Transferbefehle

Befehl	Funktion
MOV [Ziel], [Quelle]	Kopie, bezogen auf Register und internen Datenbereich
MOVX [Ziel], [Quelle]	Kopiert Daten zwischen Akku und externem Speicher
MOVC [Ziel], [Quelle]	Kopiert ein Byte von Code oder Programmspeicher in den Akku
XCH A, [Operand]	Tauscht die Bytes zwischen Akku und dem Operanden aus.
XCHD A, [Rx]	Tauscht das niederwertige Nibble des Akkus mit dem niederwertigen Nibble des indirekten RAM
CLR [Ziel]	Setzt das angegebene Ziel auf 0
SWAP A	Tauscht das niederwertige mit dem höherwertigen Nibble des Akkus aus
PUSH [Quelle]	Inkrementiert den Stack Pointer und speichert den Wert der Quelle an der entsprechenden Stelle im RAM
POP [Ziel]	Speichert den Wert an der Stelle des SP im angegebenen Ziel

unbedingte Sprungbefehle

Befehl	Funktion
AJMP [Label]	Springt innerhalb des aktuellen 2 KByte Blocks. Befehl wird in 2 Buszyklen abgearbeitet.
LJMP [Label]	Springt innerhalb des kompletten 64 KByte Blocks. Befehl wird in 2 Buszyklen abgearbeitet.
SJMP [Label]	Springt um -128 Bytes oder +127 Bytes.
JMP @A+DPTR	Im DPTR kann die Grundadresse einer Sprungtabelle gespeichert werden. im Akkumulator der Offset innerhalb der Tabelle

Bedingte Sprungbefehle

Abhängig von Testbit

Befehl	Funktion
JB [Testbit] [Label]	Springt zum angegebenen Ziel, wenn das Testbit = 1.

Bedingte Sprungbefehle (cont)

JNB [Testbit] [Label]	Springt zum angegebenen Ziel, wenn das Testbit ungleich 1.
JNC [Testbit] [Label]	Springt zum angegebenen Ziel, wenn das Testbit = 1. **Löscht das Testbit!**

Abhängig von Akkumulatorinhalt

Befehl	Funktion
JZ [Label]	Springt, wenn Akkumulatorinhalt = 0
JNZ [Label]	Springt, wenn Akkumulatorinhalt ungleich 0

Abhängig von Carry Flag

Befehl	Funktion
JC [Label]	Springt, wenn Carry Flag = 1
JNC [Label]	Springt, wenn Carry Flag ungleich 0

Vergleichen und Manipulieren

DJNZ [Register], [Label]	Dekrementiert das Register und springt, wenn Ergebnis ungleich 0
DJNZ [direkt adr. Speicherstelle] [Label]	Dekrementiert die Speicherstelle und springt, wenn Ergebnis ungleich 0

Vergleich von 2 Speicherstellen

CJNE A, [direkt adr. RAM-Inhalt], [Label]	Vergleicht Akkumulatorinhalt mit direktem RAM-Inhalt und springt bei Ungleichheit
CJNE A, [Konstante], [Label]	Vergleicht Akkumulatorinhalt mit der Konstanten und springt bei Ungleichheit.
CJNE [Register], [Konstante], [Label]	Vergleicht Registerinhalt mit der Konstanten und springt bei Ungleichheit.
CJNE [@Register], [Konstante], [Label]	Vergleicht indirekten RAM-Inhalt mit der Konstanten und springt bei Ungleichheit.

By **TimSch**
cheatography.com/timsch/

Published 13th November, 2016.
Last updated 16th November, 2016.

Bitverarbeitungsbefehle

Unäre Befehle

Befehl	Funktion
CLR [Ziel]	Löscht das angegebene Bit (setzt es auf 0)
SETB [Ziel]	Setzt das angegebene Bit. (1)
CPL [Ziel]	Komplementiert das angegebene Bit

Befehle für 2 Operanden

Befehl	Funktion
ANL [Ziel] [2. Operand]	Logisches UND; Vergleicht die beiden Operanden und speichert das Ergebnis im Zieloperanden
ORL [Ziel] [2. Operand]	Logisches ODER; Vergleicht die beiden Operanden und speichert das Ergebnis im Zieloperanden
MOV [Ziel], [Quelle]	Kopiert das angegebene Bit an die Zieladresse

Rotationsbefehle

Befehl	Funktion
RL A	Rotationsbefehl links ohne Carry *Bit 7 -> Bit 0; Bit 0 -> Bit 1*
RLC A	Rotationsbefehl links mit Carry *Bit 7 -> Carry; Carry -> Bit 1*
RR A	Rotationsbefehl rechts ohne Carry *Bit 0 -> Bit 7; Bit 7 -> Bit 6*
RRC A	Rotationsbefehl rechts mit Carry *Bit 0 -> Carry; Carry -> Bit 7*

Arithmetische Verarbeitungsbefehle

Befehl	Funktion
ADD A, [Operand]	Addiert den Wert des Operanden zum Akkumulator ohne Berücksichtigung des Carrys
ADDC A, [Operand]	Addiert den Wert des Operanden zum Akkumulator mit Berücksichtigung des Carrys
SUBB A, [Operand]	Subtrahiert den Wert des Operanden und den Carry vom Operanden. *Für Subtraktion ohne Borrow vorher Carry löschen.*

Arithmetische Verarbeitungsbefehle (cont)

MUL AB	Multipliziert Wert des Akkumulators mit dem Wert des Hilfsakkumulators B. *Das High-Byte wird in B gespeichert, das Low-Byte wird in A gespeichert.*
DIV AB	Dividiert Wert des Akkumulators mit dem Wert des Hilfsakkumulators B. *Legt das 8-Bit Ergebnis in A und den Rest in B ab.* ***Overflow bei Div/0***
INC [Operand]	Inkrementiert den angegebenen Operanden
DEC [Operand]	Dekrementiert den angegebenen Operanden.
DA A	Korrigiert das Ergebnis der Addition zweier BCD-Zahlen.

Unterprogrammbefehle

Sprungbefehle

LCALL [Label]	Ruft Unterprogramme innerhalb des aktuellen 2 KByte Blocks auf
ACALL [Label]	Ruft Unterprogramme innerhalb des gesamten 64 KByte Adressraums auf.

Rückkehrbefehle

RET	Rückkehr aus dem Unterprogramm
RETI	Rückker von Interrupt-Service-Routinen

Sonstige Befehle

Befehl	Funktion
NOP	No Operation. Dauert 1 Taktzyklus, macht nichts.

Assemblerdirektiven

Direktive	Funktion
END	Markiert das Ende des Quelltextes
ORG	Definiert die genaue Position der auf diese Anweisung folgenden Instruktionen im Programmspeicher.
$INCLUDE [Dateiname]	Bindet Textdateien an der Stelle der Anweisung ein