Beágyazott információs rendszerek

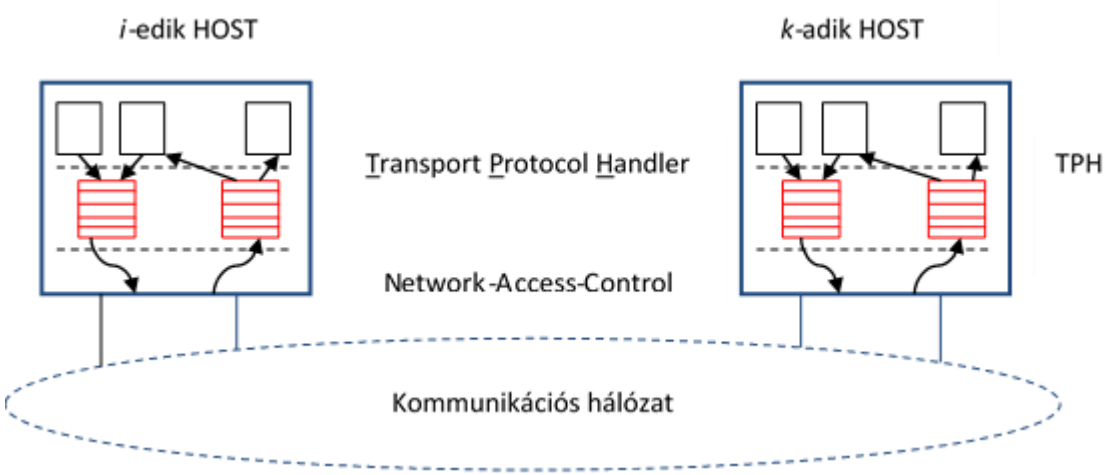
Valós idejű kommunikáció

2020. november 4.

6. Valós idejű kommunikáció

Általában bonyolult mechanizmusok, várólisták jellemzőek. Valós idejű követelmények nehezen teljesíthetőek.

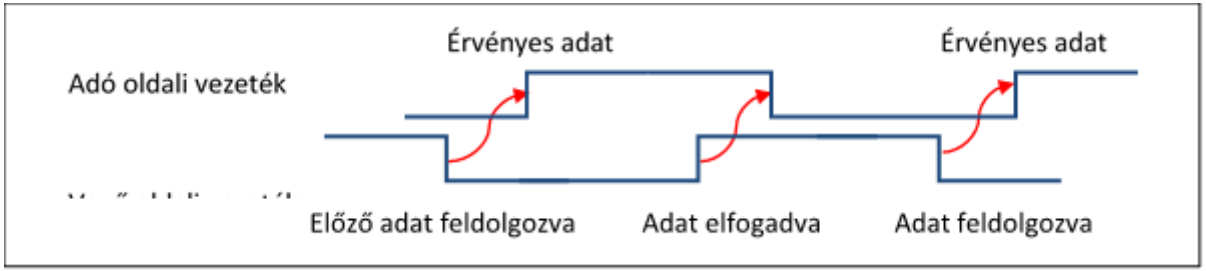
Az általános séma:



Az időviszonyok kritikus volta a fizikai szinten is jól azonosítható.

Aszinkron kommunikáció esetén szinkronizálás kell, ez a “handshaking”.

A kétvezetékes handshake:



A kommunikáció sebesség- és időviszonyait az adó és a vevő sebessége és egyéb feladatai együtt határozzák meg, hiszen az adat vevő oldali “feldolgozásáig” újabb adat továbbításában az adó nem gondolkodhat.

