Beágyazott információs rendszerek

4. Időmérés, időszolgáltatás, óra-szinkronizáció

2020. október 8.

4. Időmérés, időszolgáltatás, óra-szinkronizáció

Időmérés eszközei és módszerei:

(1) ***Időmérés elektronikus számlálóval:***

A forrás által generált ún. "kapuidő" maga a mérendő időtartam.

A mérés kezdetekor nullázott számláló a kapuidő alatt beérkezett impulzusokat számlálja.

$T_x \cong \frac{N}{f_0}$, ahol *N* a számláló tartalma, f_o pedig az órajel frekvencia.

Az időmérés (worst-case) relatív hibája: $\left|\frac{\Delta T_x}{T_x}\right| \cong \left|\frac{1}{N}\right| + \left|\frac{\Delta f_0}{f_0}\right|$

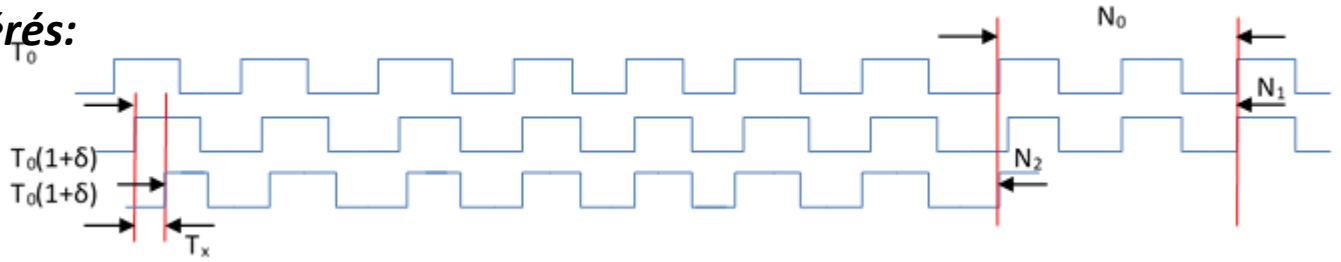
Ezt az összefüggést a teljes differenciál felírásából kiindulva származtatjuk:

$dT_x = \frac{\partial T_x}{\partial N} dN + \frac{\partial T_x}{\partial f_0} df_0 = \frac{1}{f_0} dN - \frac{N}{f_0^2} df_0$, amit elosztva $T_x = \frac{N}{f_0}$-szel $\frac{dT_x}{T_x} = \frac{dN}{N} - \frac{df_0}{f_0}$

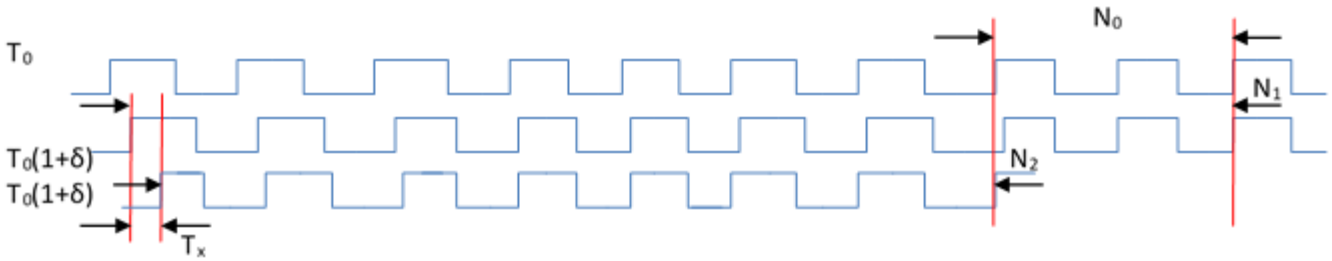
Mivel a megváltozások előjelét nem ismerjük, ezért legtöbbször a relatív megváltozások abszolút értékét írjuk fel a legkedvezőtlenebb esetre.

(2) ***Kettős nóniuszos időmérés:***

A mérendő időtartam kezdete és vége egy-egy $T_0(1+\delta)$ periódusidejű, kvarcpontosságú órát indít.



Ezek jelét egy szabadon futó T_0 periódusidejű, kvarcpontosságú óra jelével hasonlítjuk össze, figyelve a felfutó élek egybeesését.



