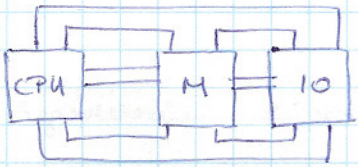
Utasítástípusok:

1) Adatmozgató:

- a „move" szóval kapcsolható össze, de ez valójában másolás marad ⇒ COPY
- ahová másoljuk, az ott álló adat VISSZAVONHATATLANUL eltűnt (megsemmisült)
- alaptípusa:

a) MOVE ⇒ igazából COPY (törlés, másolás)



• amikor a CPU és az IO között mozog az adat, ezek az IN-OUT utasítások.

b) PUSH-POP

LIFO szerkezet (stack) ⇒ pl gyökvonás sorozatának megoldása. Ha a programban gyököt akarnak vonni, de azt máshol tárolják, akkor a Program Counterben nem a köv. utasítás címe lesz, hanem a gyökvonásé. Ideiglenesen a köv. utasítás címét a stack-be kell elhelyezni el. Lévén - betét ⇒ ez a push-pop.

II) Aritmetikai:

a) összeadás - kivonás (ADD, SUB)

b) szorzás - osztás (MUL, DIV)

ezek az utasítások előjeles számoknál nem jók, ha mégis használják ⇒ a konverzió a kettes komplemenssel történik.

III) Logikai:

a) AND, OR, XOR

b) NOT

c) NEG → kettes komplemens előállítása (és aritmetikai is, és logikai is)

b) Biten fordítása → NOT A → akkumulátor minden bitje megfordul

a) AND:

```
  0 1 0 1 0 0 1 1
ÉS (1 1 1 1)(0 0 0 0)
  0 1 0 1 0 0 0 0
```

az egyik értéket visszaírja műveletvégző egységnek

- ahol a műveletvégző egységben 0 van, ott 0 lesz, ahol 1 volt, ott az szerepel, ami a másikban volt.

VAGY:

```
     0 1 0 1 0 0 1 1
VAGY (1 1 1 1)(0 0 0 0)
     1 1 1 1 0 0 1 1
```

- ahol a műveletvégzőben 1 van, ott 1 lesz, ahol 0 volt, ott a kiindulási érték lesz.

ÉS: biteket lehet nullázni

VAGY: biteket lehet egyessé tenni

XOR:

```
     0 1 0 1 0 0 1 1
XOR  1 1 1 1 0 0 0 0
     1 0 1 0 0 0 1 1
```

ahol a műveletvégző 0, ott a kiindulási nem változik, ahol 1, ott a kiindulási érték negáltja lesz.

NOT = XOR FF-fel.

IV) Programszámláló (PC) értékét módosító: (↗ mai neve: Instruction Pointer = IP)

- CALL-RETURN : valahová lépünk (nem oda, mint ami a PC-ban van), majd visszatérünk.

a) feltétlen
b) feltétel nélküli
} -UGRÁS (elágazás)

a) más néven: elágazás

ha a feltétel teljesül, máshol folytatja, ha nem teljesül, mintha a feltétel ott sem lett volna, halad a program tovább.

Egyik fajtája: SKIP (egy fix ugrás) ⇒ egy utasítást kihagy. ⇒ Ennek feltétel nélkül nincs értelme

RETI v. IRET (IT)

pl.: nyomtató rázott a munkával, meg kell jegyezni, h. más programban hol tartottunk, kiszolgáljuk a nyomtatót, majd visszatérünk.

interact reset ⇒ megszakítást befejező visszatérés

V) Léptető - forgató:

jobbra v. balra léptető utasítások (l v. O.)

Kérdések: mi lesz az utolsó és a legelső bittel?

- Ø lép be a léptetős utasításoknál
- a CARRY-bitbe lép be a „kilépő" bit, azaz felfelé, azaz lefelé lép.

- A PC processzorának még be kellett hozni az előjeles léptetést.

előjel: 0 = +
1 = −

előjeles jobbra léptetés

ábrázolás: −128-tól 127-ig. Akkor jó a szám, ha kisebb, mint 64. ill. −64.

jobbra léptetésnél a legfelső bit belép még egyszer, így megtartja az előjelét.

FORGATÓ:

A kilépő bit vissza is lép.

a) a 8 bitet kezeli egységnek CARRY-t kihagyva } forgat
b) a 9 bitet — " — CARRY-n keresztül

VI.) <u>CPU állapotát módosító:</u>

a) megszakítást tiltó/engedélyező utasítások

pl.: bérelszámoló → lekérdezik az időt. → nem jöhet közbe megszakítás.

10 óra → megszakítás → 00 perc
↓ helyette
10 óra → 59 perc → megszakítás

VII) <u>FLAG állapotát módosító:</u>

pl.: CARRY-bit legyen 1-es.

- hibák jelzésére használják.

pl.: $\sqrt{-58}$ ... stb.

ezért lehet a Flag állapotát direkt módosítani

VIII) a) HALT

- CPU-nak nincs feladata, folyton kérdezi, h. van-e utasítás → ekkor alkalmazhatjuk a Halt utasítás → célszerű → kikapcsol mindent, amit lehet.

b) IDLE

csak „éberen alszik"

HALT → ha van munka → jön egy megszakítás, akkor dolgozni kezd, kiteszi a stack-be a halt-ot. A végrehajtott utasítás után RET, és visszatér a HALT mögé.

Halt esetén sem lehet az alaplap tápját kikapcsolni → a RAM nem felejtheti el a tartalmát.

Címzési módok:

a) Abszolút címzés: (DIREKT)

- IN-OUT utasításoknál elképzelhető
megmondjuk a memóriarekesz sorszámát
- probléma: ha 100bytnyi memóriát akarunk lenullázni → nem vesszük 100 utasítást

b) Indirekt címzés:

- nem azt adjuk utasításba, hogy mi a cím, hanem, hogy melyiknél van az értéke.
- egy logikai ciklussal lehet a 100byte-nyi memóriát lenullázni, mert növelhető annak az

értéke, ahol a cím van eltárolva.

Kétféle eljárást lehet alkalmazni

- Valamihez képest címzünk (⇒ relatív címzés megfelelője/változója)

A rendszer fejlődésével más helyre kerültek a felhasználói programok ⇒ nem lehetett az abszolút címet használni. Relatív címet használtak általában a Programszámláló jelenlegi helyéhez képest.

- Index regiszter ⇒ nem egyetlen értékhez, hanem egy kiszámított értékhez képest lehet eltéríteni

Index = Indirekt hivatkozás + konstans

A index regiszter csak a hozzáadott konstans miatt volt index regiszter.

c) STACK CÍMZÉS:

- mielőtt meghívjuk, tegyük bele a stack-be, a tetejére kerül, hogy hová kell visszatérni. Kivesszük, gyököt vonunk, majd jön a RET, és a stack-ben van az eredmény.

d) Literális v. álcímzés:

van olyan, amikor a cím úgy néz ki, mint az érték, vagy fordítva. Ilyenkor az adatot kell az akkumulátorba tenni.