Követelmények:

1. Lehetőleg kis protokoll késleltetés (protocol latency) és jitter (latency jitter). (Latency árnyaltabb jelentése: lappangás, homály, elrejtettség).
2. Komponálhatóság: segíteni kell az időbeni követelmények teljesülését: HOST ↔ CNI (Communication Network Interface), időszakos tűzfal szerep, HOST önálló tesztelhetősége.
3. Flexibilitás: gépkocsi funkciók időbeni működése extrákkal, extrák nélkül …

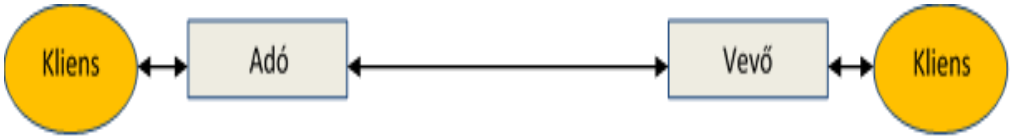
4. Hibadetektálás: Jósolható és hibatűrő kommunikáció kell. End-to-end nyugtázás. Egy szelep zárjon automatikusan, ha az állítását lehetővé tevő vezeték elszakad, de erről menjen értesítés a központnak.
5. Struktúra: pont-pont kapcsolat kezelhetetlen bonyolultságú kábelezéssel jár, helyette busz és gyűrű.

Az adatáramlás szabályozása (flow control):

Explicit forgalomszabályozás:

Példa: PAR (Positive Acknowledgement or Retransmission) protokollok:

Több változat van, de ezek közösek az alábbiakban:
(1) Az adóoldali kliens kezdeményez.
(2) A vevő jogosult késleltetni.
(3) A hibát az adó detektálja.
(4) Hibajavítás időbeni redundanciával.



Adó oldali program:
(1) Az ismételt küldések számlálóját nullázzuk.
(2) Indítjuk a visszaigazoláshoz rendelt time-out számlálót.
(3) Indítjuk az üzenetet.
(4) A time-out-on belül fogadjuk a visszaigazolást.
(5) Értesítjük a klienst a sikeres adattovábbításról.
Ha nincs visszaigazolás a time-out-on belül:
(a) Ellenőrizzük az ismételt küldések számlálóját, hogy elérte-e a maximumát.
(b) Ha igen, akkor megszakít minden tevékenységet, és hibát jelez a kliensnek.
(c) Ha nem, akkor inkrementálja az ismételt küldések számlálóját, és visszatér a fenti (2) ponthoz.

Vevő oldali program:

(1) Üzenet érkezésekor ellenőrzi, hogy ez az üzenet érkezett-e már korábban.

(2) Ha nem, akkor visszaigazol, és értesíti a kliensét.

(3) Ha igen, akkor csak visszaigazol. (Ilyenkor az előző visszaigazolás time-out időn túl érkezhetett az adóhoz, ha egyáltalán megérkezett.)

Megjegyzés:

Az adó oldalon a vétel visszaigazolása és a vevőoldalon az adat elfogadás időpontja között jelentős eltérés lehet.

Példa: Token vezérelt buszon az üzenettovábbítás ideje $\mathbf{1}\ ms$, a token körüljárás ideje $\mathbf{10}\ ms$.

A beállítandó time-out: $10 + 1 + 10 + 1 = \mathbf{22}\ ms$, hiszen worst-case esetben, ha éppen elment a token $10\ ms$-ot kell várni, erre jön az üzenettovábbítás $1\ ms$-a, majd a visszaigazoláskor ugyanez ismétlődhet.

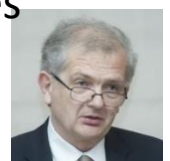
A $d_{min} = 1\ ms$, a $d_{max} = (ismétlések\ száma) * timeout + 10\ ms + 1\ ms$.

Ha kétszer ismétlünk (azaz háromszor próbálkozunk), akkor d_{max}= **55** *ms*.

Ezekkel néhány jellemző a következőképpen alakul:

- **jitter**= $d_{max} - d_{min} = 54\ ms$.
- **akció késleltetés**, ha van globális óra: $d_{max} = 55\ ms$.
- **akció késleltetés**, ha nincs globális óra: $2 * d_{max} - d_{min} = 109\ ms$.
- **a hibadetektálás késleltetése:** 3*time-out: 66 *ms*.