A mérendő időtartam **kezdetétől** az első koincidenciáig eltelt idő $N_1 T_0(1+\delta)$,

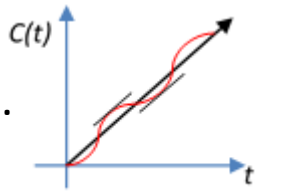
a mérendő időtartam **végétől** az első koincidenciáig eltelt idő $N_2 T_0(1+\delta)$,

a két koincidencia között eltelt idő pedig $N_0 T_0$. Ezzel $T_x = T_0[\pm N_0 + (N_1 - N_2)(1+\delta)]$

ahol az N_0 előtti előjelet a két koincidencia időbeni sorrendje határozza meg.

Ha $\boldsymbol{T_0 = 5\ nsec}$ és $\boldsymbol{\delta = 0.004}$, akkor a legkisebb, még mérhető időtartam $\boldsymbol{20\ psec}$!

Az órák, mint a valós idő adott pontosságú forrásai:



Az idő forrását **órának** nevezzük. A k-jelű óra a valós idő egy $C_k(t)$ függvénye.

Referencia óra: a teljesen pontos óra.

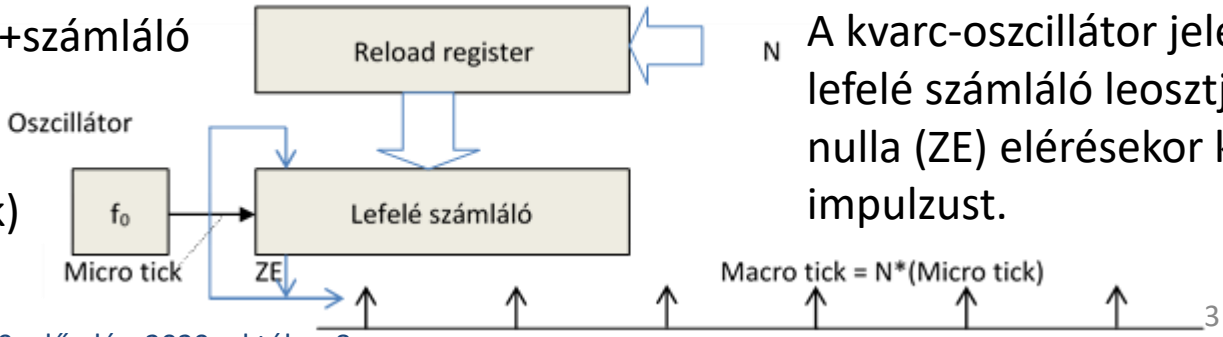
Ha a k-jelű teljesen pontos, akkor $C_k(t) = t; \forall t$

Helyes óra: a k-jelű óra helyes (correct) t_0-ban, ha $C_k\ t_0) = t_0$

Pontos óra: a k-jelű óra pontos (accurate) t_0-ban, ha $\partial C_k(t)/\partial t = 1; t = t_0$

Ha egy óra pontatlan egy adott időpillanatban, akkor azt mondjuk, hogy csúszik.

A fizikai óra: Oszcillátor+számláló felbontóképessége g (g: granularity), mikro-óraütés (microtick)



A kvarc-oszcillátor jelét egy lefelé számláló leosztja, és a nulla (ZE) elérésekor kiad egy impulzust.

Beágyazott információs rendszerek 1. zárthelyi 2019-ben

1. Mit jelent az, hogy egy periodikusan frissített valós idejű kép fázisérzéketlen (max. 1 pont)?

Egy periodikusan frissített RT képet *parametrikusnak*, vagy *fázis-érzéketlennek* hívunk, ha

$$d_{pontosság} > (d_{frissítés} + WCET_{\text{üzenet továbbítás}}).$$

A parametrikus RT kép a vevő oldalon bármikor felhasználható anélkül, hogy a beérkezés és a felhasználás fázisviszonyait mérlegelni kellene: még a pontossági időn belül megjön a frissítés.

