最大遅れ時間解析によるスケーラブルCANプロトコルの性能評価

倉地 亮[†] 高田 広章[†]
西村 政信[††] 堀端 啓史[††]

現在，様々な機能を実現するために，車載ネットワークに流れるデータ量は急激に増加しており，広く採用されている Controller Area Network (CAN) では回線容量が不足している．しかしながら，Controller Area Network (CAN) の最大伝送速度はバス上での送信権の調停や ACK 応答メカニズムにより制約されており，最大伝送速度を向上するためにはプロトコルの改良が必要となる．そこで，我々は CAN を改良したスケーラブル CAN プロトコルを提案している．本論では，スケーラブル CAN プロトコルの最大遅れ時間解析手法を提案し，スケーラブル CAN プロトコルが CAN よりスループットを向上させつつ，CAN と同等の応答性を達成できることを示す．

Performance analysis of Scalable CAN with worst-case response time analysis

Ryo Kurachi ,[†] Hiroaki Takada ,[†] Masanobu Nishimura [††]
and Satoshi Horihata [††]

The Controller Area Network (CAN) is widely employed in in-vehicle networks to install new functions. However, CAN has significant restrictions on upper bound of maximum speed due to its bitwise arbitration and acknowledgement sequences. Therefore, we have proposed a new high-speed protocol "Scalable CAN" to improve upper bound speed of CAN. In this paper, we propose the worst-case response time analysis for "Scalable CAN". According to our analysis, the Scalable CAN has achieved much higher throughput performance and schedulability than CAN.

1. はじめに

現代の自動車では，多くの Electric Control Unit (ECU) が車載ネットワークを介して大量のデータを交換することにより，様々な制御や機能を実現している．車載ネットワークの中でも特に厳しいリアルタイム性が要求される制御系ネットワークでは Controller Area Network (CAN)[1] が広く使われている．CAN は自動車制御のようなリアルタイムシステムのための通信ネットワークに適したプロトコルであるが，その最大伝送速度が 1Mbps であるために X-by-Wire[2] のような将来の車載ネットワークシステムに対する要求を満たしているとはいえない．

そこで，これまでに CAN プロトコルを改良することにより高速化を実現する様々な研究が進められている[3)~5)]．しかしながら，これらの研究で提案される CAN を改良したプロトコルは物理的に 1 対 1 接続を採用するため，バス接続が主流である CAN を置き換えるにはゲートウェイなどの追加のハードウェアが必要になりコストが高くなることや，配線量が増えるために車重が増加することが問題である．そのため，我々はバス接続を可能とするスケーラブル CAN プロトコルを提案している[6]．

本論では，スケーラブル CAN プロトコルの伝送能力を評価するために，まず最大遅れ時間解析手法を提案する．次に，スケーラブル CAN プロトコルの性能評価として，提案する最大遅れ時間解析手法を用いて CAN と比較評価する．その結果，スケーラブル CAN が CAN よりも高いスループットを実現すると同時に，CAN と同等の応答性を達成できることを示す．

本論の構成は，2 章でスケーラブル CAN プロトコルの概要を説明し，3 章で解析の前提や定義を説明する．続く 4 章で，スケーラブル CAN メッセージの最悪状況を示し，5 章で最大遅れ時間解析手法を導出する．6 章では最大遅れ時間解析を用いてスケーラブル CAN の伝送能力を評価し，7 章で本論をまとめる．

† 名古屋大学大学院情報科学研究科附属組込みシステム研究センター
Center for Embedded Computing Systems,Nagoya University
†† 株式会社オートネットワーク技術研究所
AutoNetworks Technologies,Ltd.

