3. tétel

a. Feltétel nélküli vezérlésátadás, a processzor utasításrendszere, operandusok, címzési módok

Feltétel nélküli vezérlésátadás:

Feltétel nélküli ugrásnál az utasításban szereplő címmel tölti fel a processzor az utasításszámláló regiszter tartalmát, amely a következő utasítás címe lesz és a program innen folytatódik.

Assembly-ben: JMP címke

- nem elegáns használni
  - több nyelvben már nem is létezik ez az utasítás
  - vagy csupán végtelen ciklusból való kiugrásra lehetséges a használata
- a probléma vele többrétű
  - átláthatatlanná teheti a programstruktúrát, hibakeresést megnehezíti
  - a fordítóprogram nem tudja miatta jól meghatározni a program erőforrás- vagy memóriaigényét
- kivételkezelés programnyelvi támogatása kiválthatja a szükségességét
- FORTRAN0 vezette be, COBOL támogatja
- Pascalban címkére ugorhat blokkon belül, C-ben címkére ugorhat függvényen belül
- Modula-3-ban nincs GOTO (de van RETURN visszatérni eljárásokból, függvényekből), Java-ban nincs GOTO

Processzor utasításrendszer:

Minden számítógép – a Neumann-elvből következik – tartalmaz egy belső (operatív vagy rendszer) memóriát az éppen futó programok (utasítások) és az adatok tárolására. Az Intel processzorok memóriája byte szervezésű, azaz minden byte-nyi memóriának van egy memória címe. Bármely két szomszédos byte egy 16 bites szót alkot.

A gépi kódú utasítások négy részből állnak, méretük függ az utasítástól és a címzési módtól.

Az általános utasításszerkezet:

Prefixum | Operáció kód | Címzési mód | Operandus

1. A prefixum módosítja az utasítás értelmezését, pl. ezzel írható elő az ismétlések száma, vagy a megszakításkérés tiltása. Használata nem kötelező.
2. Az operáció kód adja meg, hogy a processzornak milyen műveletet kell végrehajtania. Használata kötelező.
3. A címzési mód az operandusok értelmezését adja meg. Nem minden utasításban található meg.
4. Az operandus lehet konstans, cím vagy címzéshez használt érték. Hossza változó, használata nem kötelező.

Az assembly utasítások pontosan megfelelnek egy gépi kódú utasításnak, azaz minden assembly utasításból egy gépi kód lesz.

Operandusok

Az Intel processzorokban egy utasítással csak egy memóriahely címezhető meg. Azaz, ha az utasítás két operandusú (pl.: MOV), akkor csak az egyik lehet memóriacím, a másiknak regiszternek kell lennie.

Vannak olyan utasítások, amelyeknek nincs operandusuk, vagy maga az utasítás egyértelműen (implicit) meghatározza azokat. (pl.: RET – nincs operandus, CBW – AL-ben lévő értéket AX-be konvertálja, implicit)

Regiszteroperandus

Az utasítás paraméterei regiszterek, amelyekben a művelethez szükséges adatokat tároljuk. Az utasítás hossza a regisztermérettől függ (8, 16, 32 bit).

Pl.: MOV AX, BX

Az utasítások egy részénél csak meghatározott regiszterek használhatók.

Közvetlen (immediate) operandus

Ilyenkor az operandust az utasítás kódja tartalmazza. Pl.: MOV AX,2

Direkt memóriacímzés

Itt az operandus egy, a Data szegmensben előre deklarált változó.

```
.DATA
        adat DB "A"
.CODE
        MOV AX, adat
```

Indirekt memóriacímzés

Itt az operandus az adat címét tartalmazza, nem az értékét.

