O ANALIZĂ COMPARATIVĂ A PROTOCOLULUI CAN ÎN SISTEMELE DE CONTROL DISTRIBUITE

Doina Banciu
doina.banciu@ici.ro

Siegfried Cojocaru
scojocaru@ici.ro

Institutul Național de Cercetare – Dezvoltare în Informatică, ICI, București

Rezumat: Protocolul CAN-Controller Area Network a fost creat de compania germană Bosch în anul 1985 inițial pentru aplicațiile auto, cu scopul de a asigura o comunicație serială robustă și sigură, și de a reduce complexitatea și greutatea rețelei de cablare necesară interconectării unităților de control din automobile.Treptat însă și alte domenii ca automatizarea industrială, echipamentele medicale, aplicațiile militare, echipamentele de testare au inceput să realizeze beneficiile utilizării CAN. În lucrare se explică o parte din noțiunile fundamentale ale protocolului CAN punându-se accent pe comportamentul în timp al acestuia și se arată avantajele în cazul utilizării acestui protocol în sistemele de control distribuite. De asemenea, sunt analizate comparativ pentru protocoalele Ethernet, DeviceNet (CAN bus), și ControlNet anumite trăsături specifice cum ar fi timpii de blocare , metoda de soluționare a coliziunilor, eficiența utilizării rețelei .

Cuvinte cheie: CAN protocol, fieldbus, distributed system, rețele de control, automatizarea industrială.

1. Introducere

Protocoalele de comunicație pentru aplicațiile industriale le cuprind și pe cele din așa numitul domeniu "Fieldbus" în care se tratează comunicația între procesoare, senzori și actuatori. Sistemele distribuite au luat avânt în acest domeniu o dată cu marele progres în domeniul semiconductoarelor care a oferit și posibilitatea realizării de cipuri cu un raport preț/capacitate foarte avantajos. S-au dezvoltat sisteme de senzori și actuatori din ce în ce mai inteligente care au necesitat Standarde corespunzătoare pentru comunicația lor.

Arhitectura de comunicație tradițională pentru sistemele de control este "point-to-point" și a fost implementată cu succes în industrie de câteva decade. Totuși nevoia de extindere din punct de vedere fizic și al funcționalității a atins limitele arhitecturii de tip point-to-point. Astfel un sistem de control tradițional centralizat de tip point-to-point nu mai este adecvat pentru noile cerințe cum ar fi modularitatea, controlul descentralizat, diagnostice integrate, întreținerea ușoara și rapidă și costuri scăzute.

Astăzi există cele mai diverse protocoale care se diferențiază între ele prin proprietățile lor, prin tehnica de sincronizare între diferite noduri de rețea, prin lungimea datelor, prin media numărului de pachete care se transmit în unitatea de timp, prin numărul de noduri de rețea, prin lungimea fizica a rețelei, prin felul în care sunt corectate erorile, prin toleranța la greșeli, prin disponibilitate, siguranță, robustețe etc.

Pentru sistemele de control rețelele trebuie în general să satisfacă două criterii principale: întârzierea de timp în limitele dorite și garantarea transmisiei:aceasta înseamnă că un mesaj trebuie transmis cu succes în cadrul unei anumite întârzieri de timp). Rețelele de control sunt în mod tipic bazate pe unul din doua protocoale de acces la mediu,CAN(folosit în Smart Distributed System), DeviceNet și CAN kingdom) și Token Ring sau Bus (folosit de Process Field Bus(PROFIBUS), Manufacturing Automation Protocol (MAP), ControlNet, and Fiber Distributed Data Interface (FDDI)).

2. Controller Area Network- CAN

CAN este un protocol de comunicație serial dezvoltat în general pentru aplicații din industria automobilelor dar capabil totodată să ofere performanțe bune și în alte aplicații industriale critice în timp. Protocolul CAN este optimizat pentru mesaje mici și folosește pentru accesul la mediu arbitrarea CSMA cu prioritizare a mesajelor (CSMA/MP).

2.1. Telegrama CAN

Fiecare telegramă constă dintr-un anumit număr de biți împărțiți pe anumite câmpuri. Aceste câmpuri ar putea fi, de exemplu, End of Frame, câmpul CRC, câmpul de date și câmpul de arbitrare. În câmpul de arbitrare este conținută pe de o parte prioritatea telegramei, iar pe de altă parte acest câmp reprezintă în același timp, și adresa logică a informației. Adresa logică constă din 11 biți de identificare. Conform

specificației pentru o telegrama normală - Standardul de Cadru CAN 2.0A - sunt admise 2032 de adrese diferite (0-2031). Aceasta înseamnă că într-o rețea pot fi transmise maximum 2032 informații diferite. În multe aplicații acest număr poate să nu fie suficient. De aceea, specificația a fost dezvoltata (extinsa) și s-a ajuns la 29 de biți în câmpul identificator, permițând astfel 2^{29} telegrame diferite. Aceste telegrame sunt cunoscute conform specificației CAN-2.0B sub denumirea de cadre CAN extinse [2].

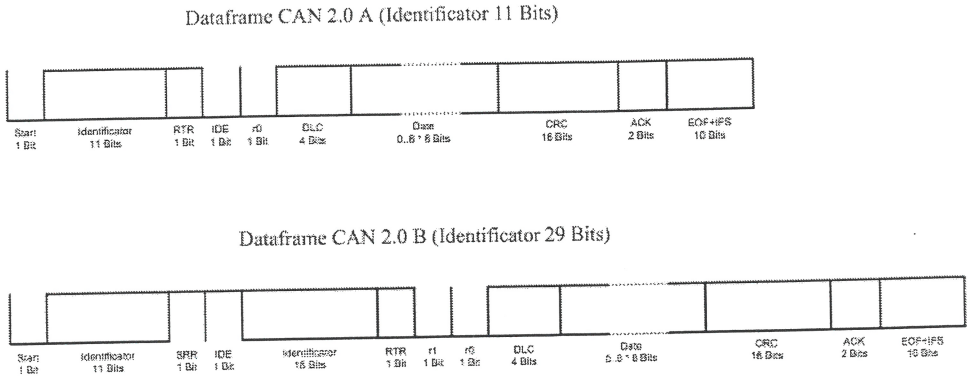

Figura 2.1. 1 Formatul de cadru pentru standardele CAN2.0 A și CAN 2.0 B

2.2. Arbitrarea la CAN

Controlul accesului la bus este realizat cu ajutorul arbitrării bit cu bit și fără să distrugă telegramele implicate în coliziune. Fără să le distrugă înseamnă că un câștigător al arbitrajului, adică telegrama cu prioritatea cea mai mare, nu trebuie retransmisă din nou de la început.