2. スケーラブル CAN

本章では，我々が提案しているスケーラブル CAN プロトコルについて説明する．尚，本論では性能評価に必要となる機能に着目して説明するため，その他のプロトコルの詳細は文献 6) を参照されたい．

2.1 プロトコルの概要

スケーラブル CAN プロトコルは，最大伝送速度を 1Mbps とする CAN プロトコルのバス上での送信権の調停や ACK 応答メカニズムを改良することにより，その最大伝送速度を改善することを目的とするプロトコルであり，CAN 同様，OSI 参照モデルにおける Logic Link Control (LLC) および Medium Access Control (MAC) に位置づけられるプロトコルである．また，スケーラブル CAN は CAN から送信権の調停方法と ACK 応答メカニズムを改良しているため，バス上では CAN プロトコルとの互換性を持たない独自のバスプロトコルである．

このスケーラブル CAN ではソフトウェアからみて CAN コントローラと同じレジスタセットをもつ通信コントローラを用意するため，CAN のために作られたソフトウェアの移植が容易となる．FlexRay[7] などの他の次世代車載 LAN プロトコルと比較する場合，CAN からスケーラブル CAN へはアプリケーションが容易に移植できるため，導入時のソフトウェアの開発規模が抑えられる点が優位である．このため，もしユーザーが CAN の回線容量に不満がある場合には，**図 1** に示すように，通信コントローラやトランシーバ等のハードウェアを交換することで容易に帯域拡張することを可能にする．さらに，FlexRay ではプロトコルにて ACK をサポートしていないために，既存する CAN アプリケーションを FlexRay 上に移植する場合には，定常的に再送を行うことで送達を保証する必要があり，CAN メッセージの送信周期を見直す必要がある．具体的には，CAN で設定された送信周期内に 2 回あるいは 3 回送信するなどして，再送を保証する必要があり，実質的な回線容量が 1/2 倍や 1/3 倍になる懸念がある．一方，スケーラブル CAN ではプロトコルにて ACK をサポートしているため，このような見直しは必要なく CAN と同じアプリケーションを使用できる点でも優位である．

2.2 巡回型スケジューラ

スケーラブル CAN プロトコルは，CAN のようなバス上でのフレーム単位での送信権の調停は行わず，各 ECU にあらかじめ割り当てられた順番で送信権が巡回するスケジューリングを採用している．このため，

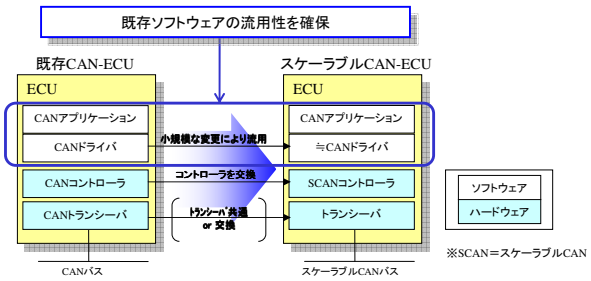

図 1 スケーラブル CAN のコンセプトと CAN からの移行イメージ
Fig. 1 Concept and migration image from CAN to Scalable CAN.

各 ECU はタイムスロットと呼ばれる送信時間が周期的に割り当てられ，その時間においてのみ送信を行うことができる．

各タイムスロットには 1 つのフレームのみを送信することを前提としており，各 ECU は自身が送信するタイムスロットを通信前に設定されるものとする．各 ECU は，自身の送信タイムスロットが回ってくるときに送信要求されたメッセージがある場合には，その送信要求されたメッセージのうち最も優先度の高いメッセージをそのタイムスロットに送信する．一方，送信要求されたメッセージが無い場合には，ACK フレームと呼ばれる最小長のフレームを送信することで，各 ECU 間のリンクを継続させる方法をとる．尚，もし ECU が故障しタイムスロットにフレームが送信されない場合には，その他の ECU はバスアイドル時間を検出すると，そのスロットをアイドルスロットと見なし次のスロットへと遷移することで，バス上での無効な時間を最小としつつ巡回を続けることができる．このように，各タイムスロット長は，そのスロットに送信するフレームの長さにより伸縮するものである．

