**DATA – adresowana bezpośrednio**

*ZMIENNA\_A DATA 40H*

*ORG 0*

*MOV A, ZMIANNA\_A*

**IDATA – adresowana pośrednio**

*MOV R0,#90H*

*MOV @R0,A*

*ZMIENNA\_B IDATA 90H*

*ORG 0*

*MOV R1,#ZMIANNA\_B*

*MOV A,@R1*

**XDATA – zewnętrzna pamięć danych**

*ZMIANNA\_C XDATA 2000H*

*ORG 0*

*MOV DPTR,ZMIANNA\_C*

*MOVX A ,@DPTR*

*ZMIANNA\_C XDATA 2000H*

*ORG 0*

*MOV DPTR,ZMIANNA\_C*

*MOVX A ,@DPTR*

*MOV B,A*

Linia programu składa się z:

* pole etykiety (nazwy) – nie wolno stosować nazw zastrzeżonych: rejestrów, pamięci, bitów, portów; długość jest nieograniczona

Etykieta:

\*definiuje „miejsce” (adres) w pamięci programu lub danych

\*jest unikalną nie będącą słowem kluczowym nazwą, którą można przypisać: poszczególnym punktom programu, podprogramom

\*może składać się z liter, cyfr i znaku podkreślenia

\*należy umieszczać w pierwszej linii tekstu programu, wskazane jest umieszczanie na końcu etykiety dwukropka

\*dana etykieta tylko raz, ale wiele odwołań

Nazwa:

\*nazwy reprezentują: stałe, adresy komórek pamięci, nazwy rejestrów

\*przyczyniają się do zwiększenia czytelności programu

* pole rozkazu (dyrektywy)

1.oryganizacyjne programu – ORG, END

2.definiujące symboliczne nazwy – EQU(nie redefiniowana), SET(redefiniowana), DATA(przypisuje ww (adres komórki wewnętrznej pamięci danych adresowanej bezpośrednio) nazwę), IDATA, XDATA

3.inicjujące i rezerwujące (pamięć programu) – DB (związana z CODE)

*ORG 0*

*START: DB 10,3BH,10010111B*

*END*

* pole argumentów – argumentami mogą być:

1.identyfikatory (etykiety, nazwy) – jeśli występuje przed instrukcją generującą kod wynikowy to jego wartością będzie adres 1-go bajtu instrukcji, którą poprzedza

2.stałe

a)liczbowe – muszą być poprzedzone # i mogą być podane w systemie: dziesiętnym (D, d, bez niczego), binarnym, szesnastkowym, ósemkowym; musi zaczynać się od liczby #0A5H (dobrze); #A5H (źle); tylko liczby całkowite

b)tekstowe – łańcuch znaków w cudzysłowie (‘Ala’, ‘ab’)

3.nazwy zastrzeżone

4.adresy komórek pamięci

* pole komentarza (zaczyna się od średnika, może wystąpić w dowolnej części linii programu)

[Etykieta: ] Dyrektywa

[operand] [,operand] [komentarz]

[Nazwa] Mnemonik `51

Monitor:

1. umożliwia przyjęcie programu wynikowego z komputera i umieszczenie go w pamięci RAM

2. umożliwia uruchomienie przesłanego programu z komputera

3. umożliwia uruchomienie programu w pracy krokowej

4. umożliwia wstawianie pułapek programowych

5. umożliwia podgląd i modyfikację zawartości komórek pamięci i rejestrów

Program źródłowy – program napisany w języku asemblera

Asembler (kompilator) – tłumaczy na program pośredni .OBJ

Linker – program łącząco – ładujący 🡪 program wynikowy .HEX

IDE `51 – służy do stworzenia programu źródłowego i przekształcenia go do programu wynikowego

Narzędzia w IDE `51: edytor, asembler, linker

256 kB wewnętrznej pamięci danych; DPTR 🡪 16 bitowy, DPH i DPL;

P0 i P2 do obsługi zewnętrznej pamięci 🡪 8 bitowy 🡪 adres 16 bitowy

XDATA – 64 kB; CODE – 64 kB

Cykl maszynowy – parametr określający czas trwania instrukcji (1, 2, 4)

tc

tp1

tp2

P2

S1

S2

S3

S4

S5

S6

P1

tp1 = tp2 = Tosc

tc = 12 \* Tosc = 1µs

**Opóźnienie 1**

*MOV R3,#WARTOSC ;1c*

*SKOK: DJNZ R3,SKOK ;2c*

Nc = 1+2\*WARTOSC

R3 – licznik pętli

WARTOSC – stała pętli

WARTOSC = 255 🡪 Nc = 1+2\*255 = 511

WARTOSC = 0 🡪 Nc = 1+2\*256 = 513

**Opóźnienie 2**

*MOV R4,#WARTOSC\_2*

*SKOK\_2: MOV R3,#WARTOSC\_1*

*SKOK\_1: DJNZ R3,SKOK\_1*

*DJNZ R4,SKOK\_2*

R3 – licznik pętli wewnętrznej

R4 – licznik pętli zewnętrznej

Pętla wewnętrzna:

SKOK\_1 ….

Pętla zewnętrzna:

SKOK\_2 ….

SKOK\_1 ….

