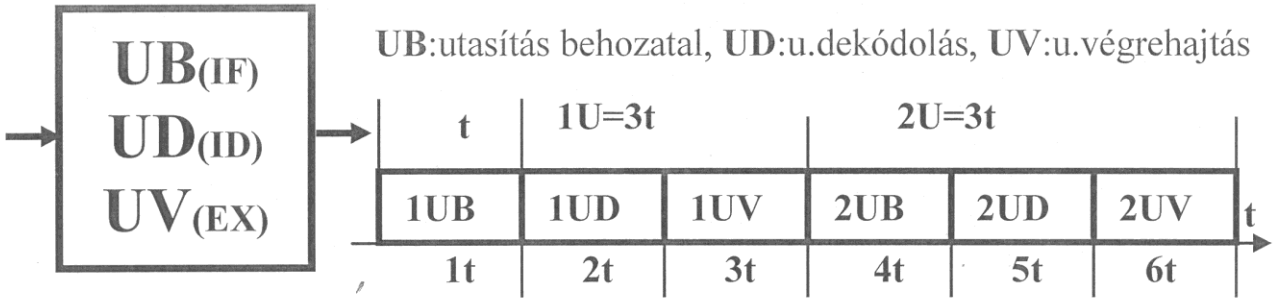
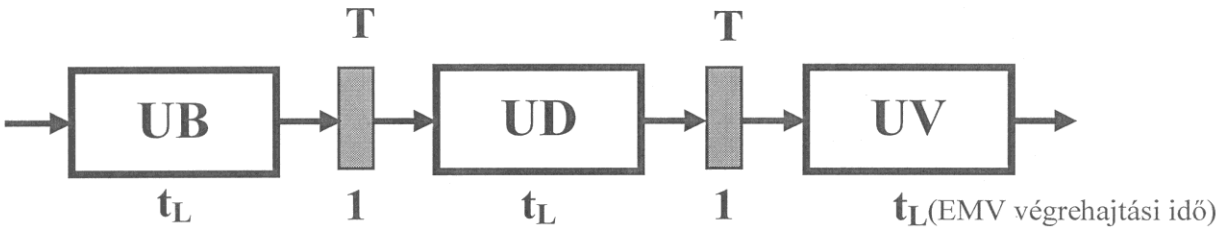
Hagyományos utasítás végrehajtás:



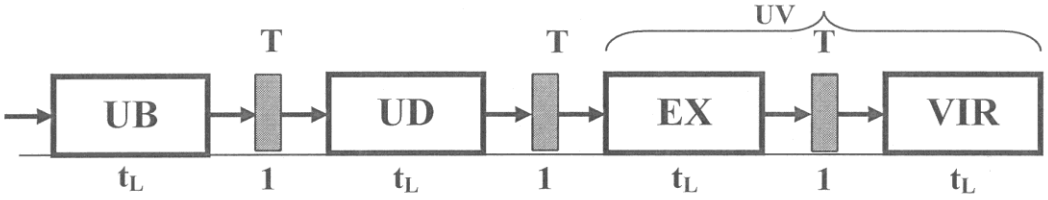
$T_u = 3 \cdot t$ $1ut = 3 \cdot t,$ $nut = n \cdot T_u = 3 \cdot n \cdot t$