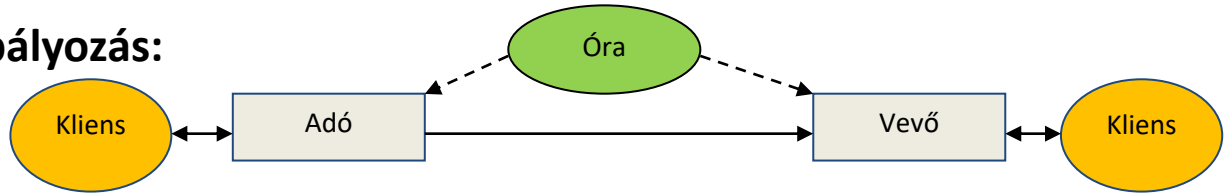
A PAR protokoll és a számpélda azt illusztrálja, hogy az ún. **explicit forgalomszabályozás** valós idejű alkalmazásokban **kedvezőtlen lehet** a nagymértékű jitter, akció késleltetés és hibadetektálás késleltetés miatt.



A PAR protokollnak sokféle változata van, de mindegyik a következőkön alapszik:

(1) A kommunikációt az az adóoldali kliens kezdeményezi.
(2) A vevő jogosult késleltetni az adót a kétirányú kommunikációs közegen keresztül.
(3) A kommunikáció hibáját az adó detektálja, nem a vevő.
A vevő nem kap arra vonatkozóan információt, hogy mikor történt a hiba.
(4) A hiba javítására időredundanciát használnak, amely növeli a protokoll késleltetést.

Implicit forgalomszabályozás:



A kommunikáció **idővezérelt.** Az adó és a vevő is rendelkezik egy tervezési időben elkészült időrendi táblázattal ("vasúti menetrend"). Ebben egyértelmű az adás és egyidejűleg a vétel időpontja/időintervalluma. Az adó az óraütés vezérlésére „kitolja" az üzenetet, a vevő pedig "behúzza" (push-pull jellegű működés). Ez a logika sok esetben **jobban illeszkedik a valós idejű követelményekhez!** A hibadetektálás például a vevő által azonnal lehetővé válik, ha a várt adat nem érkezik meg. (Az adó részéről ez egy ún**. fail-silent** üzemmódban létet jelent, azaz hibás állapotát azzal jelzi, hogy **nem küld üzenetet**.) **Globális időalap kell!**

Az adó az „óraütés" kezdeményezésére csak meghatározott időpontokban ad, **nincs handshake**.
A hibadetektálás a vevő dolga: tudja, hogy mikor kell/kellett volna üzenetnek érkeznie.
A hibatűrés **aktív redundanciával** valósul meg: *k* fizikai üzenet kópia, **ha legalább egy megérkezik**, addig sikeres.
A csatorna **egyirányú,** ami többszereplős esetben előnyös.

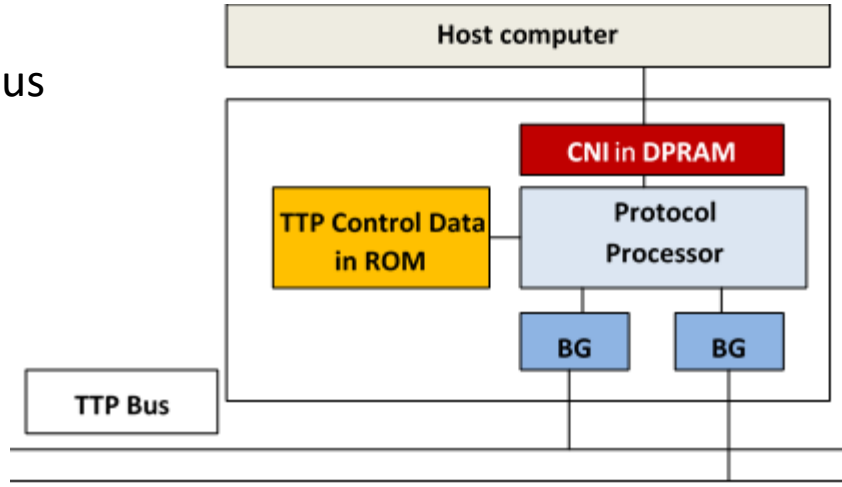
Az idővezérelt architektúra (Time Triggered Architecture, TTA) és az idővezérelt protokollok (Time-Triggered Protocols, TTP) Hard real-time (HRT) rendszerek implementálására szolgál.

