**A, AC** – akumulator

**B** – rejestr pomocniczy B

**Rr** – rejestr roboczy, r = 0÷7

**Ri** – rejestr roboczy, wskaźnik danych, i = 0,1

**DPTR** – wskaźnik danych

**PC** – licznik rozkazów

**SP** – wskaźnik stosu

**C, CY** – znacznik przeniesienia

**@** - w mnemorniku oznacza adres pośredni

**#** - w mnemorniku oznacza adres bezpośredni

**ad** – 8 bitowy adres bezpośredni

**n** – 8 bitowy argument bezpośredni

**nn** – 16 bitowy argument bezpośredni

**bit** – 8 bitowy adres bitu w RAM lub SFR

**adr11** – adres 11 bitowy

**adr16** – adres 16 bitowy

**d** – 8 bitowe przesunięcie (-128, 127)

**X** – w zapisie operacji oznacza zawartość rejestru X

**(X)** – w zapisie operacji oznacza zawartość rejestru o adresie X

**MOV TL0,#low(4095)** do TL0 zostanie wpisany młodszy bajt liczby 4095 (czyli 0ffh);

**MOV TH0,#high(4095)** do TH0 zostanie wpisany starszy bajt liczby 4095 (czyli 0fh);

**MOV TL1,#low(not(4095)** do TL1 zostanie wpisany zanegowany młodszy bajt liczby 4095 (czyli 00h);

**MOV TH1,#high(not(4095))**do TH1 zostanie wpisany zanegowany starszy bajt liczby 4095 (czyli 0f0h).

**EQU**

**stała EQU 31h** (przypisanie liczby 31h nazwie *stała*)

**SET**

**dana SET 10h**

(linie programu)

**dana SET 20h** (przypisanie nazwie *dana* kolejno dwóch, różnych wartości)

**DATA**

**ilość DATA 21h (**przypisanie nazwie *ilość* adresu komórki pa-mięci z **wewnętrznego** obszaru pamięci danych adresowanej bezpośrednio 00-0ffh)

**IDATA**

**dane IDATA 81h** (przypisanie nazwie *dane* adresu komórki pa-mięci z **wewnętrznego** obszaru pamięci danych adresowanej pośrednio 00-0ffh)

**XDATA**

**lcd XDATA 1fffh** (przypisanie nazwie *lcd* adresu komórki pamięci z **zewnętrznego** obszaru pamięci danych 0000-0ffffh)

**CODE**

**start CODE 0100h** (przypisanie nazwie *start* adresu komórki pa-mięci z obszaru **pamięci programu** 0000-0fffh)

**BIT**

**stop BIT 00** (przypisanie nazwie *stop* adresu bitu z obszaru **wewnętrznej** pamięci, adresowanej bitowo: bity 00-7fh w obszarze pamięci 20h-30h oraz bity rejestrów specjalnych w obszarze 80h-0ffh)

**DB**

**tab: DB 10,1fh,10101010b,’ABC’** (umieszcza kolejno w pamięci programu bajty wyrażeń)

**DW**

**tab1: DW 1234,1f1fh,’abcd’** (umieszcza w pamięci programu kolejno podane słowa (po 2 bajty) wyrażenia)

**ORG**

**ORG 1000h** (instrukcje są umieszczane w pamięci programu kolejno począwszy od adresu podanego dyrektywą **ORG)**

**END**

linie programu po tej dyrektywie nie są przez kompilator analizowane.

**MOV** - kopiuj.

Działanie: wpisuje do pierwszego operandu zawartość drugiego.

Dozwolone argumenty: **A,Rr    A,direct    A,@Ri    A,#data    Rr,A    Rr,direct**(2)**Rr,#data    direct,A   direct,Rr**(2)**direct,direct**(2)**direct,@Ri**(2)**direct,#data**(2)**@Ri,A    @Ri,direct**(2)**@Ri,data    DPTR,#data16**(2).

**MOVC**(2) – kopiuj z pamięci programu.

Działanie: kopiuje bajt kodu (zawartości pamięci programu).

Dozwolone argumenty: **A,@A+DPTR    A,@A+PC**.

**MOVX**(2) - kopiuj z/do zewnętrznej pamięci danych.

Działanie: kopiuje bajt z/do komórki zewnętrznej pamięci danych wskazanej rejestrem indeksowym.

Dozwolone argumenty: **A,@Ri    A,@DPTR    @Ri,A    @DPTR,A**.

**XCH** - zamień.

Działanie: wymienia między sobą zawartość dwóch rejestrów.

Dozwolone argumenty: **A,Rr    A,direct    A,@Ri**.