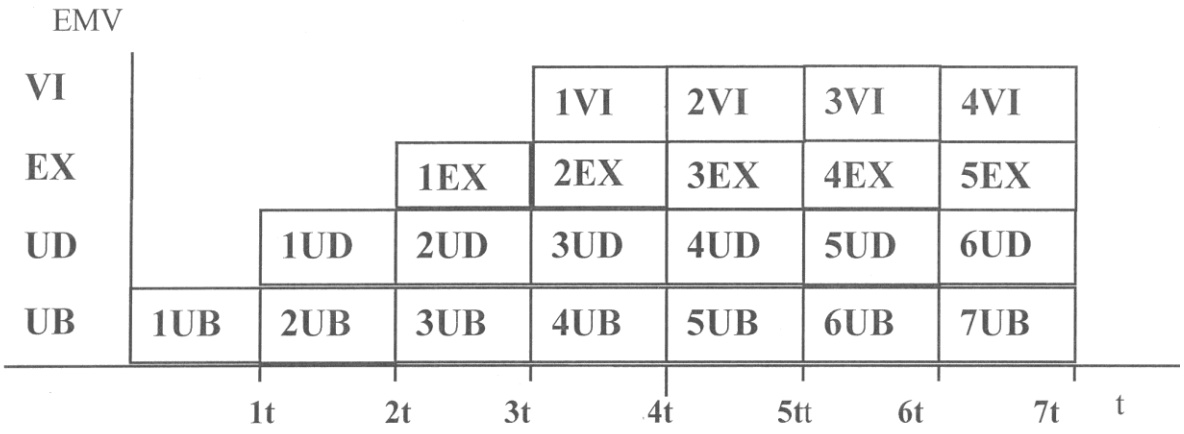
PIPE-LINE

pl.:a fenti részfeladatokat megoldó, párhuzamosan működő elemi műveletvégző, tárolóval



Finomítás: pl.: négy elemi műveletvégző (pl.:RISC Load, store)





Hatékonyság jellemzése??

Utasítás lappangási idő (Instruction latency) $T_L = EMV_{sz} \cdot t_L + 3 = 4t + 3$

Itt megegyezik az utasítás végrehajtás idejével $T_L = T_u$

4 utasítás/idő $T_N = 4/7t \approx 1ut/2t$??

Időegység alatt befejezett utasítások száma (throughput): **TP**

TP=1ut/2t ??! optimális esetben (4t után): **TP=1ut./t**

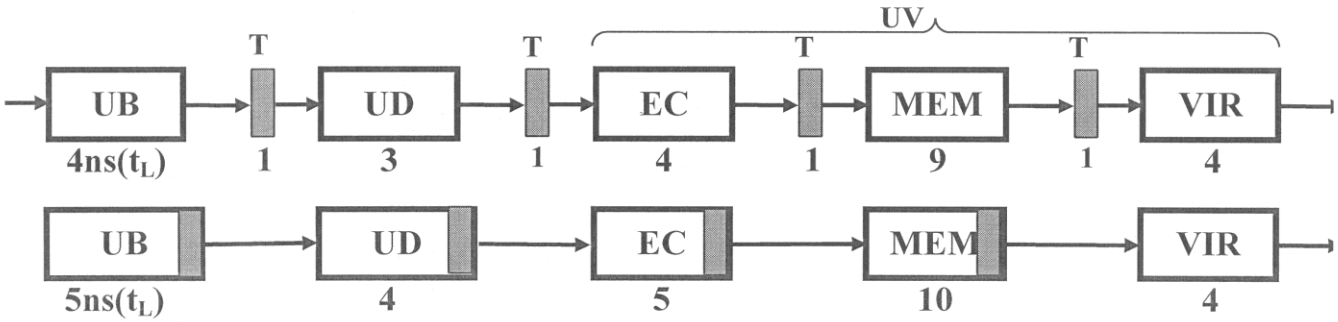
Sebesség növekedés (speed up) $Su_{PL} = \frac{T_{exec}}{T_{exePL}} = \frac{4 \cdot 4t}{7t} = \frac{16}{7} \cong 2.3$

pl.:10^5 utasításra $Su_{PL} = \frac{10^5 T_{exec}}{T_{exePL}} = \frac{10^5 \cdot 4 \cdot t}{4 \cdot t + 10^5 \cdot t} \cong 4$ **$Su_{PL} \approx EMV_{sz}=4$**

Problémák:

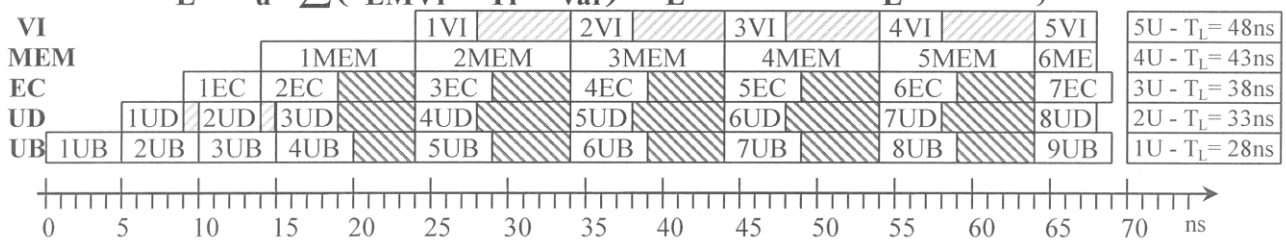
- **Műveletvégzőknek eltérő időkre van szükségük**

pl.:utasítás végrehajtás további finomítással: **EC:** aritmetika, effektív címszámítás, **MEM**: memória írás/olvasás , **VIR:** eredmény visszaírása