Két változata van: **a TTP/C**, amely hibatűrő HRT rendszerekhez készült, és a **TTP/A**, amely olcsó ipari alkalmazások esetén jön számításba (pl. terepi busz (field bus) alkalmazásokban).

- A rendszer hibatűrő egységekből (FTU: **Fault Tolerant Unit**) felépülő **fürt** (cluster).
- Minden FTU cluster egy, kettő, vagy több csomópontból áll, amelyeket a kommunikációs hálózat köt össze.
- Minden csomópont két részrendszerből, a **host számítógépből** és a **kommunikációs vezérlőből** (**CC**) áll.
- A kommunikációs hálózat interfész (**CNI**) a csomóponton belüli interfész a host és a kommunikációs vezérlő (**CC**) között. A **CNI** egy dual-portos RAM memória (DPRAM). Az adat integritást a **Non-Blocking Write** (**NBW**) Protocol biztosítja (lásd később).

A kommunikációs vezérlő lokális memóriája tartalmazza az üzeneteket leíró listát (**Message Description List: MEDL**), amely meghatározza, hogy mely időpontban küldhet a csomópont üzenetet, ill. mely időpontban várhat más csomópontból. A MEDL méretét a fürt-ciklus mérete határozza meg.

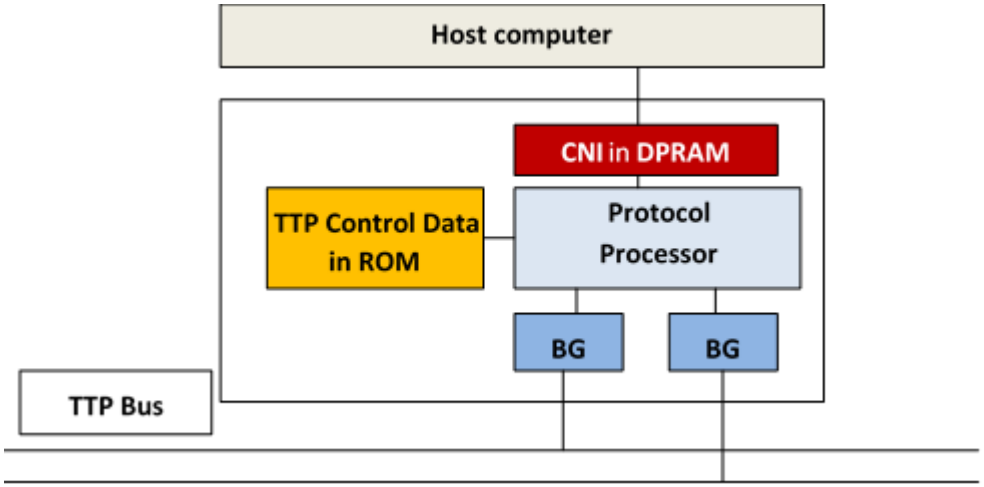
- A TTP vezérlő független hardverként ún. **Bus Guardian egységeket** is tartalmaz, amelyek figyelik a vezérlő busz-hozzáférési mintáit, és leállítják a vezérlő működését, ha a szabályos hozzáférési minták időzítése megsérül.



Fontos tulajdonságok:

(**1**) A **TTP** egy időosztásos-többszörös-hozzáférésű (**TDMA**) protokoll.

(**2**) A komponálhatóságot szolgálja, hogy a kommunikációs vezérlő **autonóm**, amelyet a **MEDL** és **a globális óra** vezérel.



A host számítógépek hibája nem tudja befolyásolni a kommunikációs rendszert, mert vezérlő jel nem megy át a **CNI**-n és a **MEDL** nem férhető hozzá a **host** felől.