2. Mi a kapcsolat a frissítési idő és a mintavételi idő között (indoklást is kérek!) (max. 1 pont)?

Közvetlenül semmi. A frissítési idő az időbeni pontossághoz kapcsolódik, a mintavételi idő pedig a folytonos jel rekonstruálhatóságához.

Ettől függetlenül gyorsabban változó jel időbeni pontossága várhatóan kisebb, ill. mintavételezése nagyobb mintavételi frekvenciát igényelhet, azaz áttételesen van kapcsolat.

3. Mit értünk állapot-megfigyelésen és mit esemény-megfigyelésen (max. 2 pont)?

Állapot megfigyelések: minden megfigyelés önállóan értelmezhető értéket ad. Jellegzetesen periodikus mintavételezéssel végezzük.

Esemény megfigyelések: az esemény adott időpontban bekövetkező állapotváltozás.

Mivel maga a megfigyelés is egy esemény, ezért nem lehetséges egy esemény közvetlen megfigyelése az irányított objektumban, csak annak következményeit tudjuk megfigyelni.

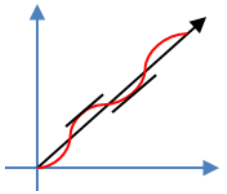
4. Mikor mondjuk, hogy az órák belülről szinkronizáltak, és mikor, hogy kívülről (max. 1 pont)?

“Belső szinkronizáció”: az órákat egymáshoz igyekszünk szinkronizálni.

“Külső szinkronizáció”: az órákat a referencia órához igyekszünk szinkronizálni.

5. Mikor mondjuk azt, hogy egy óra pontos (max. 1 pont)?

Pontos óra: a *k*-jelű óra pontos (accurate) t_0-ban, ha $\partial C_k(t)/\partial t = 1;\ t = t_0$



6. Mekkora az időtartam-mérés worst-case relatív hibája, ha egy $f_0 = 100MHz$-es, $2 * 10^{-5}$ relatív hibájú kvarcoszcillátorral és elektronikus számlálóval $150\mu s$ időtartamot mérünk (max. 3 pont)?

$$T_x = \frac{N}{f_0}, N = T_x f_0 = 150 * 10^{-6} * 10^8, \left|\frac{\Delta T_x}{T_x}\right| = \frac{1}{N} + \left|\frac{\Delta f_0}{f_0}\right| = \frac{10^{-4}}{1.5} + 2 * 10^{-5} \cong 8.7 * 10^{-5}$$

7. Egy kommunikációs rendszerben az üzenettovábbítás jittere $8\ ms$, minimális ideje $0.5\ ms$. Határozza meg az akció késleltetés idejét abban az esetben, ha

(1) a globális idő rendelkezésre áll (max. 1 pont);
(2) a globális idő nem áll rendelkezésre (max. 1 pont)!

Mit tehetünk, ill. mit kell tennünk, ha a továbbított érték időbeni pontossága $12\ ms$ (max. 2 pont)? $d_{max} - d_{min} = 8\ ms, d_{max} = 8.5\ ms$

(1) Van globális óra: akció késleltetés: $d_{max} = 8.5\ ms$
(2) Nincs globális óra: akció késleltetés: $2d_{max} - d_{min} = 16.5\ ms$

Ha a továbbított érték időbeni pontossága $12\ ms < 16.5ms$, akkor állapotbecslést kell alkalmaznunk. A frissítés gyakoriságának növelése nem segít, mert a frissített adatra is érvényesítenünk kell az akció késleltetést!

8. Mi a különbség az aperiodikus és a sporadikus taszkok között (max. 1 pont)? Miért lényeges ez a különbség (max. 1 pont)?

Sporadikus taszk: a kérések nem periodikusak, de ismert és fix egy olyan T_i időérték, ami minimálisan eltelik két kérés között.

Aperiodikus taszk: a kérések nem periodikusak, és nincs egy olyan ismert és fix T_i időérték, ami minimálisan eltelik két kérés között, tehát egy kérést követően azonnal megjelenhet egy következő kérés.

Ez lényeges különbség, mert ebben az esetben a DMA módszer nem alkalmazható!