Aszinkron (ASAP(as soon as posible)) ütemezés

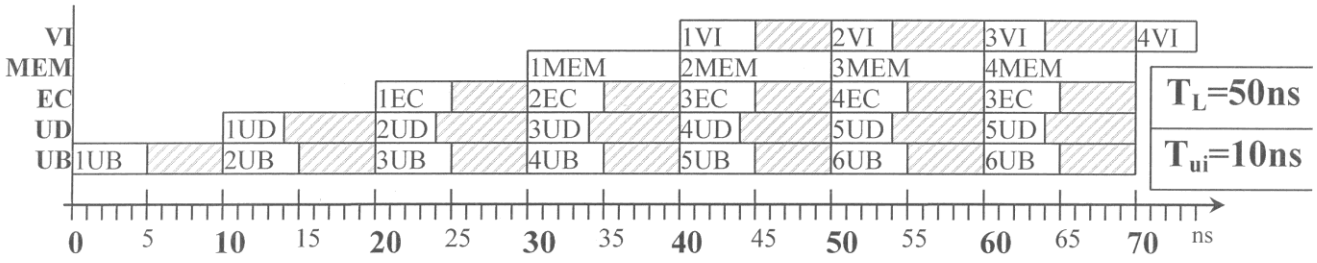
Utasítás $T_L=T_u=\sum(t_{EMVi}+t_{Ti}+t_{var})$ T_L=? 1.ut T_L =28ns, 2.ut=33ns



Probléma: ugyanazon utasítás eltérő végrehajtási ideje! Megoldás?!

Puffertároló ?!, vagy szinkron időzítés

Szinkron időzítés (a leghosszabb műveletvégző idejével) **$T_{ui}=t_{CLK}=t_{Lmax}$=10ns**



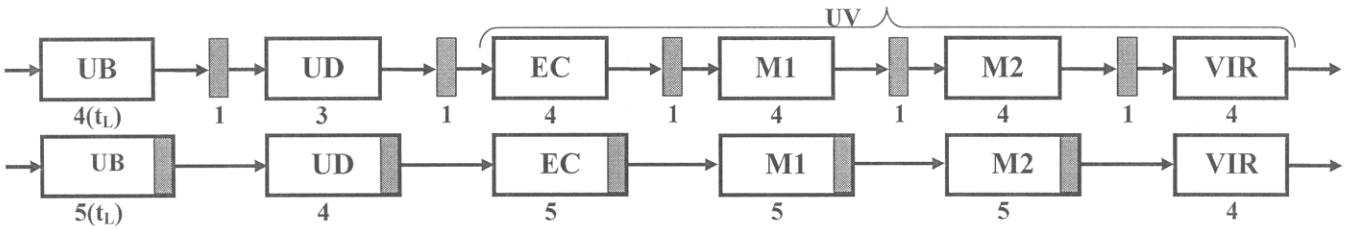
- **Probléma T_L= T_u állandó, de sok az üres idő!??**

