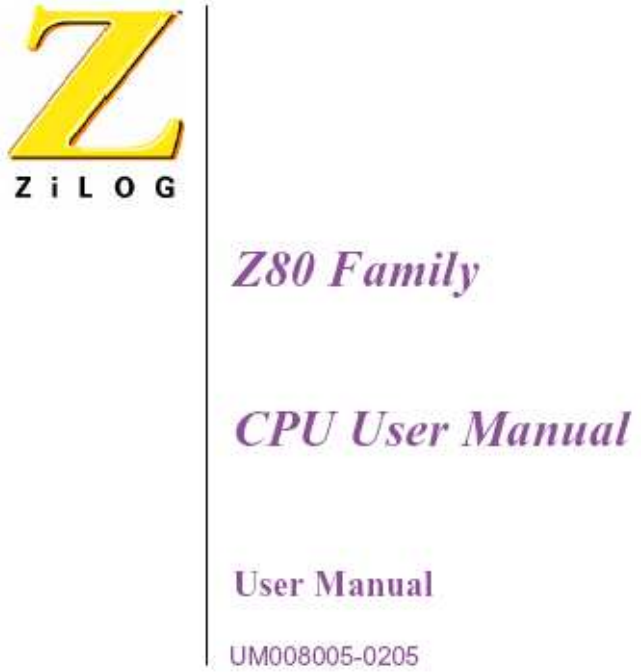
Technika mikroprocesorowa I

Studia niestacjonarne rok II

Wykład 2

Literatura:

www.zilog.com „Z80 Family, CPU User Manual”



Cykle magistrali w mikroprocesorze Z80

-odczyt kodu rozkazu,

-odczyt-zapis pamięci,

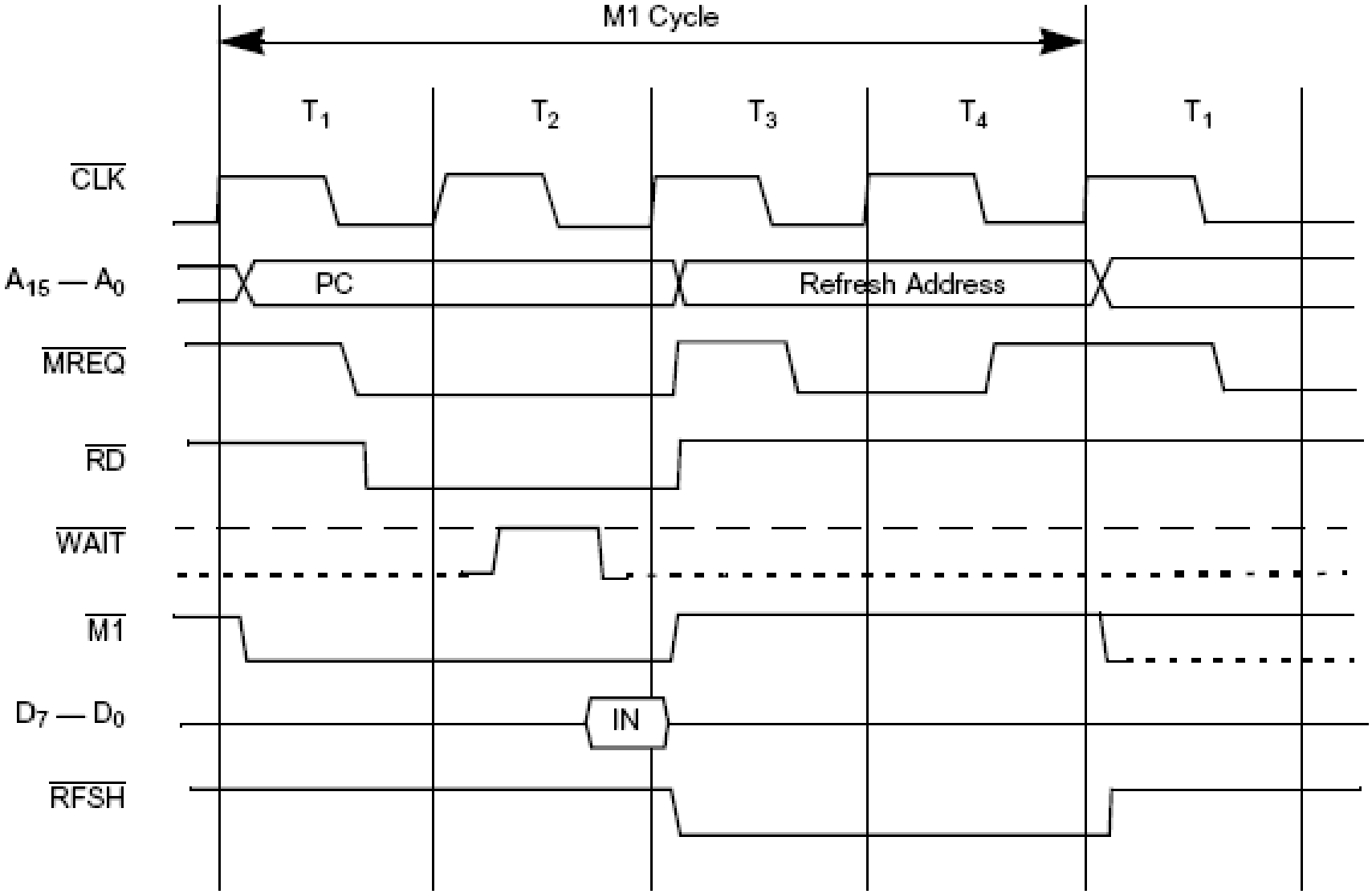
-odczyt-zapis urządzenia we-wy,

-cykl oddania magistrali,

-cykl przyjęcia przerwania maskowalnego,

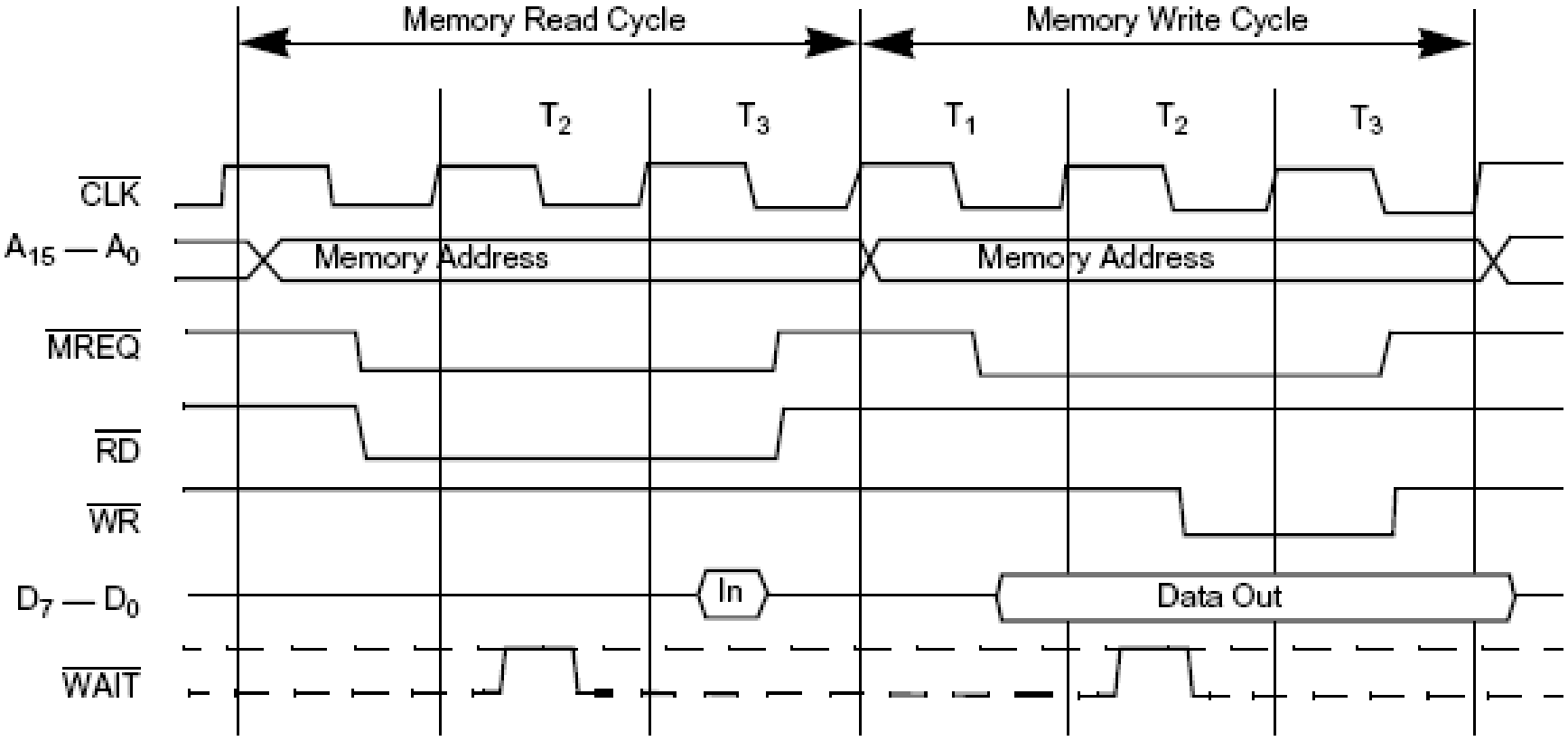
-cykl przyjęcia przerwania niemaskowalnego,

-wyjście z rozkazu HALT.