1 サイクル内に存在するすべてのタイムスロットの合計数を $turn$ と呼び，あるタイムスロットを $\theta_i(1 \leq i \leq turn)$ で表す．各タイムスロットは小さい番号順 $(\theta_1 \rightarrow \theta_2 \rightarrow \cdots \rightarrow \theta_{turn})$ に実行されるものとする．前述するように各タイムスロット長はそのタイムスロットに送信されるフレーム長により伸縮するため，1 サイクル長（1 サイクル時間）もタイムスロット長に応じて伸縮するものとする．また，$turn$ 番目のタイムスロットの次は 1 番目のタイムスロットに戻ることで，システム起動中は各タイムスロットが常に巡回し続けるラウンドロビン型のプロトコルとなる．

2.3 送信メッセージの確定タイミング

送信メッセージの確定タイミングは，次に遷移するタイムスロットが自身の送信タイムスロットである場

合にそのタイムスロットの開始時刻において確定されるものとし，以降ではこの時刻を送信開始時刻と呼ぶ．

3. 解析の前提と定義

3.1 解析の前提

本論では，システム起動時に行われる各 ECU 間のタイムスロットの同期処理については扱わず，同期完了後の最大遅れ時間解析について議論する．また，バスに接続される各 ECU の故障時は考えないものとする．さらに，メッセージの送信要求に対するジッタは発生せず，送信要求はあらかじめ決められた時刻に必ず発生するものとする．また，CAN 同様，各メッセージは最大データ長を 8 バイトとする小さなフレーム単位で通信を行うため，バスに送信を開始すると送信完了まで送信動作を継続する仕様となる．このため，解析において，各メッセージはノンプリエンプティブとして扱う．尚，以降では，対象メッセージを送信する ECU を自 ECU と呼び，それ以外の ECU を他 ECU と呼ぶものとする．

3.2 メッセージセット

メッセージセットは，m 個の ECU(ECU_1,ECU_2,..., ECU_m) で構成されるものとし，各 ECU は複数の送信メッセージを持つものとする．ある送信メッセージ τ_i は，メッセージの優先度 i，そのメッセージの最大送信時間 C_i，送信周期 T_i，初回送信時のオフセット O_i の 4 つ組み (i, C_i, T_i, O_i) で表される．尚，CAN とスケーラブル CAN ではフレームの構成が異なるため，最大送信時間 C_i の導出方法は付録 A.1 を参照されたい．

3.3 オフセットとメッセージモデル

各メッセージの送信要求は，初回送信時に限りオフセットと呼ばれる各 ECU の起動時刻からの待ち時間後に実行されるが，以降は周期的に実行される．

1 つの ECU が送信するすべてのメッセージは同じタイマを用いて送信要求を行うため，同一 ECU から送信されるメッセージの送信要求時刻は同期しているものとして扱うことができる．一方，異なる ECU が送信するメッセージは，各 ECU で独立するタイマを用いて送信要求され，各タイマを同期する機構は無いため，同期していないものとして扱う．つまり，同一 ECU 内から送信するメッセージは，オフセットにより送信要求時刻を分散することで送信要求の衝突を回避し，送信までの待ち時間を減らすことが可能だが，他 ECU からは最悪状況で邪魔される可能性がある．

本論では，あるメッセージ τ_1 が $\tau_i = (1, 3, 12, 4)$ である場合，**図 2** のように図示するものとする．

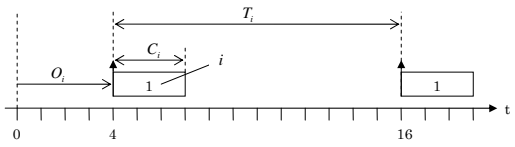

図 2 メッセージモデル
Fig. 2 Message model.