(**3**) A kommunikáció módja **tervezési időben dől el** (olyan, mint a vasúti menetrend), mindenki előre tudja, hogy mikor kap, ill. mikor küld üzenetet. Ha hiányzik/elmarad az üzenet, akkor **azonnal detektálható** a hiba.

(**4**) Az üzenet azonosítása (**naming**): az **üzenet** és **küldőjének neve** nem kell, hogy része legyen az üzenetnek, a MEDL-ből kinyerhető. Ugyanannak az RT változónak **más és más nevet** adhatunk az egyes host-ok szoftverében.

(**5**) **Visszaigazolás:** előzetesen tudjuk, minden helyesen működő vevő veszi a helyesen működő adó üzenetét. Amint egy vevő visszaigazol egy üzenetet, arra lehet következtetni, hogy az üzenet kiküldése helyesen történt és azt minden helyesen működő vevő megkapta.

(**6**) **Hiba esetén** "hallgatás" az időtartományban: a TTP feltételezi, hogy a csomópontok támogatják a "**fail silence**" absztrakciót az időtartományban, ami azt jelenti, hogy egy csomópont vagy küld üzenetet a helyes időpontban, vagy nem küld semmit. A csomópontnak ezt a tulajdonságát a TTP vezérlőn belül a **Bus Guardian** (**BG**) valósítja meg.

Az amplitúdó tartományban a hibakezelés a host felelőssége, a TTP csak CRC-t biztosít.

A CNI felépítése:

A CNI az idővezérelt architektúra **legfontosabb interfésze**, mert ez az egyetlen interfész, amely a **host szoftvere által látható**.

A **Status Register**-eket a TTP vezérlő írja, a **Control Register**-eket pedig a host.

Status Registers	Control Registers
(S1) Global Internal Time	(C1) Watchdog
(S2) Node Time	(C2) Timeout Register
(S3) Message Description List	(C3) Mode Change Request
(S4) Membership	(C4) Reconfiguration Request
(S5) Status Information	(C5) External Rate Correction

S1: A fürt közös órája két bájton.
S2: a vezérlő saját órája.
S3: **MEDL** Pointer.
S4: annyi bitből áll, ahány csomópont van a fürtben. Ha egy bit "**TRUE**", akkor működött az illető csomópont a legutolsó kommunikációs időszeletben, ha "**FALSE**", akkor nem működött.

C1: A host periodikusan frissíti, a vezérlő ellenőrzi. Ha elmarad a frissítés, akkor a vezérlő – hibát sejtve – leállítja az üzenetküldést.
C2: A host írja, lejártakor megszakítást okoz. Például a host a fürt órájához szinkronizálhatja magát egy előírt későbbi időben.
C3: Például új ütemezésre lehet áttérni ennek segítségével.
C4: Meghibásodás esetén szerepcsere kezdeményezhető.
C5: külső óra szinkronizálást (pl. GPS) tesz lehetővé.

A Message Description List (MEDL) felépítése

Node Time	Address	D	L	I	A
Mikor	*Mit: Az üzenet címe*	*irány*	*hossz*		

I: azt adja meg, hogy inicializálással kapcsolatos üzenet, vagy normál üzenet.

A: egy további paraméter-mező, amely a változtatásokkal (mode changes) kapcsolatos információkat tartalmaz.

A hibatűrő egységek (Fault-Tolerant Units) rendeltetése egy csomópont hibájának a maszkolása.

Ha a csomópont a **fail-silent** absztrakciót valósítja meg, akkor a **csomópontok duplikálása** elegendő **egyszeres csomópont-hiba tolerálására**.

Ha a csomópont nem valósítja meg a **fail-silent** absztrakciót, de lehet **érték-hibája** a CNI-nél, akkor **háromszoros moduláris redundancia** (TMR: triple modular redundancy) valósítandó meg. Ez „**háromból kettő**” szavazással maszkolja az érték-hibát.

Ha a csomópont hiba esetén fellépő viselkedéséről **semmit sem tudunk**, azaz akár **bizánci típusú hiba** is felléphet, akkor **négy csomópont** tudja maszkolni a hibát.