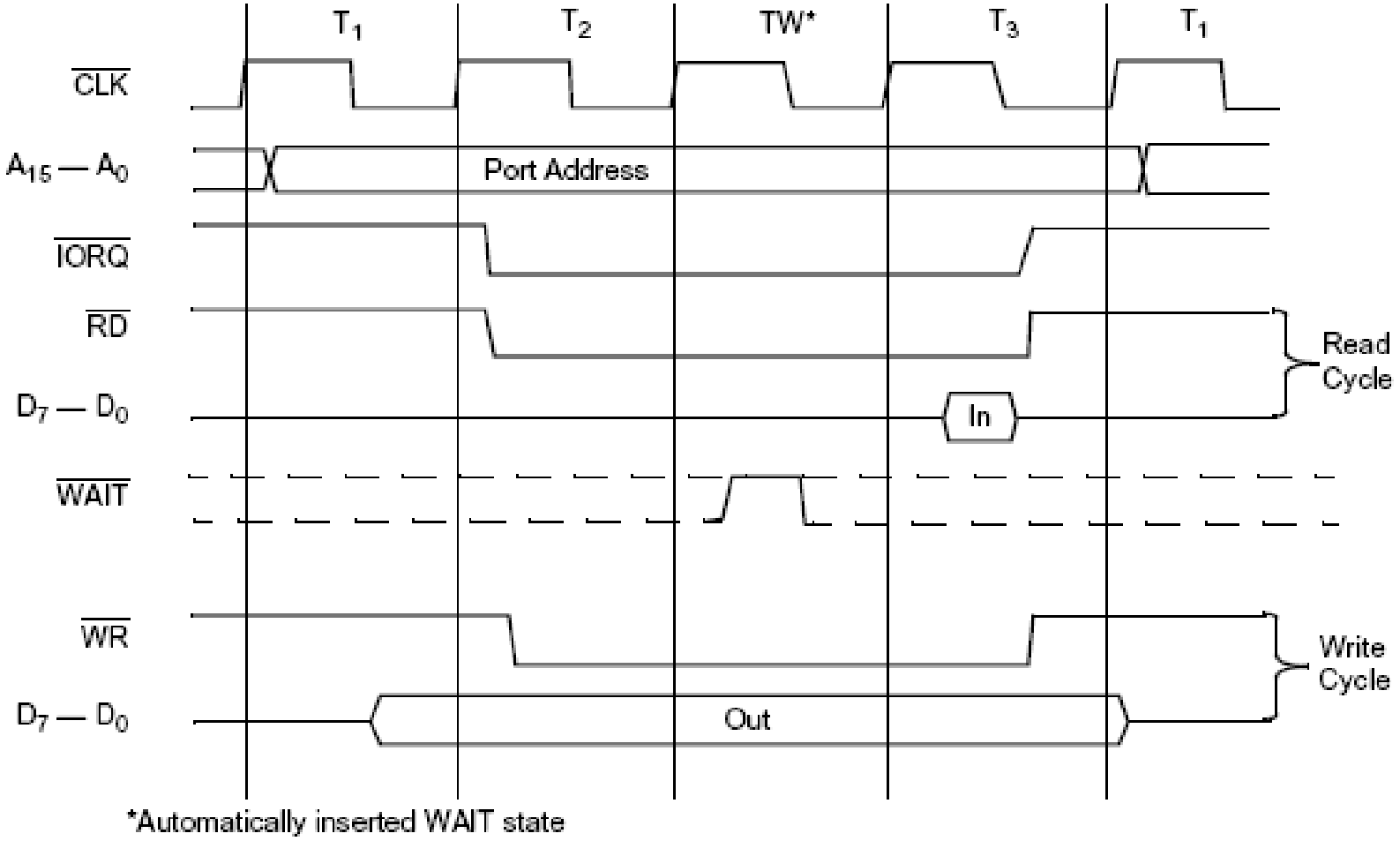


Pobranie kodu rozkazu z pamięci

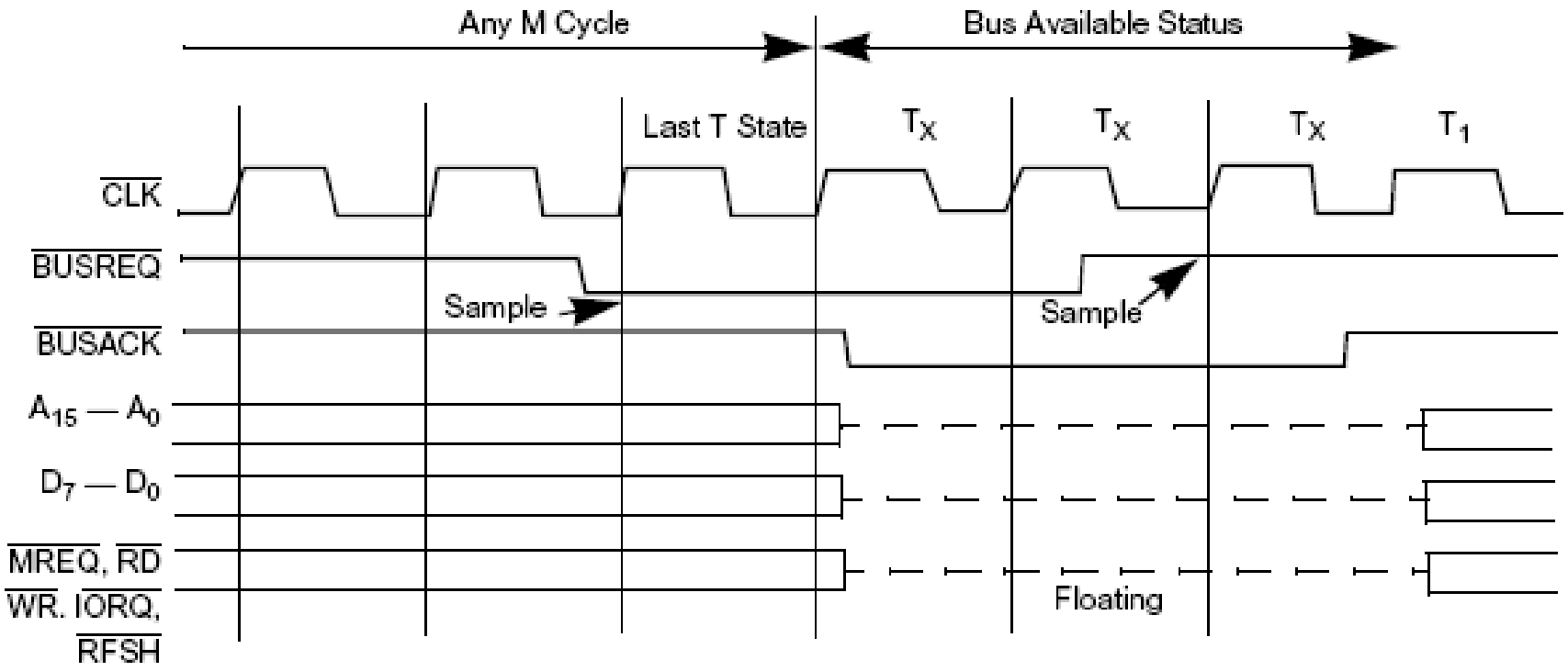
Cykl odczytu i zapisu pamięci



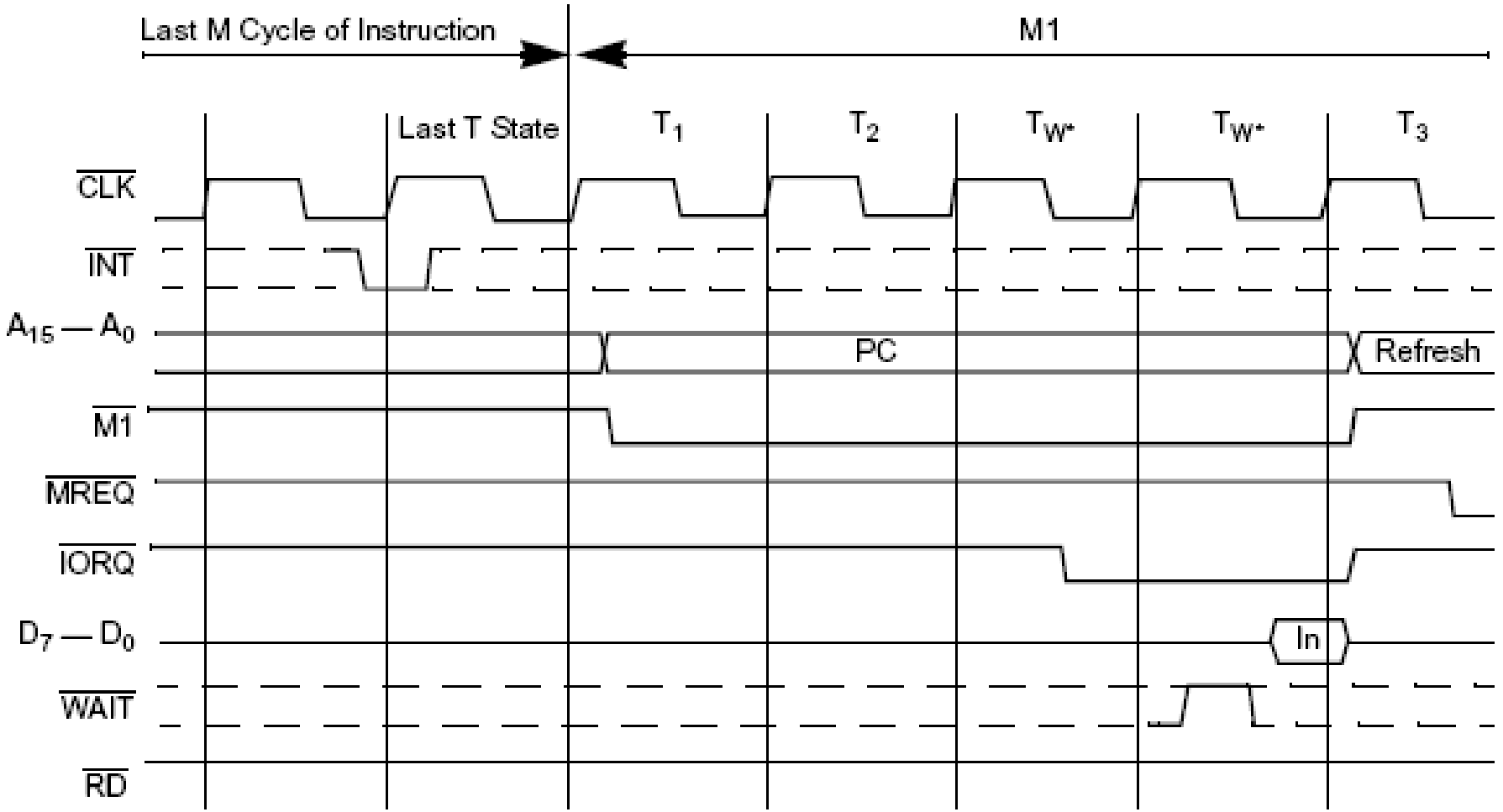
Cykl odczytu i zapisu urządzeń we-wy



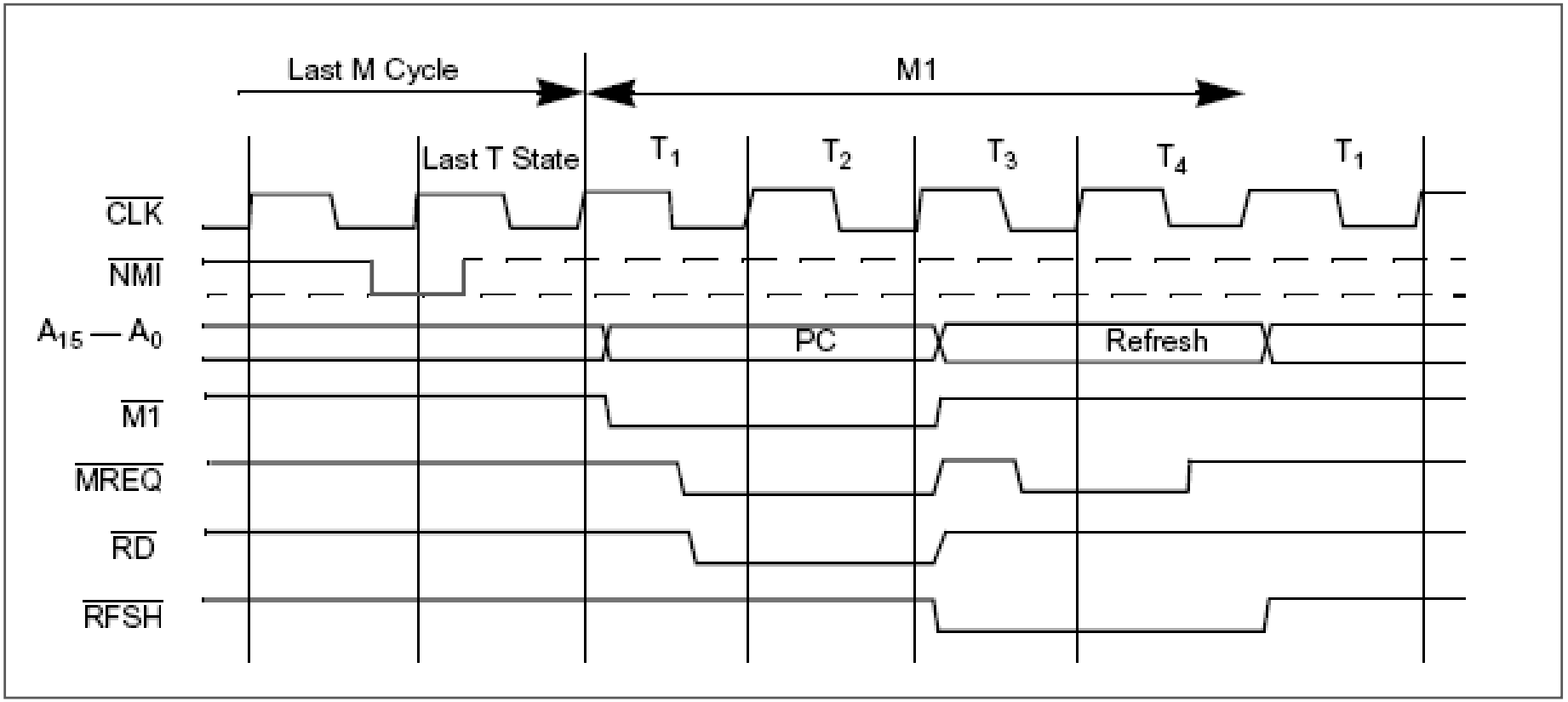
Cykl oddania magistrali



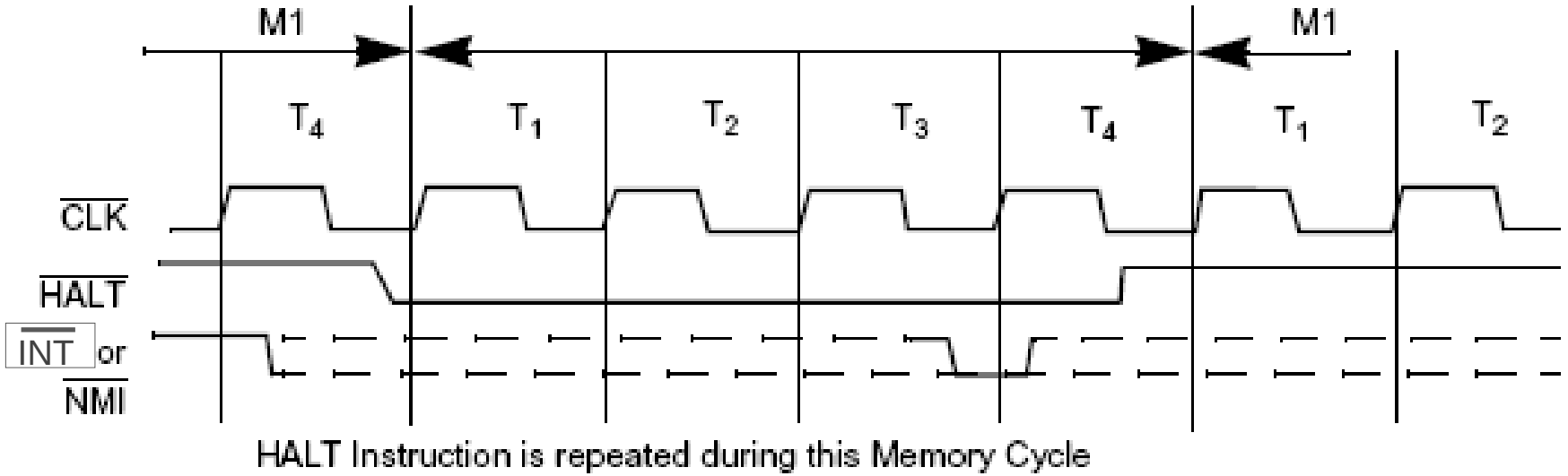
Cykl przyjęcia przerwania maskowalnego



Cykl przyjęcia przerwania niemaskowalnego



Wyjście z rozkazu HALT



System przerwań mikroprocesora Z80

Z80 posiada dwa wejścia zgłaszania przerwań:

NMI- przerwanie niemaskowalne,

INT- przerwanie maskowalne (odmaskowanie rozkazem EI (enable Ibterrupt), zamaskowanie rozkazem DI (Disable Interrupt).

Przerwanie NMI

Przerwanie jest aktywne opadającym zboczem na wejściu **NMI**.

Pojawienie się ujemnego zbocza (przejście z „1” na „0”), powoduje:

-złożenie na stosie adresu powrotu do programu głównego,

-załadowanie do PC adresu 0066h (stały adres obsługi przerwania niemaskowalnego).

Tryby przyjęcia przerwania maskowalnego:

W mikroprocesorze Z80 są 3 tryby przyjmowania przerwania maskowalnego, przełączane rozkazem IMx (Interrupt Mode x-numer modu).

W architekturze wewnętrznej mikroprocesora Z80 istnieją dwa przerzutniki flip-flop odpowiedzialne za aktywację obsługi przerwań maskowalnych. Noszą nazwę odpowiednio: **IFF1 i IFF2**.

IFF1- włącza i wyłącza obsługę przerwań maskowalnych (jest maską przerwań maskowalnych, po RESET- zerowany),

IFF2 -stanowi tymczasowe miejsce przechowywania stanu maski przerwań (stanu przerzutnika- IFF1 jest przepisywany do IFF2), **po RESET IFF1 i IFF2 są zerowane**.

Gdy pojawi się przerwanie **niemaskowalne (wyższy priorytet)**, stan maski przerwań IFF1 jest zapamiętywany w rejestrze bitowym IFF2. Zdekodowanie rozkazu powrotu z przerwania niemaskowalnego NMI (RETN) powoduje odtworzenie stanu IFF1 na podstawie IFF2.

Tryb 0- odpowiada reakcji na przerwania mikroprocesora 8080. W tym trybie urządzenie przerywające, w cyklu akceptacji przerwania, umieszcza na magistrali danych dowolną instrukcję, a mikroprocesor ją wykonuje. Dedykowaną instrukcją jest rozkaz restartu **RST n** (n:0-7). Instrukcja ta powoduje złożenie na stosie adresu powrotu i skok do procedury obsługi zależny od numeru restartu:

n=0 adres 0000h

n=1 adres 0008h

n=2 adres 0010h

n=3 adres 0018h

n=4 adres 0020h

n=5 adres 0028h

n=6 adres 0030h

n=7 adres 0038h

Tryb 1- w tym trybie mikroprocesor reaguje na przerwanie przez wykonanie procedury obsługi od adresu **0038H (po złożeniu na stosie** . Zatem reakcja jest identyczna jak reakcja na przerwanie niemaskowalne, w którym adresem wywoływanym jest adres **0066H**.

Tryb 2- jest najbardziej elastycznym trybem reakcji na przerwanie. W tym trybie programista tworzy tablicę 16-bitowych adresów startowych dla każdej procedury obsługi przerwania. Tablica ta może być umieszczona w dowolnym miejscu w pamięci (decyduje o tym zawartość rejestru I). Gdy przerwanie zostanie przyjęte, urządzenie zgłaszające wystawia na magistrali danych 8-mio bitową, mniej znaczącą, zawsze parzystą część 16-bitowego wskaźnika, w celu pobrania z tablicy adresu startowego obsługi przerwania. W związku z powyższym tryb 2 dopuszcza tylko 128 rozróżnialnych źródeł przerwania.

Tryby adresacji w mikroprocesorze Z80

Mikroprocesor Z80 realizuje następujące tryby adresowania:

-natychmiastowy – argument bezpośrednio za kodem operacji (LD A,0F – załaduj do akumulatora liczbę 0F),