9. A Deadline Monotonic Analysis (DMA) módszerének alkalmazásával határozza meg a 4-es jelű taszk worst-case válasz-idejét abban az esetben, ha a taszkok valamelyikének egyszeres hibája legfeljebb $T_F = 50\,ms$ gyakorisággal lép fel, és számítási időigénye $2\,ms$ (max. 4 pont):

Taszk	$T[ms]$	$C[ms]$	$D[ms]$
1	100	5	10
2	10	2	10
3	100	25	50
4	100	30	100

$$R_4' = C_4 + I_4 + \left\lceil \frac{R_4}{T_F} \right\rceil C_F$$

Lépés	R_4	$I_4 + \left\lceil \frac{R_4}{T_F} \right\rceil C_F$	R_4'
1	0	0	30
2	30	5+6+25+2	68
3	68	5+14+25+4	78
4	78	5+16+25+4	80
5	80	5+16+25+4	80

A worst-case válaszidő: $80ms < 100ms$.

A Dual Priority Scheduling módszer alkalmazását feltételezve határozza meg a 3-as jelű taszk promóciós idejét (max. 2 pont)!

Taszk	$T[ms]$	$C[ms]$	$D[ms]$
1	100	5	10
2	10	2	10
3	100	25	50
4	100	30	100

Lépés	R_3	$I_3 + \left\lceil \frac{R_3}{T_F} \right\rceil C_F$	R_3'
1	0	0	25
2	25	5+6+2	38
3	38	5+8+2	40
4	40	5+8+2	40

A worst-case válaszidő: $40ms < 50ms$.

A promóciós idő a kérés beérkezése után $10ms$, hiszen csak ekkor képes határidőre mindenképpen lefutni a 3-as taszk.

10. Mi a prioritás-öröklés algoritmus lényege, mikor használjuk, és milyen veszéllyel jár több kritikus szakasz esetén (max. 2 pont)?

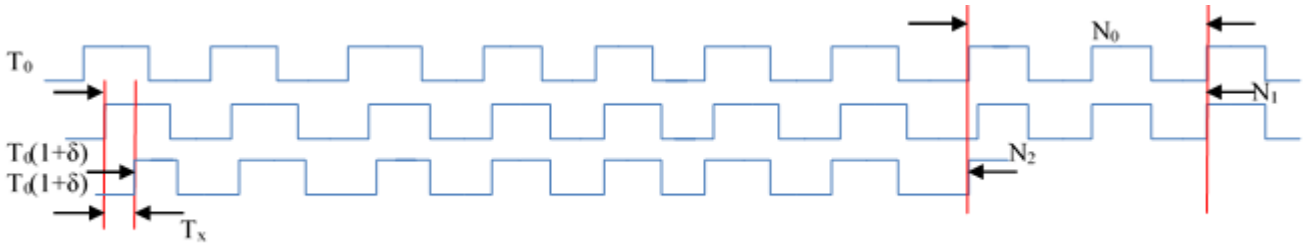
A prioritás öröklés algoritmus (Priority Inheritance Protocol, PIP)**:**

A prioritás inverzió elkerülése úgy lehetséges, hogy a *H* taszk kritikus szakaszba lépési szándékának megjelenésekor az *L* task ideiglenesen „megörökli” a *H* taszk prioritását (dinamikus prioritás), hogy mielőbb fejezze be a kritikus szakaszbeli teendőit, majd ezt követően visszatér az eredeti (statikus) prioritási rend.

Több kritikus szakasz alkalmazása esetén **prioritás inverzió** léphet fel.

11. Mutassa be és magyarázza el a kettős nóniuszos időmérés elvét és számításának összefüggését (max. 2 pont)! Mitől függ a mérés relatív pontossága (max. 1 pont)?

Kettős nóniuszos időmérés: A mérendő időtartam kezdete és vége egy-egy $T_0(1+\delta)$ periódusidejű, kvarcpontosságú órát indít. Ezek jelét egy szabadon futó T_0 periódusidejű, kvarcpontosságú óra jelével hasonlítjuk össze, figyelve a felfutó élek egybeesését.



A mérendő időtartam kezdetétől az első koincidenciáig eltelt idő $N_1 T_0(1+\delta)$, a mérendő időtartam végétől az első koincidenciáig eltelt idő $N_2 T_0(1+\delta)$, a két koincidencia között eltelt idő pedig $N_0 T_0$. Mindezek alapján

$$T_x = T_0[\pm N_0 + (N_1 - N_2)(1+\delta)]$$

ahol az N_0 előtti előjelet a két koincidencia időbeni sorrendje határozza meg.