**XCHD** - zamień młodsze 4 bity.

Działanie: wymienia między dwoma rejestrami zawartość ich 4 młodszych bitów.

Dozwolone argumenty: **A,@Ri**.

**CLR** - zerowanie.

Działanie: zeruje podany bit.

Dozwolone argumenty: **C    bit**.

**SETB** - ustawienie bitu.

Działanie: przypisuje danemu bitowi wartość 1.

Dozwolone argumenty: **C    bit**.

**CPL** - negacja.

Działanie: neguje podany bit.

Dozwolone argumenty: **C    bit**.

**ANL**(2) - iloczyn logiczny AND.

Działanie: wykonuje iloczyn logiczny dwóch bitów, wynik zapisuje w pierwszym.

Dozwolone argumenty: **C,bit    C,/bit**.

**ORL**(2) - suma logiczna OR.

Działanie: wykonuje sumę logiczną dwóch bitów, wynik zapisuje w pierwszym.

Dozwolone argumenty: **C,bit    C,/bit**.

**MOV** - kopiuj.

Działanie: wpisuje do pierwszego bitu wartość zapisaną w drugim.

Dozwolone argumenty: **C,bit    bit,C**(2).

**ANL - iloczyn logiczny AND.**

Działanie: wykonuje logiczny iloczyn na odpowiednich bitach argumentów, wynik zapisuje do pierwszego argumentu.

Dozwolone argumenty: **A,Rr    A,direct    A,@Ri    A,#data    direct,A    direct,#data**(2)

**ORL - suma logiczna OR.**

Działanie: wykonuje sumę logiczną odpowiednich bitów argumentów, wynik zapisuje do pierwszego argumentu.

Dozwolone argumenty: **A,Rr    A,direct    A,@Ri    A,#data    direct,A    direct,#data**(2)

**XRL** - suma mod 2 (XOR).

Działanie: wykonuje sumowanie mod 2 odpowiednich bitów argumentu, wynik zapisuje do pierwszego z nich.

Dozwolone argumenty: **A,Rr    A,direct    A,@Ri    A,#data    direct,A    direct,#data**(2)

**CLR** - zerowanie (bajtowe).

Działanie: zeruje wszystkie bity argumentu.

Dozwolone argumenty: **A**.

**CPL** - negowanie (bajtowe).

Działanie: neguje wszystkie bity argumentu.

Dozwolone argumenty: **A**.

**SWAP** - zamiana 4 bitów bajtu.

Działanie: zamienia miejscami 4 starsze i 4 młodsze bity bajtu.

Dozwolone argumenty: **A**.

**RL** - obrót w lewo.

Działanie: przesuwa w lewo zawartość akumulatora.

Dozwolone argumenty: **A**.

**RR** - obrót w prawo.

Działanie: przesuwa w prawo zawartość akumulatora.

Dozwolone argumenty: **A**.

**RLC** - obrót w lewo przez C.

Działanie: przesuwa w lewo zawartość akumulatora, na miejsce najmłodszego bitu wpisywana jest dotychczasowa zawartość bitu C, bit najstarszy jest wpisywany jako nowa wartość C.

Dozwolone argumenty: **A**.

**RRC** - obrót w prawo przez C.

Działanie: przesuwa w prawo zawartość akumulatora, na miejsce najstarszego bitu wpisywana jest dotychczasowa zawartość bitu C, bit najmłodszy jest wpisywany jako nowa wartość C.

Dozwolone argumenty: **A**.

**RET** - powrót z procedury.

Działanie: pobiera ze stosu zawartość licznika rozkazów i przekazuje sterowanie do znajdującego się tam adresu.

**RETI** - powrót z procedury obsługi przerwania.

Działanie: pobiera ze stosu zawartość licznika rozkazów i przekazuje sterowanie do znajdującego się tam adresu.

**PUSH**(2) - przesłanie bajtu na stos.

Działanie: inkrementuje zawartość rejestru SP kopiuje zawartość komórki pod adres wskazywany przez SP.

Dozwolone argumenty: **direct**.

**POP**(2) - pobranie baju ze stosu.

Działanie: kopiuje zawartość komórki wskazywanej przez SP do komórki podanej w wywołaniu dekrementuje zawartość SP.

Dozwolone argumenty: **direct**.

**ACALL** - wywołanie podprogramu w stronie 2kB.

Działanie: zapisuje na stos zawartość PC i wykonuje skok do adresu początku procedury.

Dozwolone argumenty: **addr11**.

**LCALL** - wywołanie podprogramu.

Działanie: zapisuje na stos zawartość PC i wykonuje skok do adresu początku procedury.

Dozwolone argumenty: **addr16**.