-bezpośredni – dwubajtowy adres argumentu znajduje się w pamięci bezpośrednio za kodem operacji (LD A,(700) – załaduj do akumulatora zawartość komórki pamięci o adresie 0700),

-pośredni rejestrowy – dwubajtowy adres argumentu znajduje się w parze rejestrów BC, DE, HL lub SP (LD A,(BC) –załaduj do akumulatora zawartość komórki pamięci o adresie zawartym w parze rejestrów BC),

-indeksowy– dwubajtowy adres argumentu tworzony jest jako suma zawartości jednego z rejestrów indeksowych IX lub IY i jednobajtowego przesunięcia w kodzie U2 zapisanego za kodem operacji; np. LD A,(IX+8) – załaduj do akumulatora zawartość komórki pamięci o adresie zawartym w rejestrze IX zwiększonym o 8;

-względny– tryb wykorzystywany tylko przez rozkaz skoku względnego JR – zawartość rejestru PC jest modyfikowana przez dodanie jednobajtowego przesunięcia w kodzie U2 umieszczonego w pamięci za kodem operacji skoku;,

-rejestrowy– argument znajduje się w jednym z rejestrów lub w parze rejestrów (w przypadku argumentu 16-bitowego); np. ADD IX,SP – dodaj do zawartości rej. IX wartość rej SP;

-strony zerowej, pojedynczych bitów i implikowany – dotyczą małej liczby rozkazów sterujących (np. RST 8 – wywołanie podprogramu pod adresem 8), logicznych-operacje na pojedynczych bitach (np. RES 0,A – skasowanie bitu 0 w akumulatorze) oraz rozkazów dla których jednoznacznie określone jest położenie argumentu (np. DAA – korekcja dziesiętna akumulatora po operacjach arytmetycznych).

Lista rozkazów mikroprocesora Z80

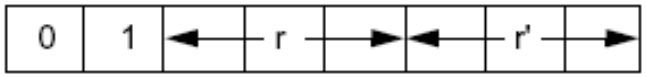
Instrukcje przesyłania danych (8-mio bitowych)

LD r, r'

Operation: r, ← r'

Op Code: LD

Operands: r, r'



Description: The contents of any register r' are loaded to any other register r. r, r' identifies any of the registers A, B, C, D, E, H, or L, assembled as follows in the object code:

Register	r, C
A	111
B	000
C	001
D	010
E	011
H	100
L	101

M Cycles	T States	MHz E.T.
1	4	1.0

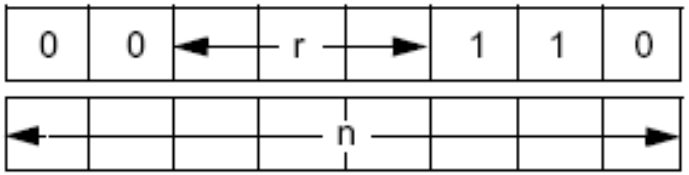
Prześlij zawartość rejestru r’ do rejestru r

LD r,n

Operation: r ← n

Op Code: LD

Operands: r, n



Description: The 8-bit integer n is loaded to any register r, where r identifies register A, B, C, D, E, H, or L, assembled as follows in the object code:

Register	r
A	111
B	000
C	001
D	010
E	011
H	100
L	101

M Cycles	T States	4 MHz E.T.
2	7 (4, 3)	1.75

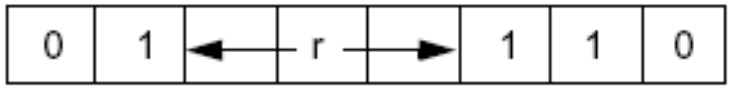
Wpisz do rejestru r liczbę 8-mio bitową n

LD r, (HL)

Operation: r ← (HL)

Op Code: LD

Operands: r, (HL)



Description: The 8-bit contents of memory location (HL) are loaded to register r, where r identifies register A, B, C, D, E, H, or L, assembled as follows in the object code:

Register	r
A	111
B	000
C	001
D	010
E	011
H	100
L	101

M Cycles	T States	4 MHz E.T.
2	7 (4, 3)	1.75

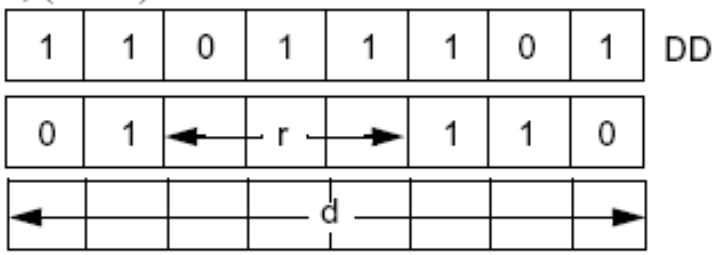
Załaduj do rejestru r zawartość komórki pamięci o adresie zawartym w HL

LD r, (IX+d)

Operation: r ← (IX+d)

Op Code: LD

Operands: r, (IX+d)



Description: The operand (IX+d), (the contents of the Index Register IX summed with a two's complement displacement integer d) is loaded to register r, where r identifies register A, B, C, D, E, H, or L, assembled as follows in the object code:

Register	r
A	111
B	000
C	001
D	010
E	011
H	100
L	101

M Cycles	T States	4 MHz E.T.
5	19 (4, 4, 3, 5, 3)	2.50

Załaduj do r zawartość komórki o adresie zawartej w IX+ offset d

LD A, (DE)

Operation: A ← (DE)

Op Code: LD

Operands: A, (DE)

0	0	0	1	1	0	1	0	1A

Description: The contents of the memory location specified by the register pair DE are loaded to the Accumulator.