A mérés relatív pontossága a T_x-et megadó kifejezés értékétől, és a megvalósítás pontatlansága miatt bizonytalan paramétereitől (T_0 és δ) függ. A számláló tartalmak bizonytalanságáról nem beszélhetünk. Részletesebb analízis esetén az indított órajelek tranziens hibáit, valamint a koincidencia detektálás hibáit is figyelembe kell vennünk.

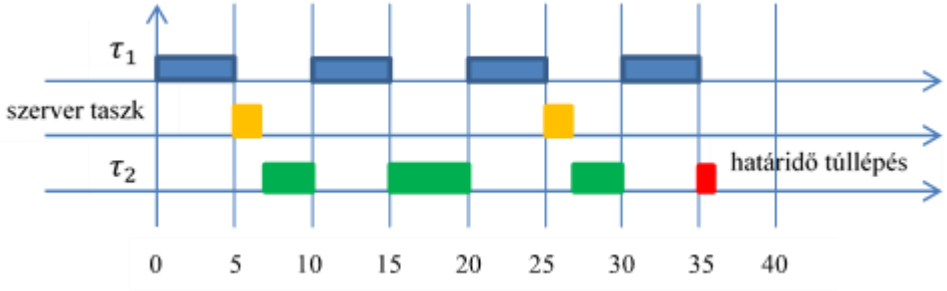
12. Két kemény valós idejű taszk ütemezését kell megoldanunk. Induláskor mindkettő futásra kész, és kéri a processzort.

	$C[ms]$	$T[ms]$
τ_1	5	10
τ_2	12	30

Emellett aperiodikus taszkokat szolgálunk ki egyidejűleg induló szerver taszk segítségével. A szerver taszk periódusa $20\ ms$, számítási kapacitása $2\ ms$. Mutassa be a három taszk ütemezését grafikus formában az RM algoritmussal (max. 2 pont)!

Az ütemezhetőség szükséges feltétele teljesül: $\mu = \frac{5}{10} + \frac{12}{30} + \frac{2}{20} = 1.$

Ha a szerver taszk kitöltené a kapacitásának megfelelő időt, akkor RM szerint a taszkok nem ütemezhetőek:

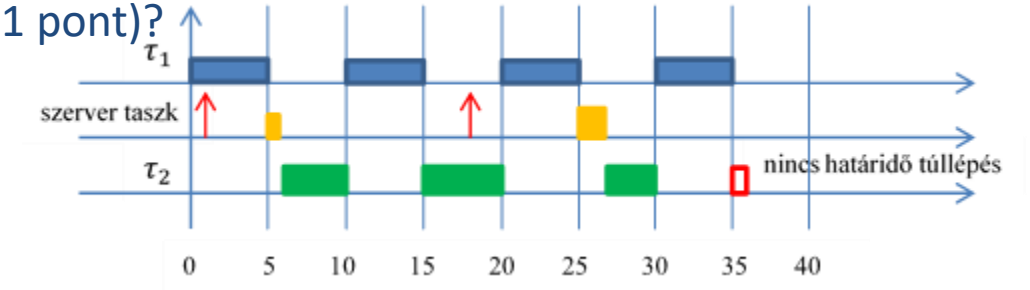


Ebből fakadóan az ütemezhetőség esetleges, az aperiodikus taszkok igényétől függ. Biztonságosabb (és helyesebb (!)) $2\,ms$ kapacitás helyett $1\,ms$ kapacitás licitálása, de ez egy másik feladat.

Először Polling Server, majd Deferrable Server alkalmazása esetére mutassa be, mikor kap kiszolgálást a $t_1 = 1ms$-ban beérkező, $1\,ms$ processzor időt kérő, és a $t_2 = 18ms$-ban beérkező, $2\,ms$ processzor időt kérő aperiodikus taszk (max. 4 pont)? Mekkora az aperiodikus taszkok válaszideje (max. 1 pont)?