Pentru acest procedeu trebuie ca drivere-le fizice corespunzătoare să fie realizate intr-un anumit fel. Valorile logice de pe bus Null și Unu trebuie să fie dominante și rezesive.

Conform convenției trebuie ca în timpul arbitrarii un Unu transmis de un nod sa fie rescris de un Null transmis de un alt nod. Filozofia de atribuire a dreptului de acces la bus constituie la rețelele de tip multimaster un factor decisiv care caracterizează sistemele distribuite prin capacitate sau performantă, prin întârzierea de transmisie și astfel prin capacitatea de funcționare în timp real. În acest sens CAN se comportă foarte bine. Când mai multe noduri solicită accesul la bus simultan acesta va fi acordat celui mai important concurent. Dacă un nod vrea sa transmită și busul este ocupat atunci el va trebui să aștepte până când se termină transmisia în curs.

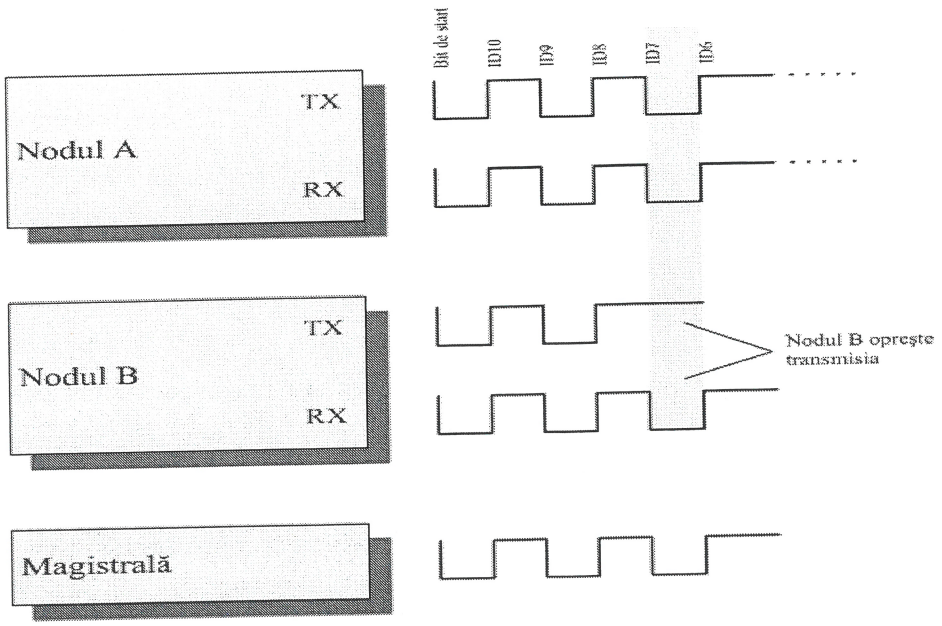

Figura 2.2.1 Arbitrarea la CAN [3]

Până când apar primii biți diferiți, niciunul dintre cele două noduri nu îl observa pe celalalt. În acest exemplu, biții diferă pentru prima dată la bitul numărul 7 din câmpul identificator. În acest moment nodul B realizează că a pierdut arbitrarea şi întrerupe imediat transmisia telegramei sale. În continuare comută nodul B pe poziția de receptor deoarece este posibil ca telegrama pe care o trimite nodul A sa trebuiască sa fie prelucrată chiar de el.

2.3. Analiza de timp pentru CAN

La universitatea din York a fost analizată într-un proiect de cercetare analiza matematica a evoluțiilor în sistemele care folosesc priorități cu alocare fixă. La CAN accesul la bus este organizat în funcție de prioritățile cu alocare fixă şi astfel este posibilă aplicarea modelului de analiză dezvoltat. La prima conferință internațională despre CAN a avut loc o expunere cu acest model de analiză.

Procedeul de analiză face câteva presupuneri simple cu privire la modelul de transmisie[1]:

1. un cadru m se transmite numai o data în intervalul de timp T_m;
2. cadrul pregătit de transmisie întârzie în controllerul CAN cel mult J_m înainte ca el sa fie transmis pe bus;
3. un cadru m are o lungime de până la s_m baiți;
4. există o funcție de eroare E(t), care dă eventuala întârziere maximă care poate apărea în intervalul t ca urmare a erorilor şi a reluărilor de transmisie;
5. dreiverul software pentru controllerul CAN garantează că , mesajul cu cea mai mare prioritate locală este pregătit pentru transmisie;
6. toți identificatorii sunt cunoscuți.

Pornind de la aceste presupuneri se poate calcula întârzierea maximă pentru fiecare cadru (definită ca R_m). R_m este măsurată din momentul de timp în care cadrul este gata în controllerul CAN şi până în momentul în care el a fost transmis.

De obicei, se dă apriori o limită de timp şi apoi se calculează întârzierile pentru a stabili dacă sunt respectate limitele de timp. Alte teorii despre organizarea evoluțiilor sistemelor propun ca identificatorii să fie astfel aleşi încât cadrele cu limitele de timp cele mai restrictive să primească prioritățile cele mai mari în timp ce celor cu tolerante de timp mai mari sa le revină prioritățile mai mici. Din acest motiv trebuie arătat cum este făcută alocarea identificatorilor la cadrele de mesaj.

Ecuația de bază pentru calculul întârzierii maxime R_m este prezentată în cele ce urmează. Mai întâi, se determină întârzierea unui cadru m. Acesta se compune din timpul de aşteptare până în momentul începerii transmisiei şi durata transmisiei.

$$R_m = T_m + C_m$$

Timpul de aşteptare începe cu momentul în care cadrul ajunge în coada de aşteptare ca urmare a apariției unei cereri de transmisie şi se termină imediat ce arbitrarea a fost câştigată. Timpul de transmisie este timpul de care este nevoie pentru transmisia propriu-zisă. Acest timp poate fi calculat din lungimea maxima a cadrului şi din numărul maxim de biți de stuff. Timpul de transmisie se calculează din următoarea ecuație care dă valoarea pentru C_m.[1]

$$C_m = \left(\left[\frac{34 + 8s_m}{4} \right] + 47 + 8s_m \right) \tau_{bit}$$

Termenul s_m reprezintă mărimea cadrului m m în baiți. Mărimea τ_{bit} reprezintă durata unui bit care în cazul unei rate de transmisie de 1 Mbit/s este 1 μs. Este de notat faptul că numitorul ecuației este 4 şi nu 5 cu toate ca regula de *stuffing* prevede valoarea 5.