M Cycles	T States	4 MHz E.T.
2	7 (4, 3)	1.75

Condition Bits Affected: None

Załaduj do akumulatora A zawartość komórki pamięci o adresie zawartym w parze rejestrów DE

LD (BC), A

Operation: (BC) ← A

Op Code: LD

Operands: (BC), A

0	0	0	0	0	0	1	0	02

Description: The contents of the Accumulator are loaded to the memory location specified by the contents of the register pair BC.

M Cycles	T States	4 MHz E.T.
2	7 (4, 3)	1.75

Załaduj zawartość akumulatora do komórki pamięci o adresie zawartym w parze rejestrów BC

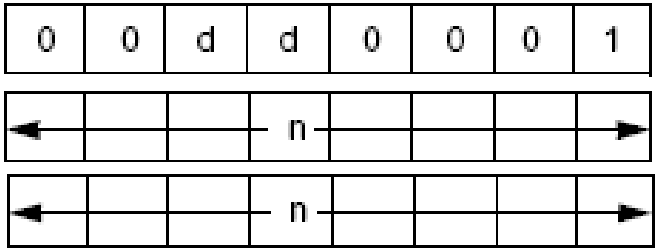
Instrukcje przesyłania danych (16-to bitowych)

LD dd, nn

Operation: dd ← nn

Op Code: LD

Operands: dd, nn



Description: The 2-byte integer nn is loaded to the dd register pair, where dd defines the BC, DE, HL, or SP register pairs, assembled as follows in the object code:

Pair	dd
BC	00
DE	01
HL	10
SP	11

The first n operand after the Op Code is the low order byte.

M Cycles	T States	4 MHz E.T.
2	10 (4, 3, 3)	2.50

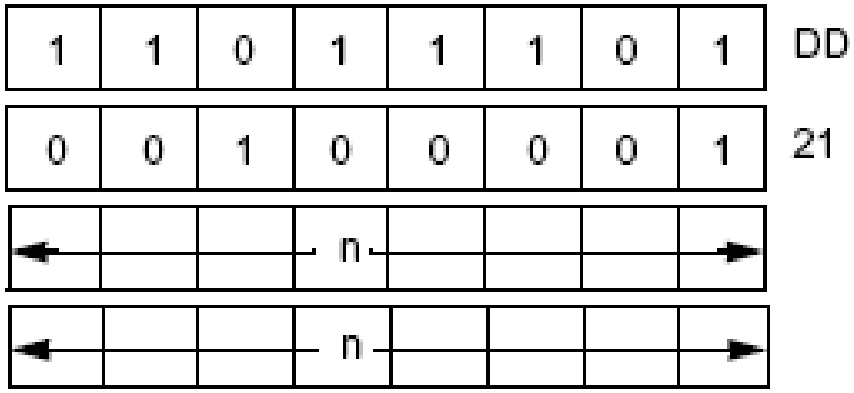
Zapisz do pary rejestrów dd daną 16-to bitową nn

LD IX, nn

Operation: Ix ← nn

Op Code: LD

Operands: IX, nn



Description: Integer nn is loaded to the Index Register IX. The first n operand after the Op Code is the low order byte.

M Cycles	T States	4 MHz E.T.
4	14 (4, 4, 3, 3)	3.50

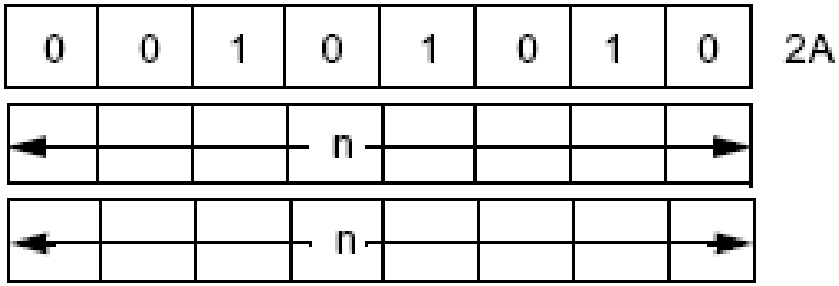
Zapisz do rejestru indeksowego IX liczbę 16-to bitową nn

LD HL, (nn)

Operation: H ← (nn+1), L ← (nn)

Op Code: LD

Operands: HL, (nn)



Description: The contents of memory address (nn) are loaded to the low order portion of register pair HL (register L), and the contents of the next highest memory address (nn+1) are loaded to the high order portion of HL (register H). The first n operand after the Op Code is the low order byte of nn.

M Cycles	T States	4 MHz E.T.
5	16 (4, 3, 3, 3, 3)	4.00

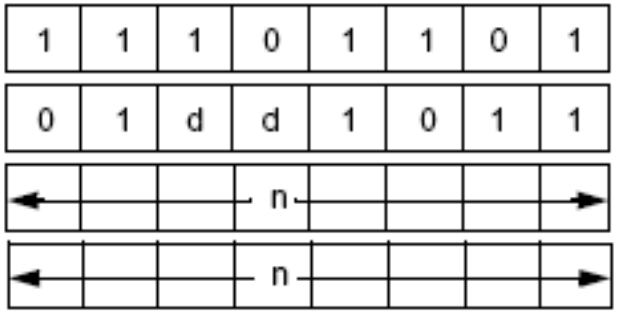
Załadowanie zawartość dwóch kolejnych komórek pamięci spod adresu nn do pary rejestrów HL

LD dd, (nn)

Operation: ddh ← (nn+1) ddl ← (nn)

Op Code: LD

Operands: dd, (nn)



Description: The contents of address (nn) are loaded to the low order portion of register pair dd, and the contents of the next highest memory address (nn+1) are loaded to the high order portion of dd. Register pair dd defines BC, DE, HL, or SP register pairs, assembled as follows in the object code:

Pair	dd
BC	00
DE	01
HL	10
SP	11

The first n operand after the Op Code is the low order byte of (nn).

M Cycles	T States	4 MHz E.T.
6	20 (4, 4, 3, 3, 3, 3)	5.00

Załadowanie do pary rejestrów dd zawartości komórek pamięci spod adresu nn

PUSH qq

Operation: (SP-2) ← qqL, (SP-1) ← qqH

Op Code: PUSH

Operands: qq

1	1	q	q	0	1	0	1

Description: The contents of the register pair qq are pushed to the external memory LIFO (last-in, first-out) Stack. The Stack Pointer (SP) register pair holds the 16-bit address of the current top of the Stack. This instruction first decrements SP and loads the high order byte of register pair qq to the memory address specified by the SP. The SP is decremented again and loads the low order byte of qq to the memory location corresponding to this new address in the SP. The operand qq identifies register pair BC, DE, HL, or AF, assembled as follows in the object code:

Pair	qq
BC	00
DE	01
HL	10
AF	11

M Cycles	T States	4 MHz E.T.
3	11 (5, 3, 3)	2.75

Złóż na stosie parę rejestrów qq

POP qq

Operation: qqH ← (SP+1), qqL ← (SP)

Op Code: POP

Operands: qq

1	1	q	q	0	0	0	1

Description: The top two bytes of the external memory LIFO (last-in, first-out) Stack are popped to register pair qq. The Stack Pointer (SP) register pair holds the 16-bit address of the current top of the Stack. This instruction first loads to the low order portion of qq, the byte at memory location corresponding to the contents of SP; then SP is incriminated and the contents of the corresponding adjacent memory location are loaded to the high order portion of qq and the SP is now incriminated again. The operand qq identifies register pair BC, DE, HL, or AF, assembled as follows in the object code:

Pair	r
BC	00
DE	01
HL	10
AF	11

M Cycles	T States	4 MHz E.T.
3	10 (4, 3, 3)	2.50

Ściągnij ze stosu dane do pary rejestrów qq

Instrukcje arytmetyczne 8-mio bitowe

ADD A, r

Operation: A ← A + r

Op Code: ADD

Operands: A, r

Description: The contents of register r are added to the contents of the Accumulator, and the result is stored in the Accumulator. The symbol r identifies the registers A, B, C, D, E, H, or L, assembled as follows in the object code:

Register	r
A	111
B	000
C	001
D	010
E	011
H	100
L	101

M Cycles	T States	4 MHz E.T.
1	4	1.00

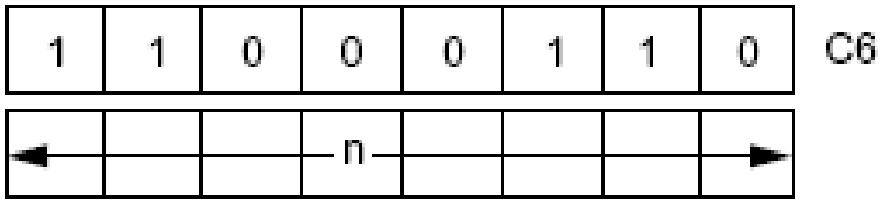
Dodaj do zawartości A zawartość innego rejestru, wynik umieść w A

ADD A, n

Operation: $A \leftarrow A + n$

Op Code: ADD

Operands: A, n



Description: The integer n is added to the contents of the Accumulator, and the results are stored in the Accumulator.