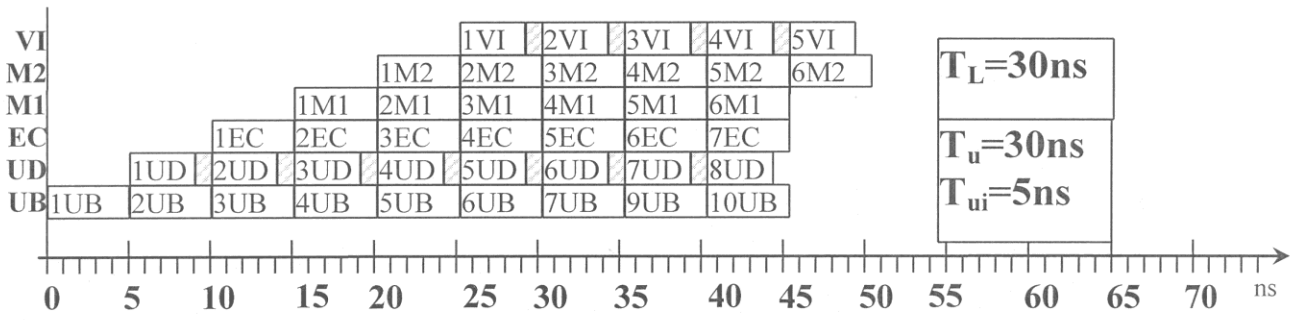
MEM ideje nagyobb, kiegyensúlyozatlanság (unbalance)!!

TP=1ut./ t_{CLK} = 1ut/10ns

10^5 utasítás esetén $Su_{PL} = \frac{10^5 T_{exec}}{T_{exePL}} = \frac{10^5 \cdot 5 \cdot t}{5 \cdot t + 10^5 \cdot t} \cong 5$ **$Su_{PL} \approx EMV_{sz}=5$**

Megoldás: a memória kezelés két egység: M1, M2



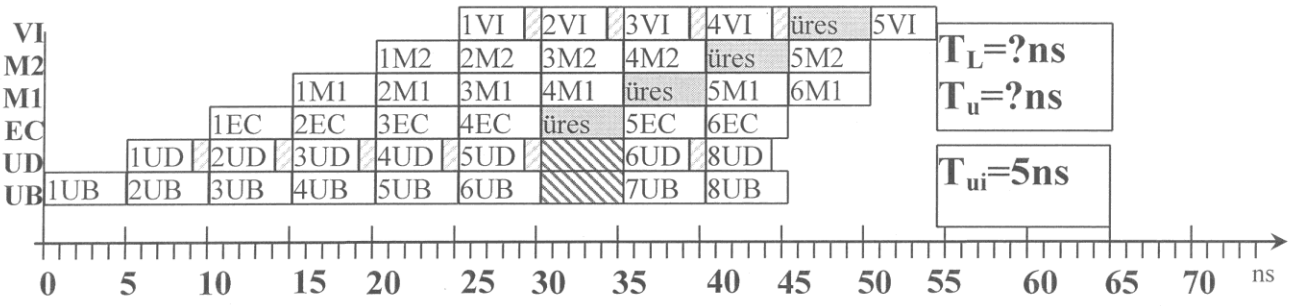


TP=1ut./5ns

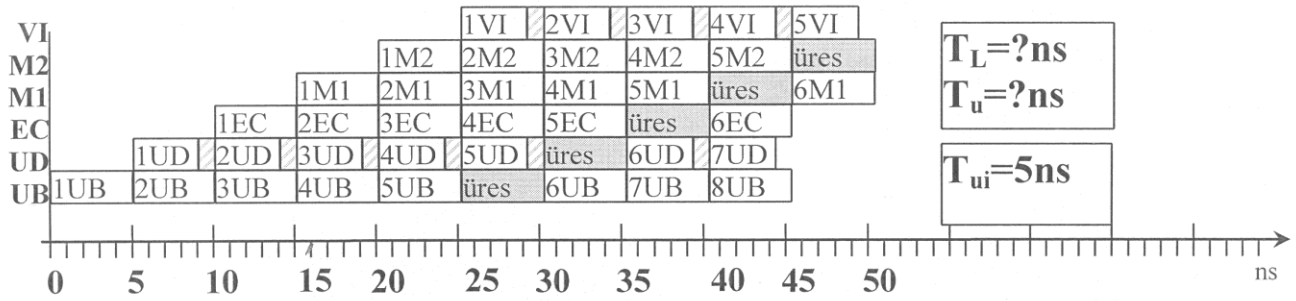
10^5 utasítás esetén $Su_{PL} = \frac{10^5 T_{exec}}{T_{exePL}} = \frac{10^5 \cdot 6t}{6t + 10^5 \cdot t} \cong 6$ **$Su_{PL} \approx EMV_{sz}=6$**