Pornind de la şirul de date

.....0000111100001111000011111.....

Trebuie ca la început sa fie intercalat un bit de stuff de "0". După aplicarea algoritmului de sttuffing

rezultă următorul şir:

.....0000111110000011111000001111.....

Acest caz este foarte puțin probabil dar este de luat în calcul când se analizează cazul cel mai defavorabil.

Pentru a calcula întârzierea maximă în procesul de transmisie, trebuie aflat cât timp poate să blocheze busul un mesaj cu prioritate inferioară (cu identificator mai mare), înainte să reîncepă un nou ciclu de arbitrare. Aceasta valoare corespunde duratei de transmisie a celui mai lung mesaj cu prioritatea cea mai mică. Pentru calcul, mai este nevoie, de asemenea, de intervalul de timp în care intervin alte mesaje de prioritate mai mare în timp ce mesajul m aşteaptă să fie transmis.

Acest timp rezultă din ecuația:

$$\sum_{\forall j \in hp(m)} \left\lceil \frac{t_m + j_j + \tau_{bit}}{T_J} \right\rceil C_j$$

în care $hp_{(m)}$ reprezintă mulțimea de cadre cu prioritate mai mare decât m. Cu t_m se analizează timpul maxim în care cadrul m se află în coada de aşteptare înainte ca el însuşi să câştige procesul de arbitrare. j_i reprezintă "Queuing Jitter" , timpul scurs de la dorința de transmisie până la prima participare la procesul de arbitrare.

În cele din urmă, trebuie analizată şi întârzierea de timp care rezultă din tratarea erorilor (prin trimiterea de cadre de eroare şi reluarea procesului de transmisie). Pentru ca el să poată fi calculat, se introduce funcția de timp $E(t)$, adică funcția de eroare, care furnizează timpul consumat prin reluările transmisiei în intervalul t. În cazul unui bus perfect fără erori valoarea acestuia este 0.

Din cele prezentate mai sus rezulta timpul maxim de aşteptare[1]:

$$t_m = B_m + \sum_{\forall j \in hp(m)} \left\lceil \frac{t_m + j_j + \tau_{bit}}{T_J} \right\rceil C_j + E(t_m)$$

Această ecuație nu este uşor de soluționat şi conduce la un şir cu valoarea inițială 0 pentru t_m.

$$t_m^{n+1} = B_m + \sum_{\forall j \in hp(m)} \left\lceil \frac{t_m^n + j_j + \tau_{bit}}{T_J} \right\rceil C_j + E(t_m)$$

3. O analiză comparativă a rețelelelor Ethernet, ControlNet şi DeviceNet(CAN)

3.1. Ethernet (CSMA/CD)

Ethernet foloseşte mecanismul Carrier Sense Multiple Access cu Collision Detection (CSMA/CD) pentru a rezolva disputa pentru mediul de comunicație. Protocolul CSMA/CD este specificat în standardul de rețea IEEE 802.3 şi se poate descrie pe scurt astfel [5].Când un nod vrea să transmită el ascultă la rețea. Dacă rețeaua este ocupată, el aşteaptă până când este libera "idle"; altfel, el transmite imediat. Dacă două sau mai multe noduri depistează rețeaua idle şi decid să transmită simultan, mesajele acestor noduri transmițătoare intra în coliziune şi mesajele sunt "corupted". În timp ce transmite, un nod trebuie de asemenea să asculte pentru a detecta o eventuală coliziune a mesajului. La detectarea unei coliziuni între doua sau mai multe mesaje, un nod transmițător opreşte transmisia şi aşteaptă o perioada aleatoare de timp până încearcă să retransmită. Această perioadă aleatoare este determinată de algoritmul standard Binary Exponential Backooff (BEB). Timpul până când încearcă din nou este ales aleator dintre sloturile de timp 0 şi $2^i - 1$, în care i reprezintă al i-lea eveniment de coliziune detectat de nod, iar un slot de timp este minimul de timp necesar pentru o transmisie "round-trip". Oricum, atunci când se ajunge la 10 coliziuni intervalul este fixat la maximul de 1023 sloturi. După 16 coliziuni nodul încetează încercările de transmisie şi raportează "failure" înapoi la procesor. Rezolvarea acestei situații poate fi încercată în straturile superioare.

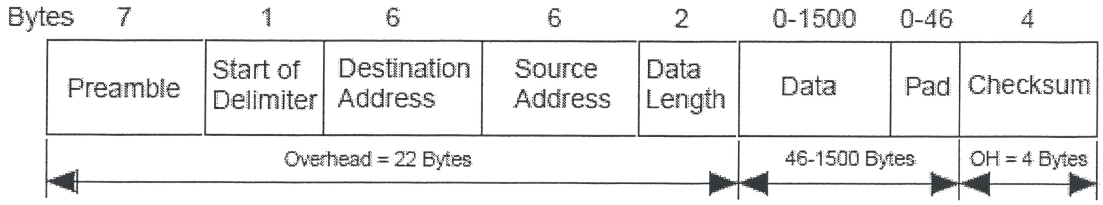

Figura 3.1. 1 Formatul de cadru Ethernet (CSMA/CD)

Formatul de cadru Ethernet este arătat în figura 3.1.1[5]. Mărimea pachetului de date este între 46 şi 1500 baiţi. Există o cerinţă ca mărimea datelor să nu fie 0 deoarece standardul impune ca frame-rile valide trebuie sa aibă o lungime de cel puţin 64 baiţi, de la adresa destinaţie la câmpul checksum (72 baiţi incluzând preambulul şi începutul câmpului Delimiter). Dacă porţiunea de date a unui frame este mai mică de 46 baiţi atunci se foloseşte câmpul Pad pentru a completa câmpul până la valoarea minimă. Există două motive pentru limitarea valorii minime a acestui câmp. Primul înseamnă că devine mai uşoară distincţia dintre cadrele valide şi cele care nu conţin informaţii (garbage). Când un tranciever detectează o coliziune el trunchiază fream-ul curent ceea ce înseamnă ca *stray bits* şi bucăţi din cadru apar frecvent pe cablu. Al doilea motiv este acela că împiedică un nod să termine transmisia unui cadru scurt înainte ca primul bit să fi atins capătul cel mai îndepărtat al cablului unde el poate intra în coliziune cu un alt frame. Pentru un Ethernet de 10-Mbs cu lungime maximă de 2500 m şi 4 repetoare timpul minim admis pentru durata unui cadru sau a unui slot este de 51.2μs care reprezintă timpul necesar pentru a transmite 64 baiţi la 10Mbs[5].