Regiszter indirekt címzés

```
Pl.: MOV AX, [BX]          ;AX-be tölti a BX által megcímzett memória tartalmát
```

Indexelt (bázis-relatív) címzés

```
.DATA
        adat DB 10h, 20h
.CODE
        MOV AX, adat[BX]   ;AX-be tölti az adat+BX címen lévő adatot
```

Bázis plusz index címzés

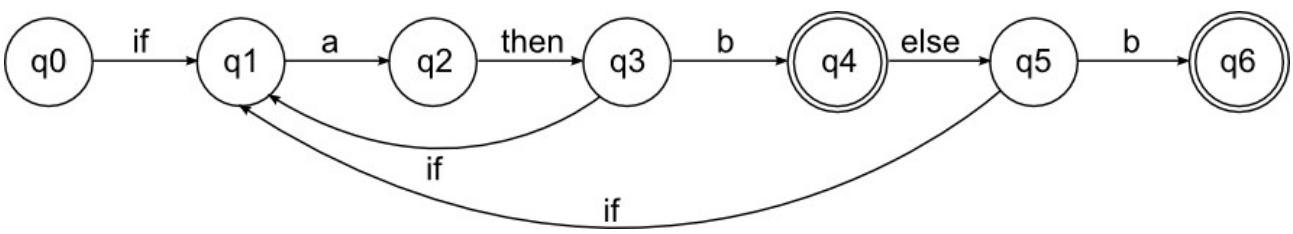
```
Pl.: MOV AX, [BX][DI]      ;AX-be tölti a BX+DI címen lévő adatot
```

Bázis plusz relatív címzés

```
Pl.: MOV AX, adat[BX][DI] ;AX-be tölti az adat+BX+DI címen lévő adatot
```

Ez a megoldás használható többdimenziós tömbök kezelésére.

b. Ismertesse az IF szerkezete felismerő automatát



c. Determinisztikus véges automaták

Mint már említettük, a véges automaták a formális nyelvek felismerőinek egy viszonylag egyszerű matematikai modelljét adják. A véges automaták általában a következő fő részekből tevődnek össze: egy bemenő szalag, amelyre egy szó van felírva, s amelyet bizonyos megengedett bemenő jelekből alkottunk – ezt a szót a véges automata működése során balról jobbra haladva elolvassa –, valamint egy vezérlőmű, amelynek véges sok ún. belső állapota lehet, amit úgy mondunk, hogy véges a memóriája. A megengedett bemenő jelek halmaza is véges, és ezek a jelek alkotják az automata bemenő ábécéjét.

Az automaták működése diszkrét jellegű: egymás utáni pillanatokban – ütemekben – lépnek egyet-egyet. Az ún. determinisztikus véges automaták vezérlőművéhez minden ütemben pontosan egy belső állapotot tudunk hozzárendelni, és a lépés során az automata elolvassa a bemenő szó soron következő betűjét. Bekapcsoláskor a véges automata egy rögzített belső állapotba, az ún. kezdőállapotba kerül, majd a bemenő szónak balról vett legelső elemét olvassa el. Az éppen elfoglalt belső állapottól, valamint a beolvasott jeltől függően a vezérlőmű minden ütemben átkerül valamilyen belső állapotba, és rátér a jobbról következő jel elolvasására.

A determinisztikus véges automata akkor fejezi be a munkáját, ha a bemeneti szalagra írott szó összes betűjét leolvasta, de akkor is leáll a működéssel, ha valamilyen ütem során a leolvasott bemeneti jel és az aktuális belső állapot nem határozzák meg egyértelműen az automata soron következő belső állapotát.

Ha egy determinisztikus véges automata, kiindulva a rögzített kezdőállapotából, miután leolvasta a bemenő szalagjára felírt szót, egy olyan belső állapotba kerül, amely az állapotok egy előre rögzített részhalmazának, az ún. végállapotok halmazának eleme, akkor azt mondjuk, hogy az automata felismerte, elfogadta ezt a bemenő szót. Az automata által felismert nyelvet azok a szavak alkotják, amelyeket az automata – az előbbi értelemben – elfogad.

A determinisztikus véges automatákat tehát leginkább olyan felismerőkként ábrázolhatjuk, amelyek nem rendelkeznek külső memóriával (4.4. ábra).