M Cycles	T States	4 MHz E.T.
2	7 (4, 3)	1.75

Dodaj do zawartości A liczbę 8-mio bitową, wynik umieść w A

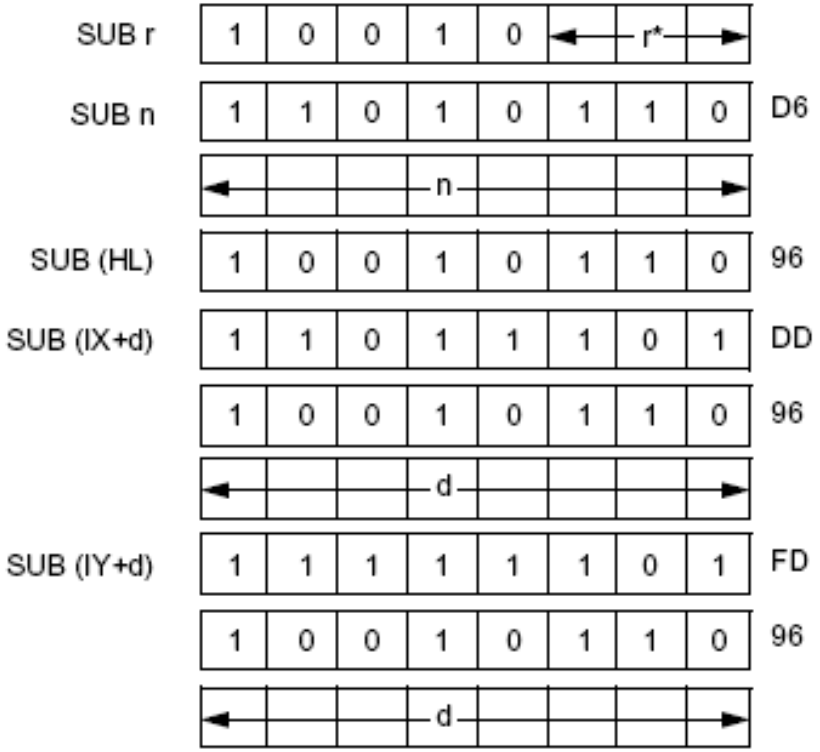
SUB s

Operation: A ← A - s

Op Code: SUB

Operands: s

This s operand is any of r, n, (HL), (IX+d), or (IY+d) as defined for the analogous ADD instruction. These possible Op Code/operand combinations are assembled as follows in the object code:



Register	r
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Odejmij od zawartości A zawartość: rejestru, liczbę n, zawartość komórki pamięci adresowanej przez HL, IX, IY, wynik w A

SBC A, s

Operation: A ← A - s - CY

Op Code: SBC

Operands: A, s

The s operand is any of r, n, (HL), (IX+d), or (IY+d) as defined for the analogous ADD instructions. These possible Op Code/operand combinations are assembled as follows in the object code:

SBC A, r	1	0	0	1	1	←	r*	→
SBC A, n	1	1	0	1	1	1	1	0	DE
	←			n				→
SBC A, (HL)	1	0	0	1	1	1	1	0	9E
SBC A, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	0	0	1	1	1	1	0	9E
	←			d				→
SBC A, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	0	0	1	1	1	1	0	9E
	←			d				→

Register	r
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Odejmij od zawartości A zawartość: rejestru, liczbę n, zawartość komórki pamięci adresowanej przez HL, IX, IY oraz znacznika C, wynik w A

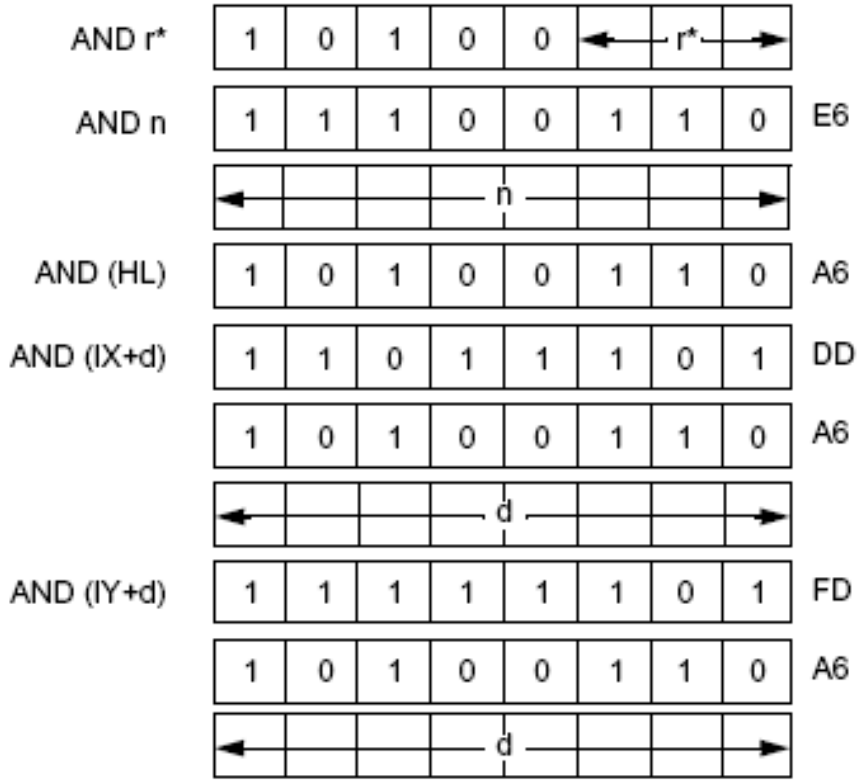
AND s

Operation: $A \leftarrow A \wedge s$

Op Code: AND

Operands: s

The s operand is any of r, n, (HL), (IX+d), or (IY+d), as defined for the analogous ADD instructions. These possible Op Code/operand combinations are assembled as follows in the object code:



Register	r
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Wymnóż logicznie zawartość A z zawartością: rejestru, liczbą n, zawartością komórki pamięci adresowanej przez HL, IX, IY, wynik w A

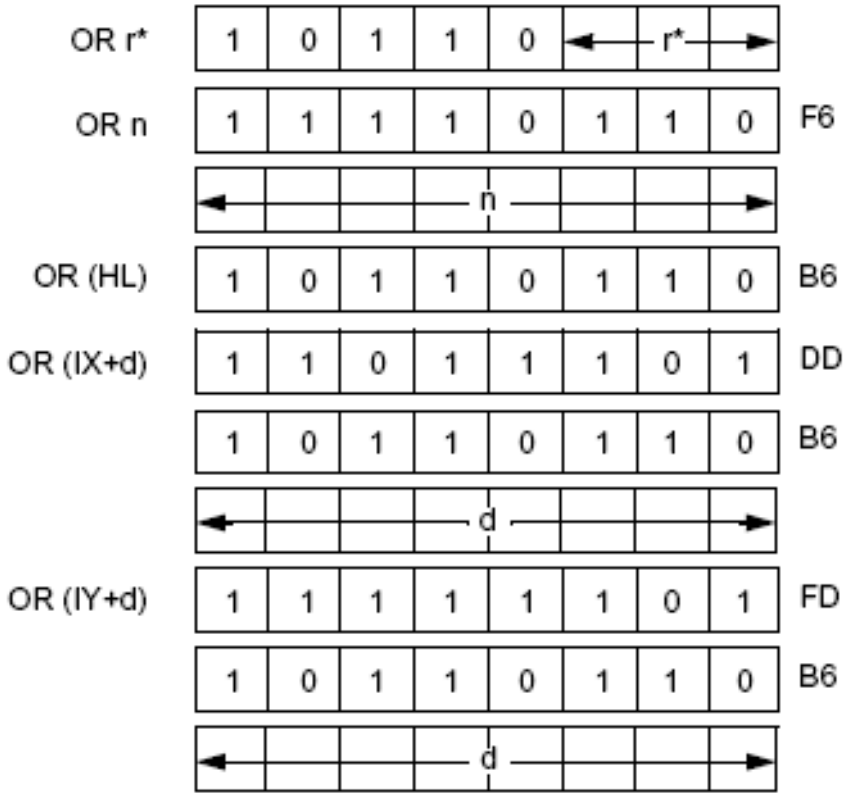
OR s

Operation: $A \leftarrow A \vee s$

Op Code: OR

Operands: s

The s operand is any of r, n, (HL), (IX+d), or (IY+d), as defined for the analogous ADD instructions. These possible Op Code/operand combinations are assembled as follows in the object code:



Register	r
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Dodaj logicznie zawartość A z zawartością: rejestru, liczbą n, zawartością komórki pamięci adresowanej przez HL, IX, IY, wynik w A

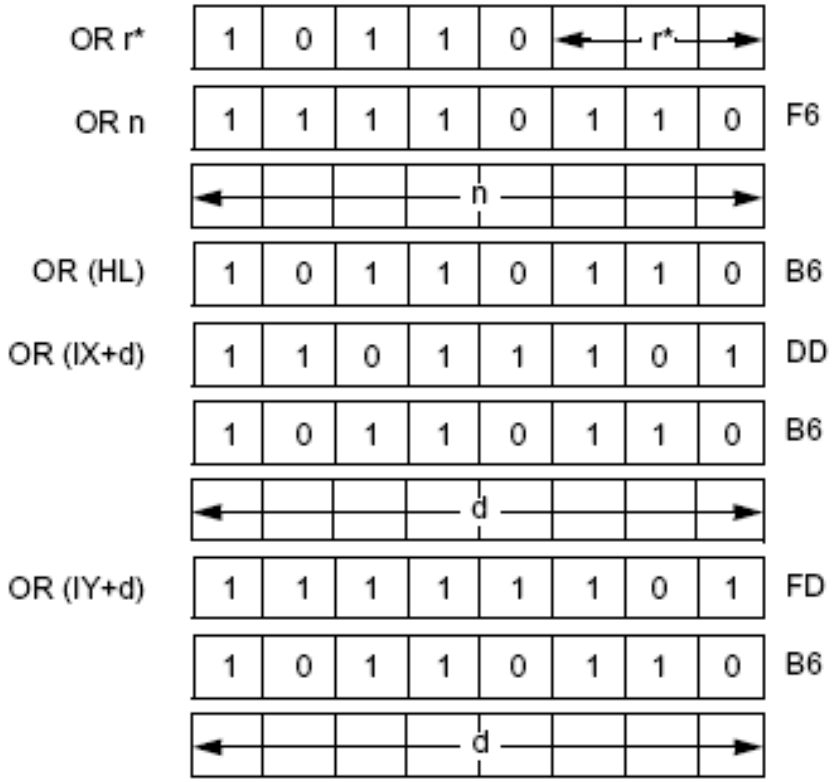
XOR s

Operation: $A \leftarrow A \oplus s$

Op Code: XOR

Operands: s

The s operand is any of r, n, (HL), (IX+d), or (IY+d), as defined for the analogous ADD instructions. These possible Op Code/operand combinations are assembled as follows in the object code:



Register	r
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Wykonaj operację XOR zawartości A z zawartością: rejestru, liczbą n, zawartością komórki pamięci adresowanej przez HL, IX, IY, wynik w A