3.1.1 Timpul de blocare pentru Ethernet

Vom considera mai întâi timpul de blocare pentru Ethernet, care include timpul ocupat în cazul coliziunilor cu alte mesaje şi apoi timpul necesar pentru a fi retransmise. Algoritmul BEB descris mai devreme indică un timp de aşteptare probabilistic. O analiză exactă a întârzieri de timp de blocare este foarte dificila. La un nivel mai ridicat timpul de blocare estimat poate fi descris de următoarea ecuaţie[7]:

$$E\{T_{block}\} = \sum_{k=1}^{16} E\{T_k\} + \quad T_{resid}$$

în care T_{resid} denotă timpul rezidual perceput de un nod i până când reţeaua este idle şi $E\{T_k\}$ este timpul aşteptat pentru coliziunea nr. k. $E\{T_k\}$ depinde de numărul de noduri blocate şi deblocate şi, totodată, de rata de sosire a mesajelor la fiecare nod. Pentru cea de-a 16-a coliziune, nodul varsă (discard) acest mesaj şi raportează un mesaj de eroare la unităţile de procesare din nivelul superior. Poate fi văzut ca T_{block} nu este deterministic şi poate fi nelimitat datorită descărcării de mesaje.

Avantaje: Datorită overhead-ului scăzut pentru accesul la mediu, Ethernet foloseşte un simplu algoritm pentru operarea în reţea şi nu are aproape nici un *delay* la încărcări uşoare, joase ale reţelei. Nu se foloseşte deloc banda de comunicaţie pentru a câştiga accesul la reţea spre deosebire de protocolul token bus sau token ring. Folosit ca o reţea de control Ethernet foloseşte în mod obişnuit standardul de 10 Mbs; Ethernetul de viteză mare(100 Mbs sau 1Gbs) este folosit în mod uzual în reţelele de date [5].

Dezavantaje: Ethernet este un protocol nondeterministic şi nu suporta nici o prioritizare a mesajului. La încărcări mari ale reţelei partea de coliziune a mesajului reprezintă o mare problema deoarece afectează substanţial capacitatea totală a datelor şi întârzierile de timp, care devin nelimitate. Efectul de captură existent în algoritmul standard BEB, în care un nod transmite în exclusivitate pachete pentru un anumit timp, în ciuda altor noduri care aşteaptă accesul la mediu duce la *„incorectitudine"* şi are ca urmare degradarea substanţială a performanţei. Bazat pe algoritmul Beb un mesaj poate fi rejectat după o serie de coliziuni; De aceea comunicaţia end-to-end nu este garantată. Datorită condiţiei ca frame-ul valid să aibă o anumita valoare minimă, Ethernet foloseşte un mesaj mare pentru a transmite o cantitate mică de date [9], [5].

3.2. ControlNet (Token Passing Bus)

MAP, PROFIBUS şi ControlNet sunt exemple tipice de reţele de control cu bus de tip token-passing. Acestea sunt reţele deterministice deoarece timpul de aşteptare maxim până să se trimită un cadru de

mesaj poate fi caracterizat de timpul de rotație a cheii. Protocolul Token bus (IEE 802.4) permite o topologie de tipul liniar, multidrop, în forma de arbore şi segmentată [8] .Nodurile din rețeaua token bus sunt aranjate logic intr-un inel iar, în cazul ControlNet, fiecare nod ştie adresa predecesorului şi a succesorului său. În timpul funcționarii rețelei, nodul cu cheia transmite cadre de date până când ori le-a terminat pe cele pe care le avea de transmis ori a trecut timpul cat putea deține cheia. Apoi nodul regenerează cheia şi o transmite succesorului său logic din rețea. Dacă un nod nu are nici un mesaj de transmis, el doar trece cheia către succesorul sau. Locația fizică a succesorului nu este importantă deoarece cheia este transmisă la vecinul logic. Nu apar coliziuni între cadrele de date deoarece un singur nod poate transmite într-un moment. Protocolul garantează un timp maxim pentru accesul la rețea al fiecărui nod şi de asemenea are posibilitatea de a regenera cheia dacă deținătorul acesteia încetează transmisia şi nu trece cheia către succesorul sau. De asemenea nodurile pot fi adăugate dinamic la bus şi pot fi scoase din inelul logic.

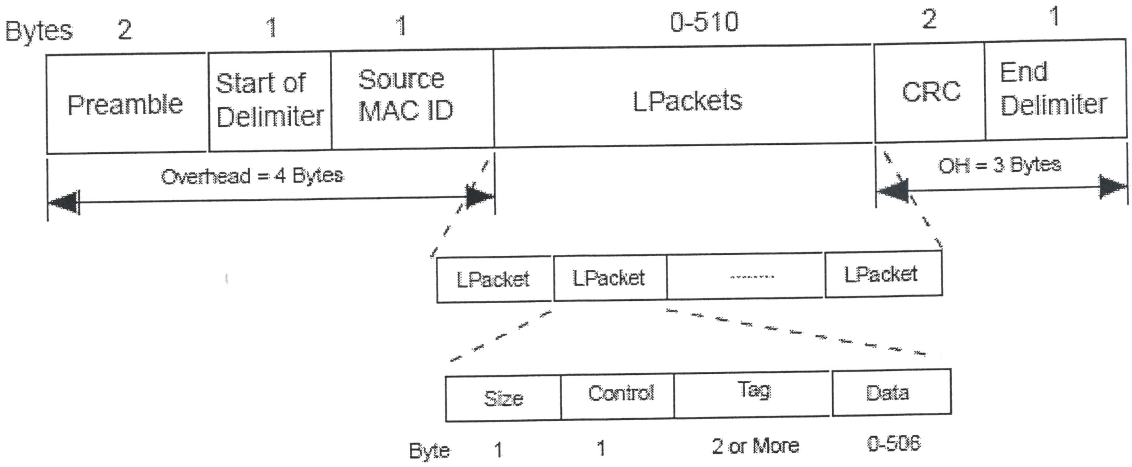

Figura 3.2.1. Cadrul de mesaj pentru ControlNet(TokenBus)

Formatul de cadru de mesaj pentru ControlNet este aratat în fig.3.2.1 [8] Overheadul total este de 7 baiți, incluzând preambulul, stard delimiter, ID-ul de MAC al sursei, CRC-ul şi delimitatorul de sfârşit. Cadrul de pachete de date, în special Lpacket sau Link Packet Frame, poate include câteva pachete Lpacket care conțin mărimea, controlul, tag-ul, data şi o adresa de destinație individuală cu o valoare totala a mărimii cadrului intre 0 şi 510 baiți. Câmpul de mărime specifică numărul perechilor de baiți (de la 3 la 255) conținuți într-un pachet individual Lpacket. Fiecare pereche de bait trebuie să includă câmpurile size, control, tag, şi data.

