unsigned) [IX+n]/2
SRL (IY+n)	4	FD CB XX 3E	23	**0P0*	[IY+n]=(unsigned) [IY+n]/2
RLD	2	ED 6F	18	**0P0-	{A, [HL]}={A, [HL]}<-4
RRD	2	ED 67	18	**0P0-	{A, [HL]}=4->{A, [HL]}

RLC is a circular 8-bit rotate, RL uses the Carry Flag as a ninth bit.  
Note that RLCA RRA etc. are quicker than RLC A RR A etc.

Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
IN A, (N)	2	DB XX	11	-----	A=[N]
IN A, (C)	2	ED 78	12	***P0-	A=[C]
IN B, (C)	2	ED 40	12	***P0-	B=[C]
IN C, (C)	2	ED 48	12	***P0-	C=[C]
IN D, (C)	2	ED 50	12	***P0-	D=[C]
IN E, (C)	2	ED 58	12	***P0-	E=[C]
IN H, (C)	2	ED 60	12	***P0-	H=[C]
IN L, (C)	2	ED 68	12	***P0-	L=[C]
IND	2	ED AA	16	***?1-	[HL]=[C], HL=HL-1, B=B-1
INDR	2	ED BA	21/16	01*?1-	IND until B=0
INI	2	ED A2	16	***?1-	[HL]=[C], HL=HL+1, B=B-1
INIR	2	ED B2	21/16	01*?1-	INI until B=0
OUT (N), A	2	D3 XX	11	-----	[N]=A
OUT (C), A	2	ED 79	12	-----	[C]=A
OUT (C), B	2	ED 41	12	-----	[C]=B
OUT (C), C	2	ED 49	12	-----	[C]=C
OUT (C), D	2	ED 51	12	-----	[C]=D
OUT (C), E	2	ED 59	12	-----	[C]=E
OUT (C), H	2	ED 61	12	-----	[C]=H
OUT (C), L	2	ED 69	12	-----	[C]=L
OUTD	2	ED AB	16	***?1-	[C]=[HL], HL=HL-1, B=B-1
OUTI	2	ED A3	16	***?1-	[C]=[HL], HL=HL+1, B=B-1
OTDR	2	ED BB	21/16	01*?1-	OUTD until B=0
OTIR	2	ED B3	21/16	01*?1-	OUTI until B=0
BIT b, (HL)	2	CB 46+8*b	12	**1*0-	[HL]&{2^b}
BIT b, (IX+n)	4	DD CB XX 46+8*b	20	**1*0-	[IX+n]&{2^b}
BIT b, (IY+n)	4	FD CB XX 46+8*b	20	**1*0-	[IY+n]&{2^b}
BIT b, r	2	CB 40+8*b+rb	8	**1*0-	r&{2^b}
SET b, (HL)	2	CB 86+8*b	15	-----	[HL]=[HL]&{~2^b}
SET b, (IX+n)	4	DD CB XX 86+8*b	23	-----	[IX+n]=[IX+n]&{~2^b}
SET b, (IY+n)	4	FD CB XX 86+8*b	23	-----	[IY+n]=[IY+n]&{~2^b}
SET b, r	2	CB 80+8*b+rb	8	-----	r=r&{~2^b}
SET b, (HL)	2	CB C6+8*b	15	-----	[HL]=[HL]v{2^b}
SET b, (IX+n)	4	DD CB XX C6+8*b	23	-----	[IX+n]=[IX+n]v{2^b}
SET b, (IY+n)	4	FD CB XX C6+8*b	23	-----	[IY+n]=[IY+n]v{2^b}
SET b, r	2	CB C0+8*b+rb	8	-----	r=rv{2^b}

b means bit. Can be 0-7. Increase the last byte of OP-Code with 8\*b.

It is quicker to AND or Shift a byte to check the status of a bit.

It is quicker to use OR or AND to set or clear a bit in the A register.

Mnemonic	Sz	OP-Code	Clock	Flags SZHPNC	Effect
NOP	1	00	4	-----	
DI	1	F3	4	-----	disable interrupts
EI	1	FB	4	-----	enable interrupts
HALT	1	76	4	-----	repeat NOP until interrupt
LD A,I	2	ED 57	9	**0*0-	A=I
LD I,A	2	ED 47	9	-----	I=A
IM 0	2	ED 46	8	-----	set interrupt 0
IM 1	2	ED 56	8	-----	set interrupt 1
IM 2	2	ED 5E	8	-----	set interrupt 2
LD A,R	2	ED 5F	9	**0*0-	A=R
LD R,A	2	ED 4F	9	-----	R=A

Interrupt Mode 1 uses the normal 50Hz Spectrum ROM interrupt routine at \$0038 (which is where a RST 56 or RST 38H instruction will go to).

Interrupt Mode 2 is the user-defined 50Hz vectored interrupt that typically requires a table 257 bytes long containing all the same value, starting on a 256-byte page boundary.

Set I to the high byte of the table address using LD A,n then LD I,A. The low byte of the table address is an unknown value when the interrupt occurs. The CPU will take two bytes from that address (low, then high) and jump to that address.

Note that when an interrupt occurs, interrupts become disabled so an EI must be used before another interrupt can occur.

If interrupts are disabled, or not re-enabled, HALT will hang forever.

Collected by Oscar Lindberg 1996-03-24  
(offler@skip.adb.gu.se) from:  
Z80 pocketbook  
Z80 assembly language programming

Edited and Promulgated by Thomas Scherrer: <http://z80.info>

PDF Booklet format by Jason J Railton 2020-10-30