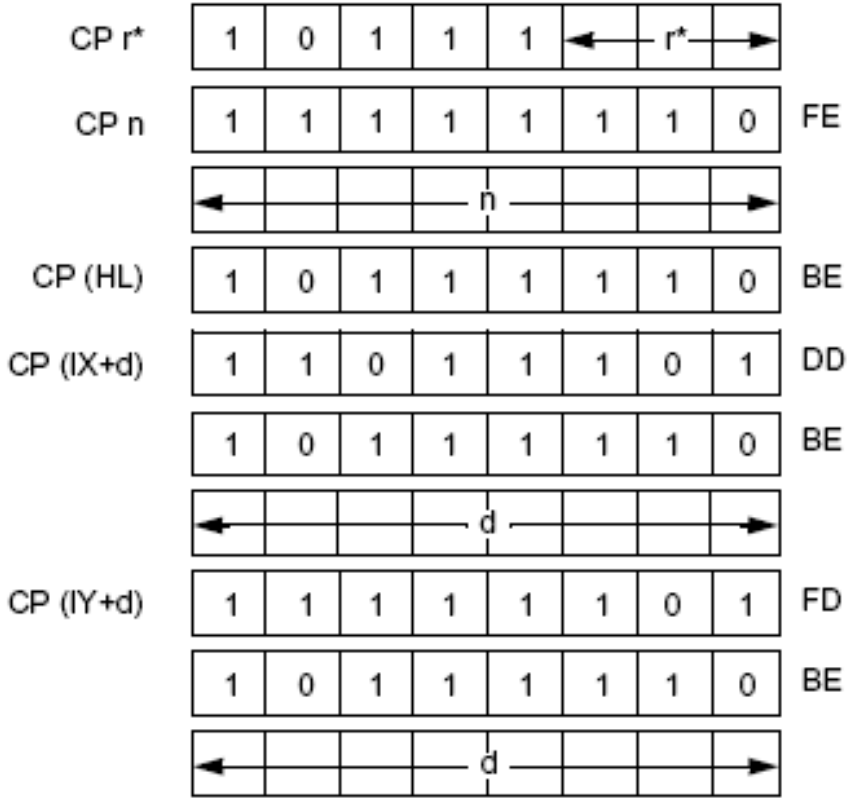
CP s

Operation: A - s

Op Code: CP

Operands: s

The s operand is any of r, n, (HL), (IX+d), or (IY+d), as defined for the analogous ADD instructions. These possible Op Code/operand combinations are assembled as follows in the object code:



Register	r
B	000
C	001
D	010
E	011
H	100
L	101
A	111

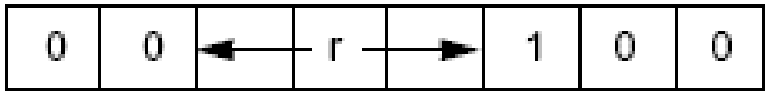
Porównaj zawartość A z zawartością: rejestru, liczbą n, zawartością komórki pamięci adresowanej przez HL, IX, IY, wyniku nie zapisuj

INC r

Operation: r ← r + 1

Op Code: INC

Operands: r



Description: Register r is incremented and register r identifies any of the registers A, B, C, D, E, H, or L, assembled as follows in the object code.

Register	r
A	111
B	000
C	001
D	010
E	011
H	100
L	101

M Cycles	T States	4 MHz E.T.
1	4	1.00

Zwiększ o 1 zawartość rejestru r

DEC m

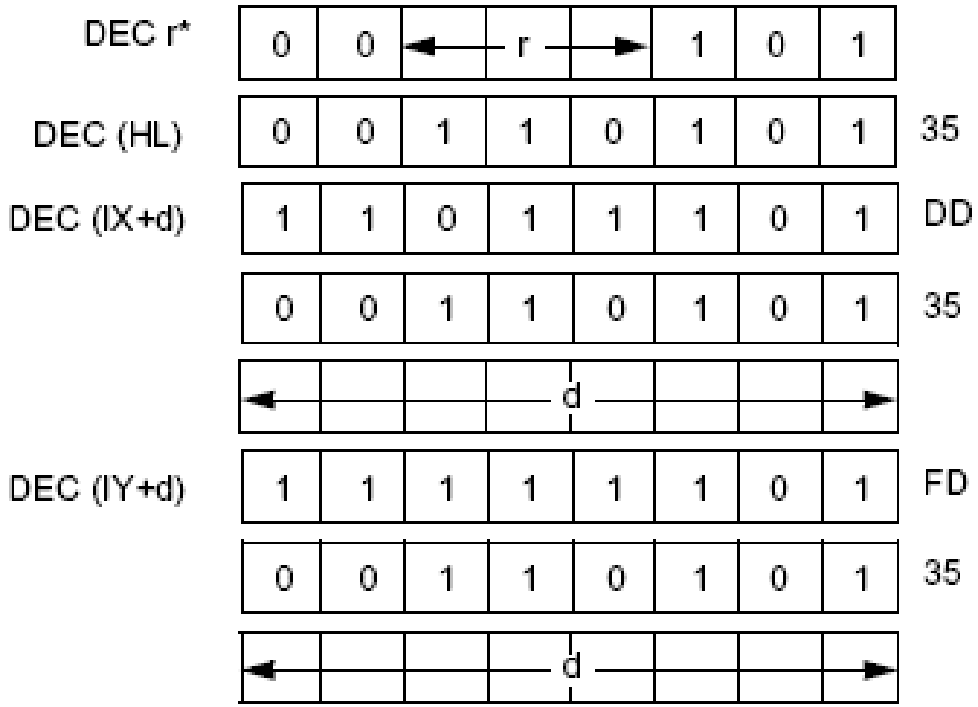
Register	r
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Operation: m ← m- 1

Op Code: DEC

Operands: m

The m operand is any of r, (HL), (IX+d), or (IY+d), as defined for the analogous INC instructions. These possible Op Code/operand combinations are assembled as follows in the object code:



Zmniejsz o 1 zwartość rejestru r, komórki pamięci o adresie zawartym w HL, o adresie zwartym w IX lub IY z offsetem d

Rozkazy arytmetyczne ogólnego przeznaczenia oraz sterujące CPU

DAA

Operation:

Op Code: DAA

0	0	1	0	0	1	1	1	27

M Cycles	T States	4 MHz E.T.
1	4	1.00

Korekcja dziesiętna po dodawaniu i odejmowaniu

Operation	C Before DAA	Hex Value In Upper Digit (bit 7-4)	H Before DAA	Hex Value In Lower Digit (bit 3-0)	Number Added To Byte	C After DAA
	0	9-0	0	0-9	00	0
	0	0-8	0	A-F	06	0
	0	0-9	1	0-3	06	0
ADD	0	A-F	0	0-9	60	1
ADC	0	9-F	0	A-F	66	1
INC	0	A-F	1	0-3	66	1
	1	0-2	0	0-9	60	1
	1	0-2	0	A-F	66	1
	1	0-3	1	0-3	66	1
SUB	0	0-9	0	0-9	00	0
SBC	0	0-8	1	6-F	FA	0
DEC	1	7-F	0	0-9	A0	1
NEG	1	6-7	1	6-F	9A	1

Przykłady korekcji dziesiętnej

CPL

Operation: $A \leftarrow \overline{A}$

Op Code: CPL

0	0	1	0	1	1	1	1	2F

Description: The contents of the Accumulator (register A) are inverted (one's complement).

M Cycles	T States	4 MHz E.T.
1	4	1.00

Zanegowanie zawartości akumulatora

NEG

Operation: A ← 0 - A

Op Code: NEG

1	1	1	0	1	1	0	1	ED
0	1	0	0	0	1	0	0	44

Description: The contents of the Accumulator are negated (two's complement). This is the same as subtracting the contents of the Accumulator from zero. Note that 80H is left unchanged.

M Cycles	T States	4 MHz E.T.
2	8 (4, 4)	2.00

Example: If the contents of the Accumulator are

1	0	0	1	1	0	0	0

at execution of `NEG` the Accumulator contents are

0	1	1	0	1	0	0	0

Negacja zawartości akumulatora

CCF

Operation: $CY \leftarrow \overline{CY}$

Op Code: CCF

0	0	1	1	1	1	1	1	3F

Description: The Carry flag in the F register is inverted.

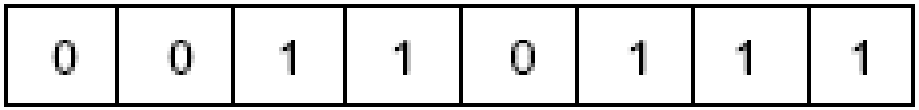
M Cycles	T States	4 MHz E.T.
1	4	1.00

Negacja bitu przeniesienia

SCF

Operation: CY ← 1

Op Code: SCF



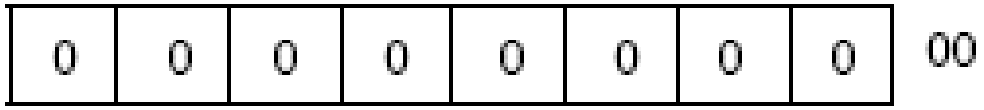
Description: The Carry flag in the F register is set.

M Cycles	T States	4 MHz E.T.
1	4	1.00

Ustawienie bitu przeniesienia

Operation: —

Op Code: NOP



Description: The CPU performs no operation during this machine cycle.

M Cycles	T States	4 MHz E.T.
1	4	1.00

Operacja pusta (No operating) nic nie rób

Rozkazy arytmetyczne 16-to bitowe

ADD HL, ss

Operation: HL ← HL + ss

Op Code: ADD

Operands: HL, ss

0	0	s	s	1	0	0	1

Description: The contents of register pair ss (any of register pairs BC, DE, HL, or SP) are added to the contents of register pair HL and the result is stored in HL. Operand ss is specified as follows in the assembled object code.

Register Pair	ss
BC	00
DE	01
HL	10
SP	11

M Cycles	T States	4 MHz E.T.
3	11 (4, 4, 3)	2.75

Do zawartości HL dodaj zawartości pary rejestrów ss, wynik w HL

ADC HL, ss

Operation: HL ← HL + ss + CY

Op Code: ADC

Operands: HL, ss

1	1	1	0	1	1	0	1	ED
0	1	s	s	1	0	1	0	

Description: The contents of register pair ss (any of register pairs BC, DE, HL, or SP) are added with the Carry flag (C flag in the F register) to the contents of register pair HL, and the result is stored in HL. Operand ss is specified as follows in the assembled object code.

Register Pair	ss
BC	00
DE	01
HL	10
SP	11

M Cycles	T States	4 MHz E.T.
4	15 (4, 4, 4, 3)	3.75

Do zawartości HL dodaj zawartości pary rejestrów ss oraz bitu C, wynik w HL

SBC HL, ss

Operation: HL ← HI - ss - CY

Op Code: SBC

Operands: HL, ss

1	1	1	0	1	1	0	1	ED
0	1	s	s	0	0	1	0	

Description: The contents of the register pair ss (any of register pairs BC, DE, HL, or SP) and the Carry Flag (C flag in the F register) are subtracted from the contents of register pair HL, and the result is stored in HL. Operand ss is specified as follows in the assembled object code.

Register Pair	ss
BC	00
DE	01
HL	10
SP	11

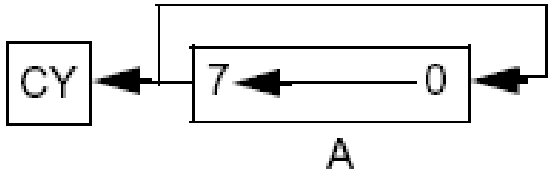
M Cycles	T States	4 MHz E.T.
4	15 (4, 4, 4, 3)	3.75

Odejmij od zawartości HL zawartość pary rejestrów ss oraz bitu C, wynik w HL

Rozkazy rotacji i przesunięć

RLCA

Operation:



Op Code: RLCA

Operands: —

0	0	0	0	0	1	1	1	07

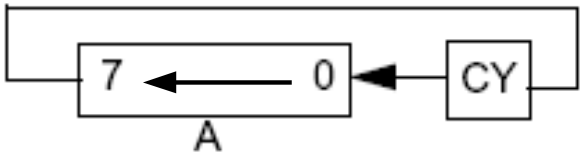
Description: The contents of the Accumulator (register A) are rotated left 1-bit position. The sign bit (bit 7) is copied to the Carry flag and also to bit 0. Bit 0 is the least-significant bit.