Protocolul ControlNet adoptă un mecanism implicit token-passing şi atribuie fiecărui nod un ID de MAC unic(de la 1 la 99). Ca la majoritatea bus-urilor cu pasare a cheii , nodul care poseda cheia poate transmite data; oricum nu este vorba de o pasare reala a cheii prin rețea. În schimb, fiecare nod monitorizează ID-ul MAC de sursă al fiecărui mesaj primit. La sfârşitul unui cadru de mesaj fiecare nod setează un „registru de cheie implicit" la ID-ul de MAC de sursă primit +1. Dacă registrul de cheie implicit este egal cu propriul ID de MAC al nodului, atunci acel nod poate să transmită mesaje. Toate nodurile au aceeaşi valoare în registrele lor implicite prevenind astfel coliziunile la mediu. Dacă un nod nu are date de transmis, el doar trimite un mesaj cu un câmp gol în pachetul Lpacket denumit cadru null.

Lungimea unui ciclu denumit Network Update Time (NUT) în ControlNet sau Token Rotation Time (TRT), în general, se împarte în trei mari parți: scheduled, unscheduled şi guard band după cum este arătat în figura 3.2.2 [4]

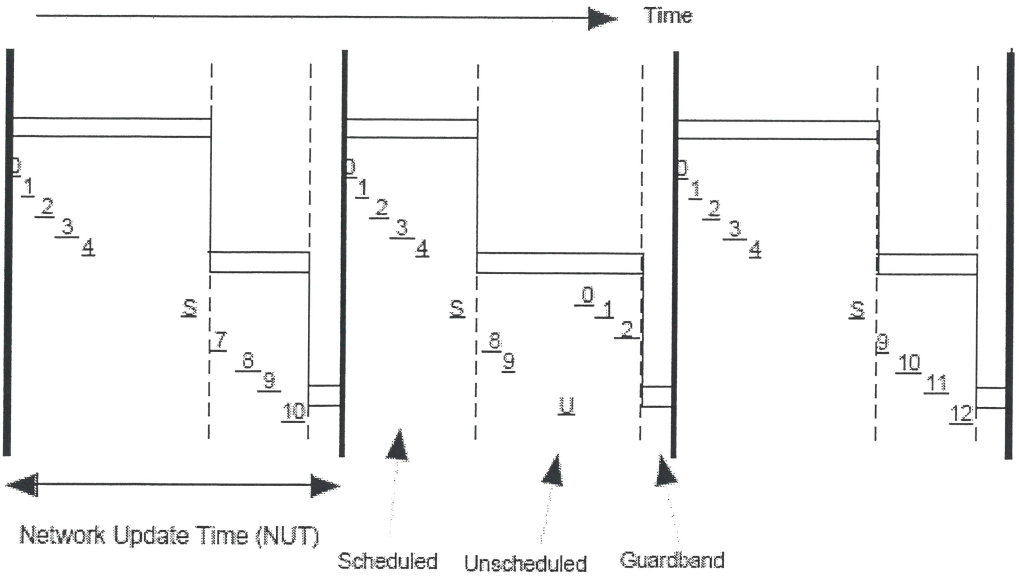

Figura 3.2.2. Accesul la mediu pe durata *scheduled*, *unscheduled* şi *guardband*

Pe durata scheduled a timpului NUT , fiecare nod poate transmite date critice în timp sau programate prin obținerea cheii de la 0 la S. Pe durata unscheduled a timpului NUT fiecare nod de la 0 la U îşi împart ocazia de a transmite date necritice în timp dupa metoda round-robin până când durata „de neprogramare" expiră. Când se ajunge la timpul de guardband, toate nodurile încetează să mai transmită şi numai nodul cu ID-ul MAC cel mai mic denumit „moderator" poate transmite un mesaj de întreținere denumit "cadru de moderare", care realizează sincronizarea tuturor ceasurilor din interiorul fiecărui nod şi publicarea tuturor parametrilor de link cum ar fi NUT, node time, S, U etc. Dacă frame-ul moderator nu este auzit după două NUT-uri consecutive, nodul cu cel mai mic ID de MAC va începe transmisia cadrului moderator în perioada *guardband* a celui de-al treilea NUT. Mai mult, dacă un nod moderator observă ca un alt nod are un ID de MAC mai mic decât al său el imediat îşi anulează rolul său de moderator.

3.2.1. Timpul de blocare la ControlNet

În ControlNet, dacă un nod vrea sa transmită un mesaj, el trebuie să aştepte să primească cheia de la nodul anterior logic. De aceea, timpul de blocare, T_{block} , poate fi exprimat de timpul de transmisie şi de timpul de rotire a cheii pe la nodurilor anterioare. Formula generală este pentru T_{block} poate fi descrisă de următoarea ecuație[4]:

$$T_{block} = T_{resid} + \sum_{j \in Nnoqueue} T_{token}^{(j)} + \sum_{j \in Nqueue} \min(T_{tx}^{(j,n_j)}, T_{node}) + T_{guard}$$

în care T_{resid} este timpul rezidual de care are nevoie primul nod pentru a termina transmisia, $N_{noqueue}$ şi N_{queue} denotă setul de noduri cu mesaje sau respectiv fără mesaje în coadă, şi T_{guard} este timpul scurs pe perioada guardband după cum s-a definit mai devreme.