4. メッセージの最悪状況

本章では，スケーラブル CAN におけるメッセージの最大遅れ時間を解析するために，どのような状況においてメッセージの遅れ時間が最悪となるかを示す．以降では，まず，解析対象となるメッセージが他 ECU から受ける影響のみに限定した場合の最悪状況を示した後，自 ECU の送信メッセージから受ける影響も含めた議論を行う．

4.1 あるメッセージの滞留時間

あるメッセージが通信コントローラ内の送信メールボックスに格納されてから，その送信完了するまでの時間をメッセージの滞留時間と呼ぶ．解析対象となるメッセージの滞留時間には，そのメッセージが他の ECU から送信されるメッセージにより邪魔される時間を含むが，自 ECU から送信される他のメッセージによる邪魔時間は含まれない．以下では，メッセージの滞留時間の上限を与える定理を示し，その証明を行う．

定理 4.1 (メッセージの滞留時間) あるメッセージの滞留時間は，そのメッセージが送信メールボックスに格納されると同時に，自 ECU に与えられる送信タイムスロットの送信開始時刻を過ぎ，次の自 ECU の送信タイムスロットまでの間に存在するすべての他 ECU が送信するタイムスロットにおいて，最大となるメッセージの送信が発生する状況である．

証明 4.1 $turn = n$，つまり n スロットを 1 サイクルとして巡回するネットワークにおいて，各 ECU に 1 つずつ送信タイムスロットが割り当てられているものとする．このとき，解析対象となるメッセージ τ_n は，送信タイムスロット θ_n のある開始時刻を 0 とし，その満了時刻を t_0 とすると，次サイクルの送信タイムスロット θ_n の開始時刻 t' は次の式より導出できる．

$$t' = t_0 + \sum_{j=1}^{n-1} C_j \tag{1}$$

C_j は各タイムスロットにて送信されるメッセージ τ_j の最大送信時間を表す．
対象メッセージ τ_n がオフセット O_n を用いて，時刻 0 より後に送信要求される場合，式 4 で与えられる t'

よりも短い時間で次サイクルの θ_n に送信開始することができ，メッセージ τ_n の遅れ時間は t' より短くなる．このため，最長となる遅れ時間が発生するのは，$O_n = 0$ のときに送信要求される場合と仮定できる．また，もしメッセージ τ_n が時刻 0 よりも前に送信要求される場合には，τ_n は時刻 t_0 に終了するタイムスロットにて送信する機会が与えられることになり，メッセージ τ_n が時刻 0 に送信要求される場合よりも短い待ち時間になる．それゆえ，送信タイムスロットの送信開始時刻に送信要求される場合がクリティカルインスタントとなる．

4.2 自 ECU から送信されるメッセージの影響

あるメッセージの滞留時間を求めるために，そのメッセージが時刻 0 に送信メールボックスに格納されるという初期条件を用いている．この条件は，自 ECU から送信されるメッセージが 1 つしか存在しない場合には成立するが，複数のメッセージが同時に送信要求される場合には，自 ECU からそのメッセージよりも先に送信要求されたメッセージ（先行メッセージ）や，より後に送信要求されるメッセージ（後続メッセージ）に邪魔される可能性がある．この状況を扱うために，*busy period* の概念を導入する．

ある対象メッセージを解析する場合には，そのメッセージに影響を及ぼす先行メッセージや後続メッセージの影響を考える必要がある．ここで，先行メッセージあるいは後続メッセージに邪魔される状況を表 1 のメッセージセットを用いて具体的に説明する．

表 1 メッセージセット例
Table 1 Example of a message set.

ECU_i	τ_i	i	C_i	T_i	O_i	θ_l
ECU_1	τ_1	1	1	25	0	θ_1
	τ_2	2	2	25	5	
	τ_3	3	3	25	15	
ECU_2	τ_4	4	4	25	0	θ_2
	τ_5	5	5	25	7	
ECU_3	τ_6	6	6	25	0	θ_3

表 1 の例は，各 ECU に 1 つずつ送信スロットが割り当てられているものとし，$ECU_1(\theta_1) \rightarrow ECU_2(\theta_2) \rightarrow ECU_3(\theta_3) \rightarrow ECU_1(\theta_1) \rightarrow ...$ のように巡回される $turn = 3$ のネットワークである．また，各送信タイムスロットにおいて ACK フレームが送信される場合には，$C_i = 1$ とする．以降では，対象メッセージを τ_3 とする場合の遅れ時間について議論する．