M cycles	T States	4 MHz E.T.
1	4	1.00

Przesuń zawartość akumulatora A o jeden bit w lewo, najstarszy bit 7 wpisz na bit 0 oraz na bit przeniesienia C

RLA

Operation:



Op Code: RLA

Operands: —

0	0	0	1	0	1	1	1

17

Description: The contents of the Accumulator (register A) are rotated left 1-bit position through the Carry flag. The previous content of the Carry flag is copied to bit 0. Bit 0 is the least-significant bit.

M Cycles	T States	4 MHz E.T.
1	4	1.00

Przesuń zawartość akumulatora o jeden bit w lewo „via” bit przeniesienia C. Najstarszy bit 7 A wpisz do C, C przepisz na najmłodszy bit 0 A

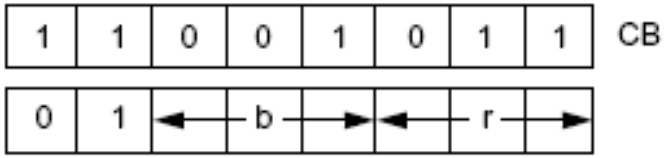
Rozkazy ustawiania, kasowania i testowania bitów

BIT b, r

Operation: $Z \leftarrow \overline{rb}$

Op Code: BIT

Operands: b, r



Description: This instruction tests bit b in register r and sets the Z flag accordingly. Operands b and r are specified as follows in the assembled object code:

Bit Tested	b	Register	r
0	000	B	000
1	001	C	001
2	010	D	010
3	011	E	011
4	100	H	100
5	101	L	101
6	110	A	111
7	111		

M Cycles	T States	4 MHz E.T.
2	8 (4, 4)	4.50

Wpisanie od znacznika Z zanegowanej wartości bitu o numerze b (0-7) w rejestrze r

BIT b, (HL)

Operation: $Z \leftarrow \overline{(HL)}b$

Op Code: BIT

Operands: b, (HL)

1	1	0	0	1	0	1	1	CB
0	1		b		1	1	0	

Description: This instruction tests bit b in the memory location specified by the contents of the HL register pair and sets the Z flag accordingly. Operand b is specified as follows in the assembled object code:

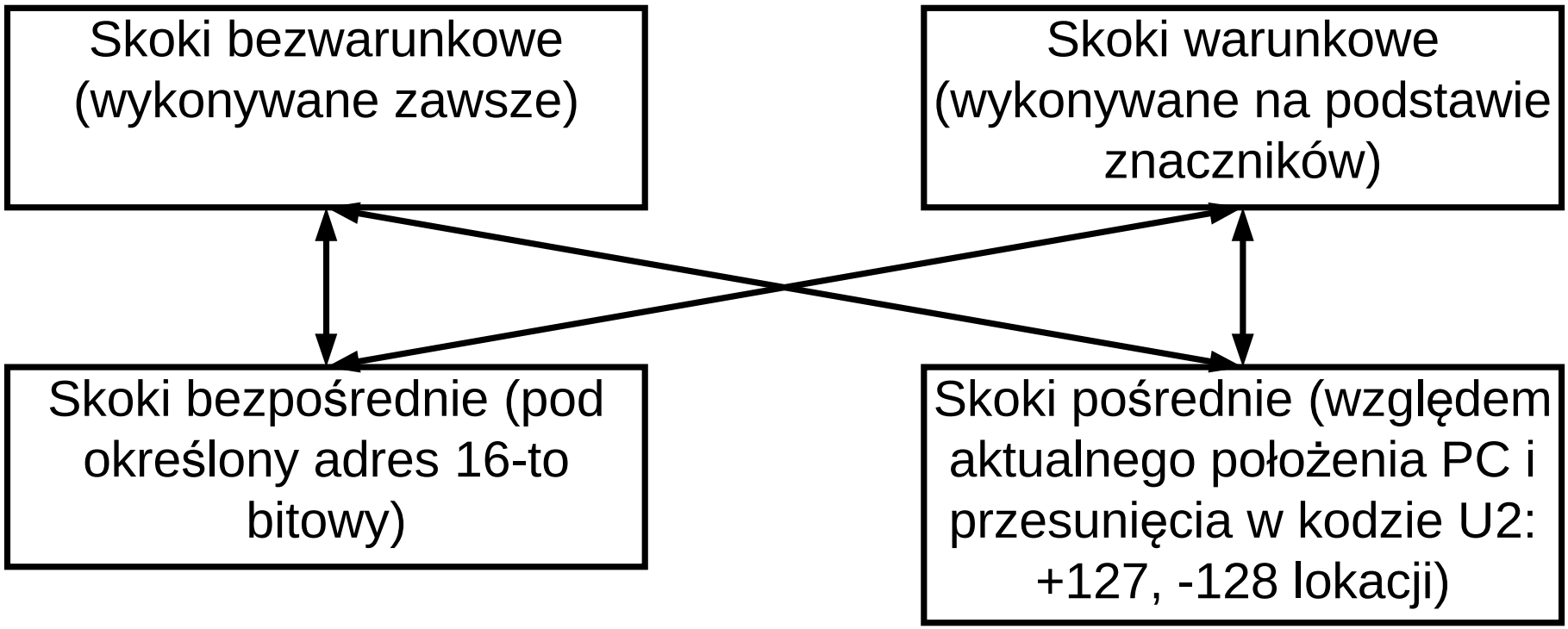
Bit Tested	b
0	000
1	001
2	010
3	011
4	100
5	101
6	110
1	111

M Cycles	T States	4 MHz E.T.
3	12 (4, 4, 4) 4	3.00

Wpisanie od znacznika Z zanegowanej wartości bitu o numerze b (0-7) w komórce pamięci o adresie zawartym w HL

Rozkazy skoków

Klasyfikacja rozkazów skoków



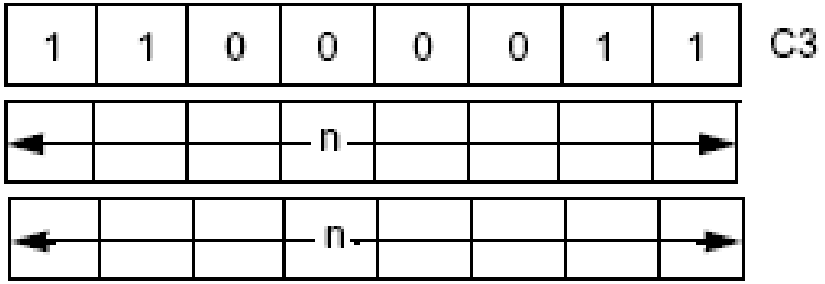
W Z80 występują wszystkie rodzaje skoków

JP nn

Operation: PC ← nn

Op Code: JP

Operands: nn



Note: The first operand in this assembled object code is the low order byte of a two-byte address.

Description: Operand nn is loaded to register pair PC (Program Counter). The next instruction is fetched from the location designated by the new contents of the PC.

M Cycles	T States	4 MHz E.T.
3	10 (4, 3, 3)	2.50

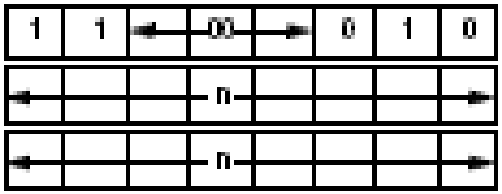
Skok bezwarunkowy, bezpośredni pod adres nn

JP cc, nn

Operation: IF cc true, PC ← nn

Op Code: JP

Operands: cc, nn



The first n operand in this assembled object code is the low-order byte of a 2-byte memory address.

Description: If condition cc is true, the instruction loads operand nn to register pair PC (Program Counter), and the program continues with the instruction beginning at address nn. If condition cc is false, the Program Counter is incremented as usual, and the program continues with the next sequential instruction. Condition cc is programmed as one of eight status that corresponds to condition bits in the Flag Register (register F). These eight status are defined in the table below that also specifies the corresponding cc bit fields in the assembled object code.

cc	Condition	Relevant Flag
000	NZ non zero	Z
001	Z zero	Z
010	NC no carry	C
011	C carry	C
100	PO parity odd	P/V
101	PE parity even	P/V
110	P sign positive	S
111	M sign negative	S

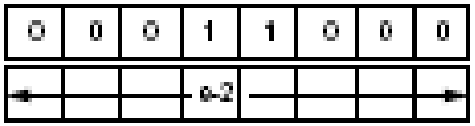
Skocz pod adres bezpośredni nn, jeśli warunek cc jest spełniony

JR e

Operation: PC ← PC + e

Op Code: JR

Operands: e



Description: This instruction provides for unconditional branching to other segments of a program. The value of the displacement e is added to the Program Counter (PC) and the next instruction is fetched from the location designated by the new contents of the PC. This jump is measured from the address of the instruction Op Code and has a range of -126 to +129 bytes. The assembler automatically adjusts for the twice incremented PC.

M Cycles	T States	4 MHz E.T.
3	12 (4, 3, 5)	3.00

Condition Bits Affected: None

Example: To jump forward five locations from address 480, the following assembly language statement is used: JR $+5

The resulting object code and final PC value is shown below:

Location	Instruction
480	18
481	03
482	-
483	-
484	-
485	← PC after jump

Skocz bezwarunkowo względem aktualnego PC o przesunięcia e (U2)

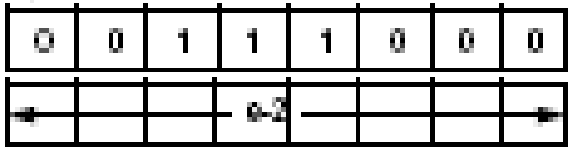
JR C, e

Operation: If C = 0, continue

If C = 1, PC ← PC+ e

Op Code: JR

Operands: C, e



Description: This instruction provides for conditional branching to other segments of a program depending on the results of a test on the Carry Flag. If the flag is equal to a 1, the value of the displacement e is added to the Program Counter (PC) and the next instruction is fetched from the location designated by the new contents of the PC. The jump is measured from the address of the instruction Op Code and has a range of -126 to +129 bytes. The assembler automatically adjusts for the twice incremented PC.

If the flag is equal to a 0, the next instruction executed is taken from the location following this instruction. If condition is met

M Cycles	T States	4 MHz E.T.
3	12 (4, 3, 5)	3.00

If condition is not met:

M Cycles	T States	4 MHz E.T.
2	7 (4, 3)	1.75

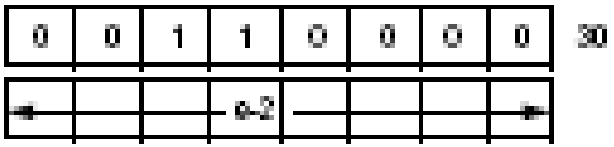
Skocz jeśli bit C=1 pod adres określony przez aktualne PC i offset e (U2)

JR NC, e

Operation: If C = 1, continue
If C = 0, PC ← PC + e

Op Code: JR

Operands: NC, e



Description: This instruction provides for conditional branching to other segments of a program depending on the results of a test on the Carry Flag. If the flag is equal to 0, the value of the displacement e is added to the Program Counter (PC) and the next instruction is fetched from the location designated by the new contents of the PC. The jump is measured from the address of the instruction Op Code and has a range of -126 to +129 bytes. The assembler automatically adjusts for the twice incremented PC.

If the flag is equal to a 1, the next instruction executed is taken from the location following this instruction.