Exemplu

De exemplu, dacă nodul 10 aşteaptă cheia, nodul 4 deține cheia şi transmite mesaje, iar nodurile 6,7 şi 8 au mesaje în coada lor, atunci $N_{noqueue}$ = {5,9} N_{queue} = {4,6,7,8}. Să admitem ca n_j reprezintă numărul de mesaje aşezate în coada la al j-lea nod şi ca T_{node} este timpul maxim posibil (adică timpul de deținere a cheii) asociat fiecărui nod pentru a utiliza pe deplin canalul de rețea; de exemplu în ControlNet T_{node} =827.2μs care este în funcție de mărimea maximă a datelor, mărimea cadrului de overhead şi de alți parametri ai rețelei. T_{token} este timpul de pasare a cheii, care depinde de timpul de care este nevoie pentru

transmiterea cheii şi de timpul de propagare de la nodul i-1 la nodul i. ControlNet foloseşte o cheie implicită şi T_{token} este practic suma între cadrul cu mărimea de date egală cu zero şi T_{prop}. Dacă un nou mesaj este aşezat în coada pentru a fi transmis în timp ce nodul deţine cheia, atunci $T_{block} = T_{tx}^{(j,n_j)}$, în care j este numărul nodului. În cel mai rău caz, dacă exista N noduri master pe bus şi fiecare are mai multe mesaje de transmis, ceea ce înseamnă ca fiecare nod foloseşte tot timpul în care deţine cheia, atunci

$T_{block} = \sum_{i \in N_{node} \setminus \{j\}} \min(T_{tx}^{(i,n_i)}, T_{node})$, unde funcţia min este folosită pentru că, indiferent dacă un nod are mai multe mesaje de transmis, el nu poate deţine cheia mai mult decât T_{node} (adică $T_{tx}^{(j,n_j)} \leq T_{node}$). ControlNet este o reţea deterministică deoarece întârzierea de timp maximă este limitată şi poate fi caracterizată de formula 3. Dacă perioadele fiecărui nod şi mesaj sunt ştiute, atunci putem descrie explicit setul $N_{noqueue}$ şi N_{queue} şi n_j. Astfel T_{block} din (3) poate fi determinat explicit.

Avantaje: Protocolul Token bus este un protocol deterministic care oferă o excelentă capacitate totală (throughput) şi eficienţă la încărcări mari ale reţelei[9]. În timpul funcţionării acesteia, se pot adăuga sau scoate dinamic noduri la busul ei, spre deosebire de cazul token ring, unde nodurile formează fizic un inel şi nu pot fi adăugate sau înlăturate dinamic. Segmentele de timp din fiecare ciclu NUT, în care se fac sau nu programări (scheduled/unscheduled) fac din protocolul ControlNet un protocol adecvat atât pentru mesajele necritice în timp, cat şi pentru cele critice.

Dezavantaje: Deşi protocolul token bus este eficient şi deterministic la mari încărcări ale reţelei, la trafic scăzut performanţa sa nu o poate atinge pe cea a protocoalelor care admit coliziuni(contention). În general, când sunt mai multe noduri într-un singur inel logic, un mare procent din timpul de reţea este folosit pentru a trece cheia pe la noduri atunci când traficul este scăzut [9].

3.3. DeviceNet (CAN Bus)

În reţelele care se bazează pe CAN datele sunt trimise şi recepţionate folosind Frame-uri de Mesaj care transportă datele de la un nod transmiţător la unul sau mai multe noduri receptoare. Nu este neapărat nevoie ca datele transmise sa conţină adresele sursei sau ale destinaţiei mesajului. În schimb, fiecare este etichetat de un identificator care este unic în toată reţeaua. Toate celelalte noduri care primesc mesajul pe care îl accepta sau îl rejectează în funcţie de configuraţia filtrelor de mascare pentru identificator. Acest mod de operare este cunoscut ca multicast.

DeviceNet, care se bazează pe specificaţia CAN, este o legătură de comunicaţie relativ ieftină, care leagă dispozitive într-o reţea. El a primit deja o recunoaştere largă în aplicaţiile de fabricaţie la nivel de dispozitiv. Specificaţia DeviceNet se bazează pe standardul CAN cu o aplicaţie suplimentară şi o specificaţie pentru nivelul fizic. Formatul de cadru prezent la DeviceNet este arătat mai jos [8].

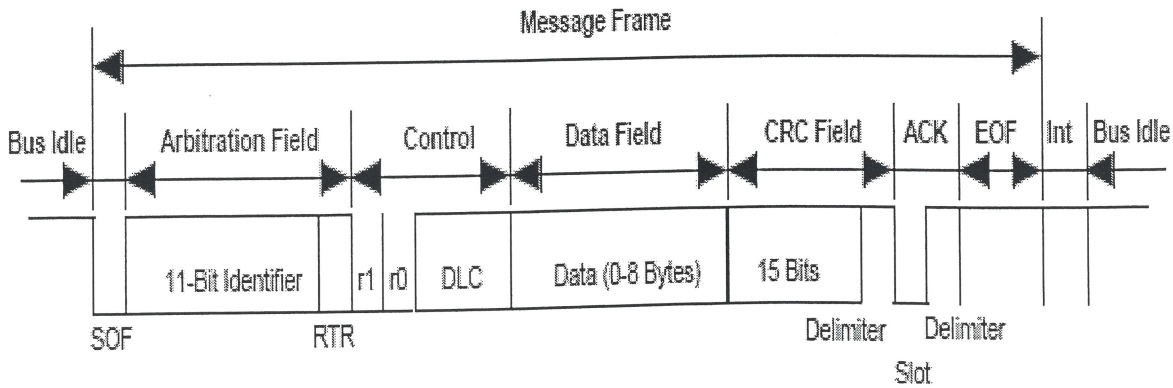

Figura 3.3.1. Formatul de cadru la DeviceNet

Overheadul total este de 47 biți, care include start of frame (SOF) câmpul de arbitrare (identificator de 11 biți),control, CRC, confirmare (ACK), sfârşitul cadrului (EOF) şi câmpul intermission (INT). Mărimea câmpului de date este cuprinsă între 0 şi 8 baiți. Protocolul DeviceNet foloseşte câmpul de arbitrare pentru a oferi adresarea sursei şi destinației şi prioritizarea mesajului.