4.2.1 先行メッセージに邪魔される例

まず，対象メッセージ τ_3 の先行メッセージである τ_2 の送信要求時刻から始まる状況が τ_3 の遅れ時間に影響を与える場合を議論する．先行メッセージ τ_2 の送信要求時刻 5 において，最大の滞留時間が発生する場合，まず最初の自 ECU の送信タイムスロット θ_1 にて ACK フレームが送信されるものとし，式 4 の $t_0 = 1$ とする．次に，ECU_2 は τ_5，ECU_3 は τ_6 の送信要求がクリティカルインスタントに発生し，続くタイムスロット θ_2 と θ_3 に τ_5 と τ_6 が送信されるものとすると，τ_3 は 5 と 6 だけ邪魔されることになる．この結果，τ_2 の送信要求時刻 5 から，$t = 1 + 5 + 6 = 12$ だけ送信された時刻 17 に，自身の送信タイムスロットである θ_1 の開始時刻が回ってくる．このとき，対象メッセージ τ_3 の送信要求時刻 15 を超えるため，先行メッセージ τ_2 が対象メッセージ τ_3 の送信に影響を与えることになる．このときの ECU_1 の送信状況と各スロットの送信状況を**図 3** に示す．以降の送信状況は，図 3 で示されるように，τ_2 が送信された後，θ_2 と θ_3 ではそれぞれ ACK フレームが送信され，τ_3 が時刻 21 で送信を開始し 24 で送信完了する．この結果，τ_3 の遅れ時間は送信完了時刻 24 から送信要求時刻 15 を引いた 9 となる．

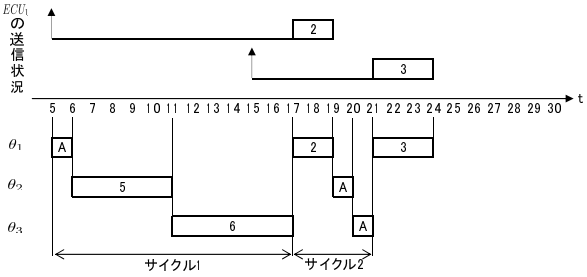

図 3 先行メッセージ τ_2 に邪魔される例 1
Fig. 3 Example 1 for interference from pre-requested message τ_2.

一方，**図 4** は図 3 の送信状況と比べ，θ_2 に送信する ECU_2 の送信状況が異なる場合であり，ECU_2 は τ_4 の送信要求時刻 0 がクリティカルインスタントに発生する状況を表している．

この状況では，最初の θ_2 では τ_4，次の θ_2 では τ_5 が送信されるため，図 3 よりも遅れ時間が大きくなる．このときの τ_3 の遅れ時間は，送信完了時刻 27 から送信要求時刻 15 を引いた 12 となる．この結果，図 3 よりも図 4 の方が，より遅れ時間が長くなる状況であるため，τ_3 にとっては図 4 がより最悪な状況といえる．

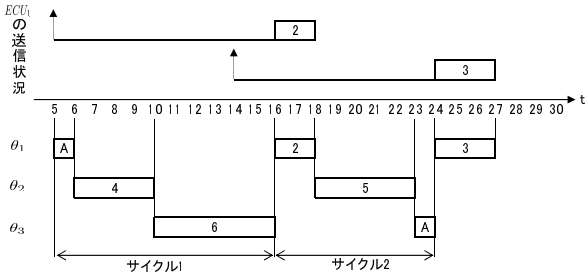

図 4 先行メッセージ τ_2 に邪魔される例 2
Fig. 4 Example 2 for interference from pre-requested message τ_2.