DJNZ R4, SKOK\_2

Nc = 1 + (1 + 2 \* WARTOSC\_1 + 2) \* WARTOSC\_2

WARTOSC\_1 = 0

WARTOSC\_2 = 194

Nc = 1 + (1 + 2 \*256 + 2) \* 194 = 99911 ≈ 0,1µs

**IDATA 🡪 DATA**

NAZWA\_1 DATA 20H

NAZWA\_2 IDATA 90H

ORG 0

MOV R1,#NAZWA\_2

MOV A,@R1

MOV NAZWA\_1,A

END

**DATA🡪IDATA**

NAZWA\_1 DATA20H

NAZWA\_2 IDATA 90H

ORG 0

MOV A, NAZWA\_1

MOV R1,#NAZWA\_2

MOV@R1,A

END

**DATA🡪XDATA**

NAZWA\_1 DATA 20H

NAZWA\_2 XDATA 2000H

ORG 0

MOV A, NAZWA\_1

MOV DPTR, #NAZWA\_2

MOVX @DPTR,A

END

**XDATA🡪DATA**

NAZWA\_1 DATA 20H

NAZWA\_2 XDATA 2000H

ORG 0

MOV DPTR, #NAZWA\_2

MOVX A,@DPTR

MOV NAZWA\_1,A

END

**IDATA🡪XDATA**

NAZWA\_1 IDATA 30H

NAZWA\_2 XDATA 2000H

ORG 0

MOV RO,#NAZWA\_1

MOV A,@RO

MOV DPTR, #NAZWA\_2

MOVX @DPTR,A

END

**XDATA🡪IDATA**

NAZWA\_1 XDATA 2000H

NAZWA\_2 IDATA 20H

ORG 0

MOV DPTR,#NAZWA\_1

MOVX A, @DPTR

MOV R1,#NAZWA\_2

MOV @R1,A

END

|  |  |  |  |
| --- | --- | --- | --- |
| **MNEMONIK** | **OPERACJA** | **BAJTY/**  **CYKLE** | **UWAGI** |
| **MOV A,***Rr* | **A** ← *Rr* | **1/1** | *Rr -* rejestr R0 ÷ R7 |
| **MOV A,***ad* | **A** ←*<ad>* | **2/1** | *ad -* bezporedni adres 8 - bitowy |
| **MOV A,@***Ri* | **A** ←*<Ri>* | **1/1** | *Ri -* rejestr R0 ÷ R1 |
| **MOV A,#***n* | **A** ← *n* | **2/1** | *n -* 8 - bitowy argument  bezporedni |
| **MOV** *Rr,***A** | *Rr* ← **A** | **1/1** | *Rr -* rejestr R0 ÷ R7 |
| **MOV** *Rr,ad* | *Rr* ←<*ad>* | **2/2** | *Rr -* rejestr R0 ÷ R7  *ad -* bezpor. adres 8-bitowy |
| **MOV** *Rr,*#*n* | *Rr* ← *n* | **2/1** | *Rr -* rejestr R0 ÷ R7  *n -* 8-bitowy argument bezp. |
| **MOV** *ad,***A** | *<ad>*← **A** | **2/1** | *ad -* bezporedni adres 8 - bitowy |
| **MOV** *ad,Rr* | *<ad>*← *Rr* | **2/2** | *ad -* bezpor. adres 8-bitowy  *Rr -* rejestr R0 ÷ R7 |
| **MOV** *ad1,ad2* | *<a1>*←*<a2>* | **3/2** | *ad1,ad2 -* 8-bitowe adresy bezp. |
| **MOV** *ad*,**@***Ri* | *<ad>*←*<Ri>* | **2/2** | *ad -* bezpor. adres 8- bitowy  *Ri -* rejestr R0 ÷ R1 |
| **MOV** *ad,*#*n* | *<ad>* ← *n* | **3/2** | *ad -* bezpor. adres 8 - bitowy  *n -* 8 - bitowy argument bezp. |
| **MOV @***Ri,***A** | *<Ri>*← **A** | **1/1** | *Ri -* rejestr R0 ÷ R1 |
| **MOV @***Ri,ad* | *<Ri>*←*<ad>* | **2/2** | *Ri -* rejestr R0 ÷ R1  *ad -* bezpor. adres 8-bitowy |
| **MOV @***Ri,***#***n* | *<Ri>* ← *n* | **2/1** | *Ri -* rejestr R0 ÷ R1  *n -* 8 - bitowy argument bezp. |
| **MOV DPTR,#***nn* | **DPTR** ← *nn* | **3/2** | *nn -* 16 - bitowy argument  bezporedni |
| **XCH A,***Rr* | **A** ↔ *Rr* | **1/1** | *Rr -* rejestr R0 ÷ R7 |
| **XCH A,***ad* | **A** ↔ *<ad>* | **2/1** | *ad -* bezpor. adres 8-bitowy |
| **XCH A,@***Ri* | **A** ↔ *<Ri>* | **1/1** | *Ri -* rejestr R0 ÷ R1 |
| **XCHD A,@***Ri* | **A**3-0↔*<Ri>*3-0 | **1/1** | *Ri* 3-0 - modsza cz rejestru *Ri* |
| **MOVX A,@***Ri* | **A** ←*<Ri>* | **1/2** | *Ri -* rejestr R0 ÷ R1 , zawiera adres 8 - bitowy |
| **MOVX @***Ri*,**A** | *<Ri>*← **A** | **1/2** | *Ri -* rejestr R0 ÷ R1 , zawiera adres 8 - bitowy |
| **MOVX A,@DPTR** | **A** ← **(DPTR)** | **1/2** | DPTR - zawiera adres 16-bitowy |
| **MOVX @DPTR,A** | **(DPTR)** ← A | **1/2** | DPTR - zawiera adres 16-bitowy |
| **MOVC A,@A+DPTR** | **A**←(**A+DPTR)** | **1/2** | DPTR - 16- bitowy wskanik  danych |
| **MOVC A,@A+PC** | **A**←(**A + PC)** | **1/2** | PC - 16 - bitowy licznik rozkazw |