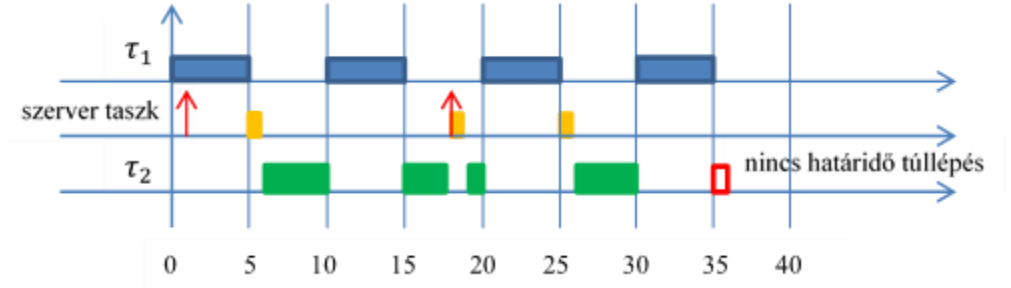
A Polling Server esete:

A válaszidők: $5\,ms$ és $9\,ms$

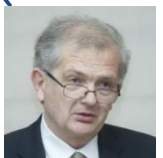


A Deferrable Server esete:

A válaszidők: $5\,ms$ és $8\,ms$



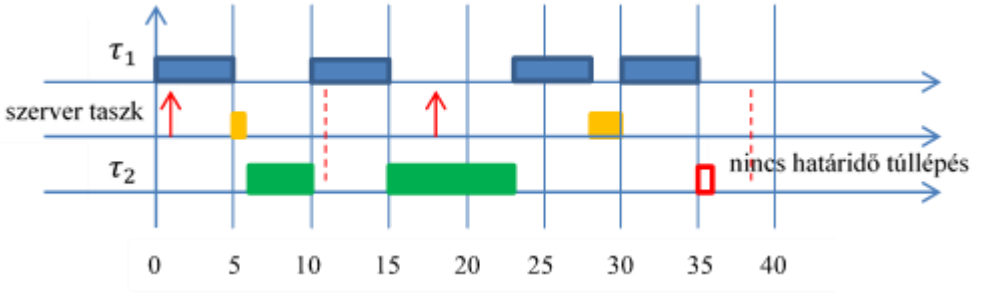
Az EDF algoritmus és a Total Bandwidth Server alkalmazása esetén hogyan alakulnak ezek a határidők (max. 3 pont)?



Mivel $\mu_P = \frac{5}{10} + \frac{12}{30} = 0.9$, és $\mu_S = \frac{2}{20} = 0.1$, ezért $d_1 = 1 + \frac{1}{0.1} = 11ms$,

ill. $d_2 = 18 + \frac{2}{0.1} = 38ms$.

A válaszidők: $5\ ms$ és $12\ ms$



13. Milyen feltétel teljesülése esetén ütemezhető egy kemény valós idejű, periodikus taszk-készlet az EDF algoritmussal, ha $D_i < T_i$ (max. 2 pont)?

Egy periodikus task-készlet akkor és csak akkor ütemezhető az EDF algoritmussal, ha minden $L > 0$ esetén

$$L \geq \sum_{k=1}^{n} \left(\left\lfloor \frac{L - D_k}{T_k} \right\rfloor + 1 \right) C_k$$

Ez az ún. processzor igény módszer.

14. Hasonlítsa össze az esemény-vezérelt és az idő-vezérelt rendszerek működését! Biztonságkritikus rendszerek esetén melyik alkalmazása célszerűbb (max. 2 pont)?

Az **eseményvezérelt** rendszerek a kiváltó események/kérések hatására hajtják végre az eseményhez rendelt programot. Ezzel a megközelítéssel kedvező válaszidők érhetők el, de a közel egyidejű események számának növekedésével a rendszer kapacitása, átbocsátóképessége, teljesítménye elégtelenné válik, és ebből adódóan a határidők betartása ellehetetlenül.

Az **idővezérelt** rendszerek esetében minden megoldandó feladathoz tervezési időben egy különálló időszeletet rendelünk, ezáltal a feladat-végrehajtás előzetesen ismert válaszidő mellett garantálható.

Ebből adódóan biztonságkritikus rendszerek esetén célszerűbb ez utóbbi.