4.2.2 後続メッセージに邪魔される例

後続メッセージに邪魔される状況とは，対象メッセージの送信が遅れ，後続の優先度の高いメッセージの送信要求時刻を超える場合に，後続の優先度の高いメッセージが対象メッセージより先に送信する状況を指す．

この具体例について，表 1 の例を用いて説明する．τ_3 の送信要求時刻にて最大の滞留時間が発生する場合，**図 5** に示すように後続の τ_1 の送信要求時刻である時刻 25 を超えるため，τ_3 よりも優先度の高い τ_1 の送信が先に行われ，τ_3 の送信は次の送信タイムスロットまで遅延される．この結果，τ_3 の遅れ時間は，送信完了時刻 36 から送信要求時刻 15 を引いた 21 となる．

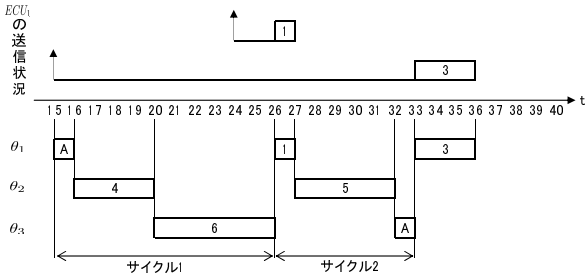

図 5 後続メッセージ τ_1 に邪魔される例
Fig. 5 Example for interference from post-requested message τ_1.

4.3 最悪状況の決定

与えられるクリティカルインスタントより，自 ECU が送信するすべてのメッセージの全送信要求時刻から最大遅れ時間を導出すれば，すべてのメッセージの影響時間を導出することができる．また，あるメッセージを対象メッセージとして着目する場合には，*busy period* の定義から前述する先行メッセージと後続メッセージが対象メッセージに影響を与えうる範囲のみを調べ尽くせばよいことになる．このため，あるメッセージを対象メッセージとする場合には，まずその送信要求時刻から最大遅れ時間を導出し，以降，対象メッセージの送信要求時刻に影響を与えうる先行メッセージの送信要求時刻に遡り，その各時刻から最大遅れ時間を計算することにより，*busy period* に存在するすべての最大遅れ時間の候補を導出することができる．その結果，すべての最大遅れ時間の候補の中で最も大きい最大遅れ時間が対象メッセージの最大遅れ時間となり，その状況が最悪状況と決定できる．

5. 解析アルゴリズム

本章では，他 ECU から送信されるメッセージ群が対象メッセージに与える最大邪魔時間の導出方法を定義した上で，ある送信メッセージの最大遅れ時間を計算するアルゴリズムを導出する．

5.1 Maximum Request Function

図 3 や図 4 に示すように，他 ECU が与える最大邪魔時間を効率的に計算することは難しい．そこで，スケーラブル CAN では，各送信要求時刻において最大で邪魔する可能性のあるメッセージの送信時間を導出するために，Maximum Request Function (MRF) を用いる．

MRF とは，ある ECU から送信されるメッセージ群が他の ECU に存在する対象メッセージを邪魔しうる最大影響時間を表す時間関数のことである．この MRF の導出方法は，CAN メッセージに Maximum Interference Function (MIF)[8] を適用した方法[9] と同様の方法ではあるが，各メッセージの送信要求時刻において，その送信時間を累積する時間関数として表される点が MIF とは異なる．

MRF の導出方法を簡単に説明すると，まず，あるECU に存在する対象メッセージよりも優先度の高いメッセージ群の送信周期の最小公倍数（LCM）までに存在するすべての送信要求時刻からの送信シーケンスを表す RF を作成する．次に，作成されたすべての RF を重ね合わせ，各時刻において最も邪魔時間が大きくなる状況が MRF である．尚，RF とは，優先度 i 以上のあるメッセージ τ_j の送信要求時刻 st から開始される送信メッセージ群の送信シーケンスを表す時間関数であり，ある時刻 t におけるその邪魔時間を $RF_j^{st}(i,t)$ で表すものとする．

この具体例として，表 1 に存在する ECU_2 の τ_4 と τ_5 を用いて，MRF の導出方法を説明する．まず，ECU_2 の τ_4 と τ_5 の送信周期の LCM は 25 であり，その中に存在するすべての送信要求時刻は τ_4 の時刻 0 と τ_5 の送信要求時刻 7 の 2 つである．この 2 つの送

信要求時刻から始まる RF は，**図 6** の下図に示す送信シーケンス ($RF_4^0(5,t)$ と $RF_5^7(5,t)$) であり，その邪魔時間を図 6 の上図のグラフで表現する．尚, 図 6 の横軸は各送信要求時刻からの相対的な経過時間を表し，上図の縦軸は送信メッセージによる邪魔時間を表す．