- **Nem vettük figyelembe az esetleges egyidejű használatot**

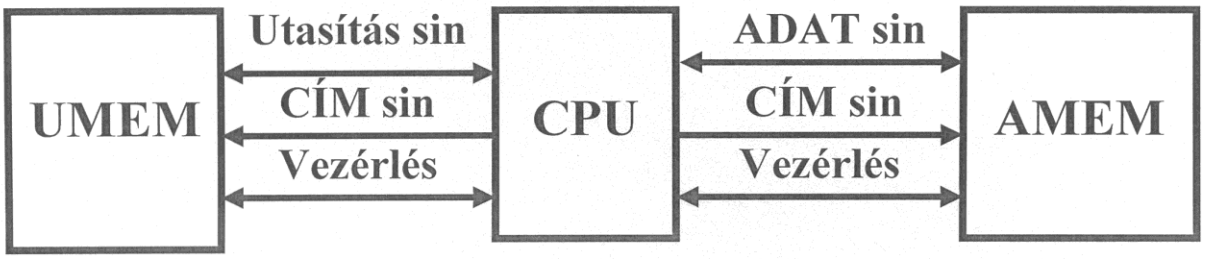
pl.: EC-VI, vagy UD-VI, néha UB-MEM, stb.
pl.: 2.utasítás VI-EC (címzési módtól függ)
megoldás: puffer tárolók, vagy üres (Nop) állapot közbeiktatása



pl.:3.utasítás az op. memóriába ír és 6.utasítást onnan kell beolvasni

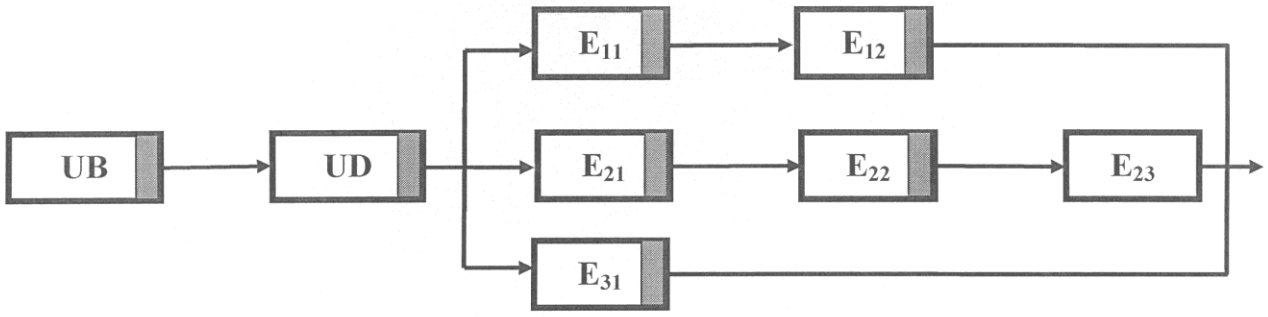


UB-MEM ütközés oldása: Queue, (puffer)
-Harvard architektúra,



-Többszintű, szeparált cache,
További problémák:

- **végrehajtás lépései utasítás csoportonként különböznek**



más utasítás, más feldolgozó ág, egyes egységek átléphetők
eleve több párhuzamos feldolgozó szalag (pl.:Pentium4)

- **utasítás egymásra hatás**
  - **Feldolgozási (structural): egyidőben ugyanarra a műveletvégzőre van szükség (megoldás lehet a fenti ábra)**

- **ADAT: utasításnak szüksége van a még nem meglévő előző utasítás eredményére** (fordítóprogram figyelheti, műveletek más sorrendben történő végrehajtása, vagy NOP beiktatása)
- **Procedurális (control):**
  feltételes vezérlésátadás, nem tudjuk hol folytatódik
  - **uP386-leállítja a betöltést és a feldolgozást a feltétel kiértékelődéséig**
  - **uP486-előre rögzített módon (pl.:nem ág) megjósolja az utat, megjelölve tölt, végrehajtással vár a kiértékelődésig…….**
  - **Pentium-dinamikusan számítja az előzmények alapján a valószínű ágat, és tovább mint a uP486-nál**

PC tartalmak, stack tartalmak megőrzése (pl.:Branch, IT)
IT, érvényre jutás!