3.3.1. Timpul de blocare pentru DeviceNet

Timpul de blocare, T_{block} , în DeviceNet poate fi descris de următoarea ecuație[6]:

$$T_{block}^{(k)} = T_{resid} + \sum_{\forall j \in N_{hp}} \left[\frac{T_{block}^{(k-1)} + T_{bit}}{T_{peri}^{(j)}} \right] T_{tx}^{(j)},$$

în care T_{resid} este timpul rezidual de care are nevoie nodul curent pentru a termina transmisia, N_{hp} este setul de noduri care au prioritatea mai mare decât nodul care aşteaptă, $T_{peri}^{(j)}$ este perioada celui de-al j-lea mesaj şi [x] reprezintă cel mai mic număr întreg care este mai mare decât x. Suma arată timpul de care este nevoie pentru a trimite toate mesajele cu prioritate mai mare. Pentru un nod cu prioritate mai mică, în timp ce el aşteaptă să devină disponibil canalul, este posibil să devină doritor de transmisie (să se aşeze la coadă) şi alte noduri cu prioritate mai mare şi, în acest caz, nodul cu prioritate mai mică va pierde din nou arbitrarea. În aceasta situație, se înregistrează timpul total de blocare. T_{resid} în cel mai rău caz la un trafic de rețea scăzut este:

$$T_{resid} = \max_{\forall j \in N_{node}} T_{tx}^{(j)},$$

în care Nnode este setul de noduri din rețea. în orice caz datorită mecanismului de arbitrare în funcție de prioritate, transmisia unui mesaj/de la un nod de prioritate scăzută poate să nu fie deterministică sau limitată în condițiile unei încărcături ridicate.

Avantaje: CAN este un protocol deterministic optimizat pentru mesajele scurte. Prioritatea mesajului este specificată în câmpul de arbitrare (identificator de 11 biți). Mesajele de prioritate mai mare câştigă întotdeauna accesul la mediu în pe timpul arbitrării. De aceea, timpul de întârziere a transmisiei mesajelor de mare prioritate poate fi garantat.

Dezavantaje: Marele dezavantaj al CAN în comparație cu alte rețele este rata de date lentă (maximum 500 Kbps). De aceea, capacitatea totală este limitată, în comparație cu alte rețele de control. Cerința de sincronizare de bit a protocolului CAN limitează, de asemenea, lungimea maximă a rețelei DeviceNet. De asemenea, CAN nu este adecvat pentru transmisia mesajelor cu un număr mare de date, deşi el suportă fragmentarea de date mai mari de 8 baiți.

3.4. O analiză grafică a celor trei rețele

În figurile 3.4.1 şi 3.4.2, sunt relevate câteva aspecte ale rețelelor de control Ethernet, ControlNet şi DeviceNet. Parametrii folosiți în aceste figuri sunt listați în:

Tabelul 1. Parametrii tipici de sistem pentru rețelele de control

	Eternet	ControlNet	DeviceNet
Rata de date[a] (Mbps)	10	5	0.5
Durata unui bit(μs)	0.1	0.2	2
Lungimea Max.(m)	2500	1000	100
Mărimea de date max. (byte)	1500	504	8
Mărimea mesajului min. (byte)[b]	72[c]	7	47/8[d]
Numărul max de noduri	>1000	99	64
Viteza tipică Tx (m/s)	Cablu coaxial: $2x10^8$		

unde:

a. rata de date tipică;

b. mărimea datelor 0;

c. incluzând câmpurile preambulul şi începutul delimitatorului;

d. Overhea-dul pentru DeviceNet este de 47 biți.

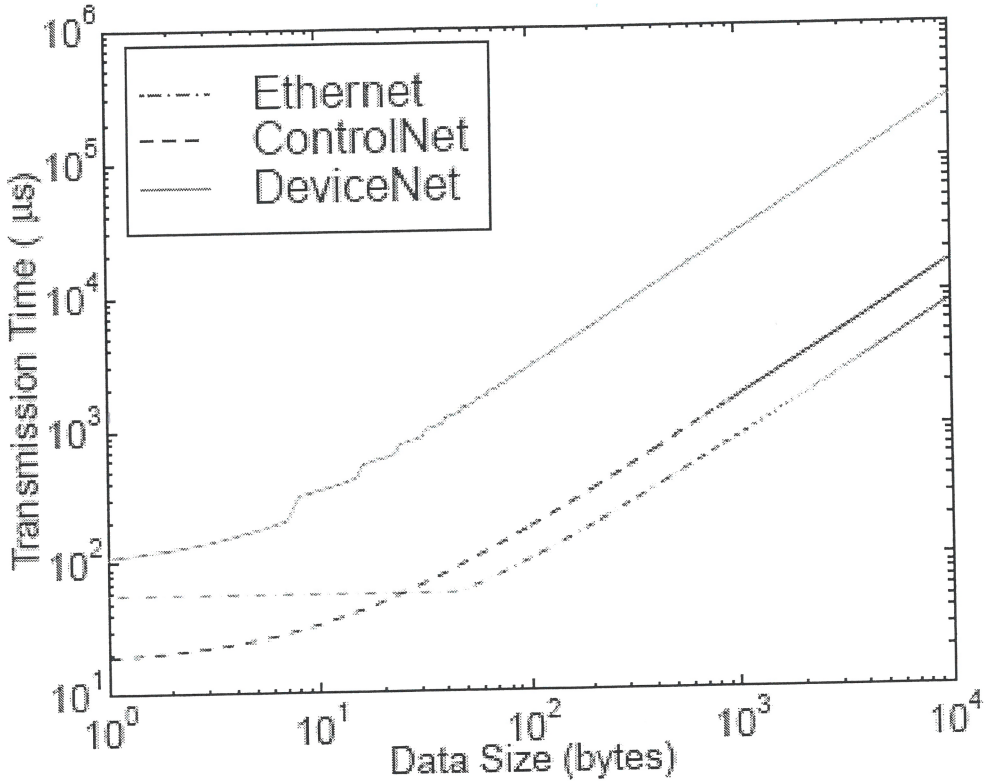

Figura 3.4.1. O comparație a timpilor de transmisie în funcție de mărimea datelor [4]

După cum se vede în figura 2.4.1, timpul de transmisie pentru DeviceNet este mai mare decât celelalte datorită ratei de date mai scăzute (500 kbps). Ethernet necesită mai puțin timp de transmisie la mărimi de date mai mari (> 20 byte) în comparație cu celelalte. Deşi ControlNet foloseşte mai puțin timp pentru a transmite aceeaşi mărime a datelor mici el are nevoie de un oarecare timp (NUT) pentru a câştiga accesul la rețea.