ある時刻 t における ECU_n の MRF は，各 RF の最も大きい邪魔時間をもつ線分を繋ぐことで導出でき，グラフの横軸を x，縦軸を y とする各座標 (x,y) の配列とその周期 T_{LCM} で表現すると，以下の式 2 で表すこともできる．

$$MRF_n(t) = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n), T_{LCM}\} \quad (2)$$

このため，式 2 より，図 6 の MRF は $MRF_2(t) = \{(0,5),(7,4),25\}$ と表記できる．

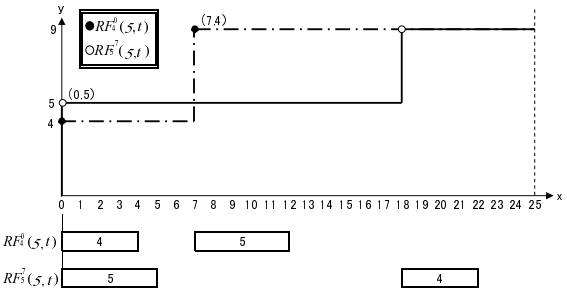

図 6 ECU_2 の MRF
Fig. 6 MRF in ECU_2.

5.2 スケーラブル CAN における他 ECU が最大で邪魔する時間の導出方法

スケーラブル CAN の各タイムスロットでは 1 つのメッセージのみしか送信できないため，各タイムスロットの最大邪魔時間を導出するために，メッセージ数を限定した MRF を検討する．具体的には，表 1 に存在する ECU_2 の τ_4 と τ_5 を用いて説明する．まず，ECU_2 における 1 つのメッセージの MRF を導出する場合，各 RF の送信シーケンスを 1 メッセージに限定することで，1 つのメッセージから邪魔される最大時間を導出できる．より具体的には，**図 7** に示すように，各 RF の送信シーケンスにおいて先頭の 1 メッセージのみに限定して導出する．その結果，この 1 メッセージ分に限定された MRF には周期性はないため，周期を 0 とし，$MRF_2(t)^1 = \{(0,5),0\}$ と表すものとする．また，同様に 2 メッセージ分の MRF は図 6 と同様の送信シーケンスをとるが，その周期を 0 とし，$MRF_2(t)^2 = \{(0,5),(7,4),0\}$ と表される．

以降では，4.2.2 節の具体例を用いて，ECU_2 が送信するメッセージ τ_4 と τ_5 が対象メッセージ τ_3 を最大で邪魔する時間を導出する方法を示す．

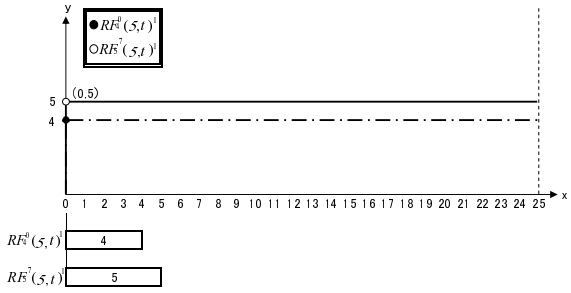

図 7 ECU_2 に存在する 1 メッセージ分の MRF
Fig. 7 MRF for a message in ECU_2.

まず，対象メッセージ τ_3 の送信要求時刻 15 にて最悪状況が発生するものとし，解析の開始時において，対象メッセージ τ_3 の遅れ時間 $WCRT$ と ECU_2 の送信タイムスロット数 $sendcnt_2$ と ECU_2 の累積的な邪魔時間 $iftime_2$ を 0 に初期化する．次に，初回送信タイムスロット θ_1 に $t_0 = 1$ だけ邪魔された後，続く θ_2 の開始時刻 16 はクリティカルインスタント発生後 $16 - 15 = 1$ だけ経過した時間であるため，$MRF_2(t)^{sendcnt_2+1}$ の x 軸方向に 1 だけ経過したときの y 軸の値 (y_c) を読み取り，もし y_c と $iftime_2$ が異なる場合にはその時刻までの邪魔時間 y_c と $iftime_2$ との差分をそのスロットの邪魔時間として遅れ時間 $WCRT$ に加算する．一方，y_c と $iftime_2$ が同じ値である場合には，前回の送信スロット以降に送信要求されたメッセージが存在しないため，ACK フレームが送信されるものとして，その最大送信時間を遅れ時間 $WCRT$ に加算する．