**ADD - dodawanie.**

*Działanie: A = A + dana.*

Dozwolone argumenty: **A,Rr    A,direct     A,@Ri    A,#data**

Modyfikowane znaczniki: **C**, **AC**, **OV**.

**ADDC - dodawanie z przeniesieniem.**

*Działanie: A = A + dana + C.*

Dozwolone argumenty: **A,Rr    A,direct    A,@Ri    A,#data**

Modyfikowane znaczniki: **C**, **AC**, **OV**.

**SUBB - odejmowanie z pożyczką.**

*Działanie: A = A – dana - C.*

Dozwolone argumenty: **A,Rr    A,direct    A,@Ri    A,#data**

Modyfikowane znaczniki: **C**, **AC**, **OV**.

**INC - inkrementacja.**

*Działanie: argument = argument t+1*

Dozwolone argumenty: **A    Rr    direct    @Ri    DPTR**(2)

**DEC - dekrementacja.**

*Działanie: argument=argument-1.*

Dozwolone argumenty: **A    Rr    direct    @Ri**

**MUL - mnożenie liczb 8-bitowych, wynik 16-bitowy.**

*Działanie: A\*B, do A zapisywana młodsza część wyniku, do B starsza.*

Dozwolone argumenty: **AB**

Modyfikowane znaczniki: **OV**, bit **C** zerowany.

**DIV - dzielenie liczb 8-bitowych, 8-bitowy wynik i 8-bitowa reszta.**

*Działanie: A/B, A - wynik, B - reszta.*

Dozwolone argumenty: **AB**

Modyfikowane znaczniki: **OV**, bit **C** zerowany.

**DA - poprawka dziesiętna.**

*Działanie: wykonuje poprawkę dziesiętną liczby 8-bitowej, przekształcając ją do postaci 2 cyfr w kodzie BCD.*

Dozwolone argumenty: **A**

Modyfikowane znaczniki: **C**.

**LJMP** - rozkaz skoku w obszarze 64kB.

Działanie: wpisuje do licznika rozkazów podany adres.

Dozwolone argumenty: **addr16**.

**AJMP** - rozkaz skoku w stronie 2kB.

Działanie: zastępuje 11 młodszych bitów PC wartością podaną w wywołaniu.

Dozwolone argumenty: **addr11**.

**SJMP** - rozkaz skoku w obszarze 256B.

Działanie: następuje zmiana zawartości PC o wartość w kodzie U2 podaną w wywołaniu.

Dozwolone argumenty: **rel**.

**JMP** - rozkaz skoku w trybie pośrednim.

Działanie: przepisuje do PC wartość będącą sumą zawartości rejestru indeksowego i bazowego.

Dozwolone argumenty: **@A+DPTR**.

**JZ** - skok warunkowy.

Działanie: wykonuje skok, jeżeli zawartość akumulatora równa jest 0.

Dozwolone argumenty: **rel**.

**JNZ** - skok warunkowy.

Działanie: wykonuje skok, jeżeli zawartość akumulatora jest różna od 0.

Dozwolone argumenty: **rel**.

**JC** - skok warunkowy.

Działanie: wykonuje skok, jeżeli bit przeniesienia C jest ustawiony.

Dozwolone argumenty: **rel**.

**JNC** - skok warunkowy.

Działanie: wykonuje skok, jeżeli bit przeniesienia C jest wyzerowany.

Dozwolone argumenty: **rel**.

**JB** - skok warunkowy.

Działanie: wykonuje skok, jeżeli dany bit jest ustawiony.

Dozwolone argumenty: **bit,rel**.

**JNB** - skok warunkowy.

Działanie: wykonuje skok, jeżeli dany bit jest wyzerowany.

Dozwolone argumenty: **bit,rel**.

**JBC** - skok warunkowy.

Działanie: gdy dany bit jest ustawiony, to nastąpi skok i wyzerowanie bitu.

Dozwolone argumenty: **bit,rel**.

**CJNE** - skok warunkowy, zależny od wyniku porównania.

Działanie: następuje porównanie operandów, jeżeli są różne, to następuje skok.

Dozwolone argumenty: **A,direct,rel    A,#data,rel    Rr,#data,rel    @Ri,#data,rel**.

Modyfikowane znaczniki: **C** (ustawiany, jeżeli pierwszy operand mniejszy od drugiego).

**DJNZ** - skok warunkowy, zależny od wyniku dekrementacji.

Działanie: następuje dekrementacja zawartości operandu, jeżeli zawartość nie osiągnęła zera, to nastąpi skok.

Dozwolone argumenty: **Rr,rel    direct,rel**.