If the condition is met:

M Cycles	T States	4 MHz E.T.
3	12 (4, 3, 5)	3.00

If the condition is not met:

M Cycles	T States	4 MHz E.T.
2	7 (4, 3)	1.75

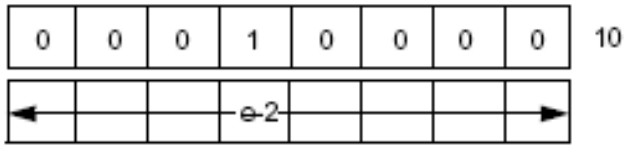
Skocz jeśli bit C=0 pod adres określony przez aktualne PC i offset e (U2)

DJNZ, e

Operation: -

Op Code: DJNZ

Operands: e



Description: This instruction is similar to the conditional jump instructions except that a register value is used to determine branching. The B register is decremented, and if a non zero value remains, the value of the displacement e is added to the Program Counter (PC). The next instruction is fetched from the location designated by the new contents of the PC. The jump is measured from the address of the instruction Op Code and has a range of -126 to +129 bytes. The assembler automatically adjusts for the twice incremented PC.

If the result of decrementing leaves B with a zero value, the next instruction executed is taken from the location following this instruction.

if B ≠ 0:

M Cycles	T States	4 MHz E.T.
3	13 (5,3, 5)	3.25

If B = 0:

M Cycles	T States	4 MHz E.T.
2	8 (5, 3)	2.00

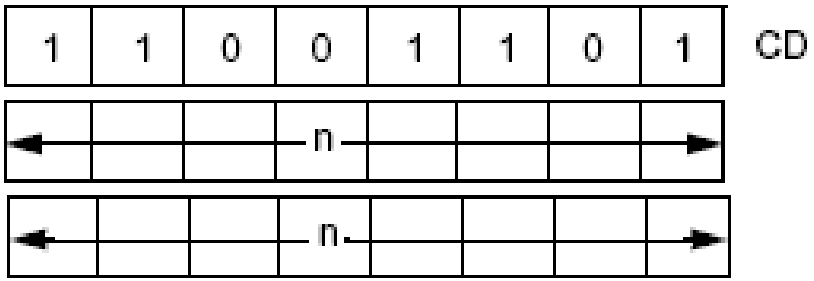
Zmniejsz zawartość rejestru B o1 i wykonaj skok względem PC z offsetem e, jeśli nie zero

CALL nn

Operation: (SP-1) ← PCH, (SP-2) ← PCL, PC ← nn

Op Code: CALL

Operands: nn



The first of the two n operands in the assembled object code above is the least-significant byte of a 2-byte memory address.

Wywołanie bezwarunkowe procedury o adresie bezpośrednim nn

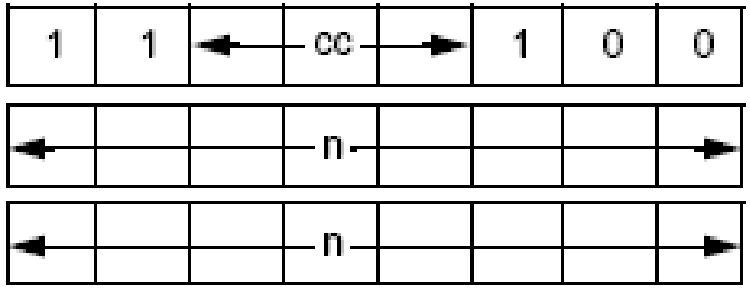
CALL cc, nn

Operation: IF cc true: (sp-1) ← PCH

(sp-2) ← PCL, pc ← nn

Op Code: CALL

Operands: cc, nn



Note: The first of the two n operands in the assembled object code above is the least-significant byte of the 2-byte memory address.

Wywołanie warunkowe procedury z adresem bezpośrednim

RET

Operation: pCL ← (sp), pCH ← (sp+1)

Op Code: RET

1	1	0	0	1	0	0	1	C9

Description: The byte at the memory location specified by the contents of the Stack Pointer (SP) register pair is moved to the low order eight bits of the Program Counter (PC). The SP is now incremented and the byte at the memory location specified by the new contents of this instruction is fetched from the memory location specified by the PC. This instruction is normally used to return to the main line program at the completion of a routine entered by a CALL instruction.

M Cycles	T States	4 MHz E.T.
3	10 (4, 3, 3)	2.50

Powrót bezwarunkowy z procedury

RET cc

Operation: If cc true: PCL ← (sp), pCH ← (sp+1)

Op Code: RET

Operands: cc

Powrót warunkowy z procedury

cc	Condition	Relevant Flag
000	NZ non zero	Z
001	Z zero	Z
010	NC non carry	C
011	C carry	C
100	PO parity odd	P/V
101	PE parity even	P/V
110	P sign positive	S
111	M sign negative	S

Warunki powrotu

RETI

Operation: Return from Interrupt

Op Code: RETI

1	1	1	0	1	1	0	1	ED
0	1	0	0	1	1	0	1	4D

Description: This instruction is used at the end of a maskable interrupt service routine to:

- Restore the contents of the Program Counter (PC) (analogous to the RET instruction)
- Signal an I/O device that the interrupt routine is completed. The `RETI` instruction also facilitates the nesting of interrupts, allowing higher priority devices to temporarily suspend service of lower priority service routines. However, this instruction does not enable interrupts that were disabled when the interrupt routine was entered. Before doing the `RETI` instruction, the enable interrupt instruction (`EI`) should be executed to allow recognition of interrupts after completion of the current service routine.

M Cycles	T States	4 MHz E.T.
4	14 (4, 4, 3, 3)	3.50

Powrót z przerwania maskowalnego

RETN

Operation: Return from non maskable interrupt

Op Code: RETN

1	1	1	0	1	1	0	1	ED
0	1	0	0	0	1	0	1	45

Description: This instruction is used at the end of a non-maskable interrupts service routine to restore the contents of the Program Counter (PC) (analogous to the RET instruction). The state of IFF2 is copied back to IFF1 so that maskable interrupts are enabled immediately following the RETN if they were enabled before the nonmaskable interrupt.

M Cycles	T States	4 MHz E.T.
4	14 (4, 4, 3, 3)	3.50

Powrót z przerwania nieaskowalnego, powoduje odtworzenie maski przerwań maskowalnych

Instrukcje wejścia-wyjścia

IN A, (n)

Operation: A ← (n)

Op Code: IN

Operands: A, (n)

Description: The operand n is placed on the bottom half (A0 through A7) of the address bus to select the I/O device at one of 256 possible ports. The contents of the Accumulator also appear on the top half (A8 through A15) of the address bus at this time. Then one byte from the selected port is placed on the data bus and written to the Accumulator (register A) in the CPU.

M Cycles	T States	4 MHz LT.
3	11 (4, 3, 4)	2.75

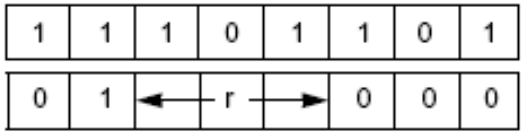
Odczyt na akumulator danej z urządzenia we-wy o adresie n

IN r (C)

Operation: r ← (C)

Op Code: IN

Operands: r, (C)



Description: The contents of register C are placed on the bottom half (A0 through A7) of the address bus to select the I/O device at one of 256 possible ports. The contents of Register B are placed on the top half (A8 through A15) of the address bus at this time. Then one byte from the selected port is placed on the data bus and written to register r in the CPU. Register r identifies any of the CPU registers shown in the following table, which also indicates the corresponding 3-bit r field for each. The flags are affected, checking the input data.

Register	r	
Flag	110	Undefined Op Code, set the flag
B	000	
C	001	
D	010	
E	011	
H	100	
L	101	
A	111	

M Cycles	T States	4 MHz E.T.
3	12 (4, 4, 4)	3.00

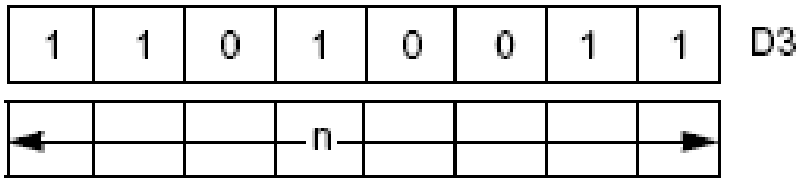
Odczyt do rejestru r danej z urządzenia we-wy o adresie zawartym w rejestrze C

OUT (n), A

Operation: (n) ← A

Op Code: OUT

Operands: (n), A



Description: The operand n is placed on the bottom half (A0 through A7) of the address bus to select the I/O device at one of 256 possible ports. The contents of the Accumulator (register A) also appear on the top half (A8 through A15) of the address bus at this time. Then the byte contained in the Accumulator is placed on the data bus and written to the selected peripheral device.

M Cycles	T States	4 MHz E.T.
3	11 (4, 3, 4)	2.75

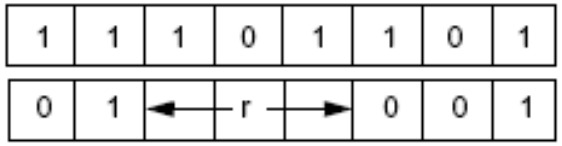
Wyślij zawartość akumulatora A do urządzenia we-wy o adresie n

OUT (C), r

Operation: (C) ← r

Op Code: OUT

Operands: (C), r



Description: The contents of register C are placed on the bottom half (A0 through A7) of the address bus to select the I/O device at one of 256 possible ports. The contents of Register B are placed on the top half (A8 through A15) of the address bus at this time. Then the byte contained in register r is placed on the data bus and written to the selected peripheral device. Register r identifies any of the CPU registers shown in the following table, which also shows the corresponding three-bit r field for each that appears in the assembled object code:

Register	r
B	000
C	001
D	010
E	011
H	100
L	101
A	111

M Cycles	T States	4 MHz E.T.
3	12 (4, 4, 4)	3.00

Wyślij zawartość rejestru r do urządzenia we-wy o adresie zawartym w C

Przesłania blokowe w obszarze pamięci

LDI

Operation: (DE) ← (HL), DE ← DE + 1, HL ← HL + 1, BC ← BC -1

Op Code: LDI

Operands: (SP), HL

1	1	1	0	1	1	0	1	ED
1	0	1	0	0	0	0	0	A0

Description: A byte of data is transferred from the memory location addressed, by the contents of the HL register pair to the memory location addressed by the contents of the DE register pair. Then both these register pairs are incremented and the BC (Byte Counter) register pair is decremented.

M Cycles	T States	4 MHz E.T.
4	16 (4, 4, 3, 5)	4.00

Condition Bits Affected:

LDIR

Operation: (DE) ← (HL), DE ← DE + 1, HL ← HL + 1, BC F↔ BC -1

Op Code: LDIR

Operands: B8

1	1	1	0	1	1	0	1	ED
1	0	1	1	0	0	0	0	B0

Description: This 2-byte instruction transfers a byte of data from the memory location addressed by the contents of the HL register pair to the memory location addressed by the DE register pair. Both these register pairs are incremented and the BC (Byte Counter) register pair is decremented. If decrementing causes the BC to go to zero, the instruction is terminated. If BC is not zero, the program counter is decremented by two and the instruction is repeated. Interrupts are recognized and two refresh cycles are executed after each data transfer. When BC is set to zero prior to instruction execution, the instruction loops through 64 Kbytes.

For BC ≠ 0:

M Cycles	T States	4 MHz E.T.
5	21 (4, 4, 3, 5, 5)	5.25

For BC = 0:

M Cycles	T States	4 MHz E.T.
4	16 (4, 4, 3, 5)	4.00