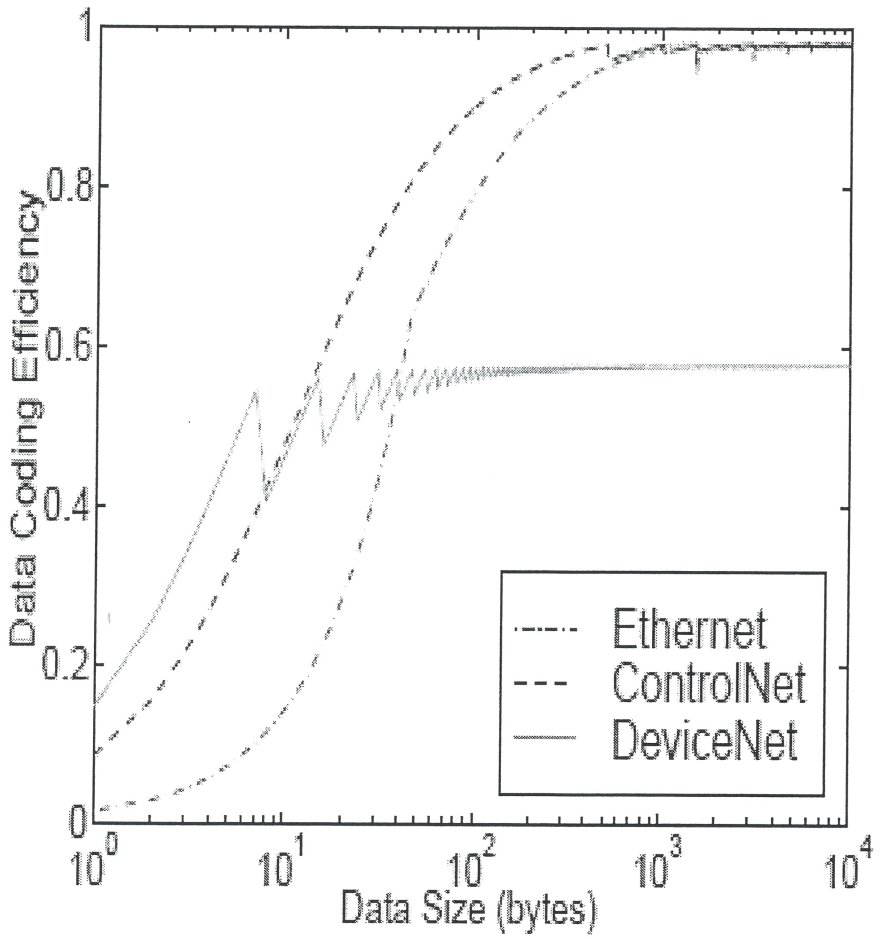

Figura 3.4.2. O comparație a eficienței de codare a datelor versus mărimea datelor [4]

Figura 3.4.2. arată eficiența codării datelor pentru cele rețele de control vizavi de mărimea datelor. Eficiența de codare a datelor se definește ca fiind raportul dintre mărimea datelor și mărimea mesajului (adică numărul total de baiți folosiți pentru a transmite date). Pentru mărimi mici de date, DeviceNet este cea mai bună dintre aceste trei tipuri de protocoale, iar Ethernet este cea mai nepotrivită. Pentru mărimi mari de date, ControlNet și Ethernet sunt mai bune decât DeviceNet (DeviceNet este eficiența în proporție de 58% în timp ce ControlNet și DeviceNet au o eficiență de 98% pentru transmisia datelor de mărime mare). Pentru sistemele de control dimensiunea datelor este, în general, mică. De aceea, analiza de mai sus sugerează că DeviceNet este de preferat în ciuda ratei scăzute de date. Oricum, înainte de a lua această, decizie trebuie investigate timpul de întârziere mediu și total, precum și capacitatea totală a rețelei.

4. Concluzie

Mecanismele de control al accesului la mediu, sunt răspunzătoare atât de satisfacerea cerințelor de răspuns în rețea în timp real/critic în timp, cât și a calității și siguranței comunicației dintre dispozitivele în rețea. Acești parametri de timp, care afectează aplicațiile de control, sunt afectați de rata de date din rețea, perioada mesajelor, de mărimea datelor sau a mesajelor de informație și de protocolul de comunicație.

Deși Ethernet se folosește mult în multe aplicații de transmisii de date și poate suporta o rată de date ridicată până la 1Gbps, ea nu este adecvată ca mediu de comunicație pentru unele sisteme de control atunci când este comparată cu sistemele de rețea deterministice.

La comunicația din interiorul unei mașini cu controllere, senzori și actuatori sunt mult mai potrivite sistemele de rețea deterministice pentru îndeplinirea cerințelor și caracteristicilor sistemelor de control.

Posibilitatea de a organiza mesajele *scheduled* și *unscheduled* la ControlNet îl fac pe acesta potrivit

pentru mesajele critice şi necritice în timp. ControlNet este, de asemenea, potrivit pentru transmisia mesajelor mari de date.

Datorită conceptelor sale de bază precum accesul la bus de tip multimaster şi prioritizat, o arbitrare care nu necesită distrugerea cadrului şi reluarea transmisiei şi o mare flexibilitate de configurare, CAN ofera cele mai bune performanţe pentru sistemele de control cu mesaje scurte şi prioritizate.

Bibliografie

1. **WOLFHARD, L.**: CAN Controller Area Network Grundlagen und Praxis, 2000.
2. * * *: Robert Bosch GmbH, CAN Specification, 1991.
3. **HELLWAGNER, H.**: Rechnerkommunication in eingebetteten Systemen: Lehrveranstaltungsleiter, Universitat Klagenfurt.
4. **FENG-LI LIAN 1, R. JAMES, MOYNE, DAWN M. TILBURY**: Performance Evaluation of Control Networks:Ethernet, ControlNet, and DeviceNet.
5. **TANENBAUM, A. S.**: Computer Networks, Upper Saddle River, NJ, Prentice-Hall Inc., Third Edition, 1996.
6. **TINDELL, K., A. BURNS, A. J. WELLINGS**: Calculating Controller Area Network (CAN) message response times," Control Engineering Practice, vol. 3, no. 8, pp. 1163{1169, Aug. 1995.
7. **LIAN, F.-L., J.R. MOYNE, D.M. TILBURY**: Performance Evaluation of Control Networks: Ethernet, ControlNet, and DeviceNet, Technical Report: UM-MEAM-99-02,
8. * * *: ControlNet Specifications, Boca Raton, Florida, ControlNet International, 1.03 Editions, 1997.
9. **KOUBIAS, S. A., G.D. PAPADOPOULOS**: Modern Fieldbus Communication Architectures for Real-time Industrial Applications. Computers in Industry, Aug. 1995, Vol. 26, No. 3, pp. 243-252.