4.2.2 節の具体例においては，初回送信タイムスロット θ_2 では，$sendcnt_2 = iftime_2 = 0$ であるため，$MRF_2(t)^1$ の x 軸方向に 1 だけ経過したときの y 軸の値 $y_c = 5$ となり，y_c と $iftime_2$ の差分がある場合として解析できる．このとき，$MRF_2(t)^1$ の $x = 1$ の $y_c = 5$ だけデータが送信されるものとして，$iftime_2$ との差分 $5 - 0 = 5$ を遅れ時間 $WCRT$ に加算する．また，このときに $sendcnt_2$ は 1 つインクリメントされ $sendcnt_2 = 1$，メッセージの邪魔時間 $iftime_2 = 5$ を代入し保持する．

この送信状況は**図 8** に示すように，次の θ_2 が開始される時刻 28 は時刻 15 から相対的に 13 だけ経過した時間となる．このとき，$sendcnt_2 = 1$ より，2 メッセージ分の MRF である $MRF_2(t)^2$ を用いて，x 軸方向に 13 だけ経過したときの y 軸の邪魔時間 $y_c = 9$

を取得する．このときの $iftime_2$ は 5 であるため，y_c と $iftime_2$ との差分 $9-5=4$ だけ送信が行われるものとして遅れ時間 $WCRT$ に加算する．この結果，メッセージ τ_3 の最大遅れ時間は $36-15=21$ であり，図 5 とは θ_2 の送信順序が異なるものの，最悪状況と同じ最大遅れ時間をえることができる．

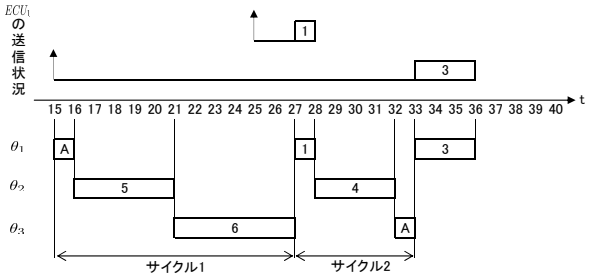

図 8 提案手法の具体例
Fig. 8 Example for our proposed analysis.

5.3 最大遅れ時間解析アルゴリズム

ある 1 つのスケーラブル CAN メッセージの最大遅れ時間解析アルゴリズムを以下に示す．

(1) 対象となるメッセージを送信する自 ECU に割り当てられるある送信タイムスロットを初回送信タイムスロットとする．

(2) 対象メッセージの初回送信要求時刻をクリティカルインスタントとする．

(3) 初回送信タイムスロットにて，CAN と同様に 1 つの低優先度メッセージが送信するものとし，自 ECU 内に低優先度メッセージが存在する場合には，その中で最長となる送信時間をもつメッセージの長さを t_0 とする．一方，もし低優先度メッセージが存在しない場合には，ACK フレームの最大送信時間を t_0 とする．

(4) 次のタイムスロット以降，そのタイムスロットが他 ECU が送信する場合は 5.2 節で提案する方法により，そのタイムスロットに送信する ECU の MRF を用いて最大送信時間を導出し，遅れ時間に加算する．一方，そのタイムスロットが自 ECU の送信タイムスロットである場合，その時刻までに送信要求されたメッセージのうち最も優先度の高いメッセージの送信時間が遅れ時間に加算される．このとき，対象メッセージが送信されるまで本手順 (4) を繰り返し，最大遅れ時間の候補を導出する．

(5) 以降，自 ECU の全送信メッセージの送信周期の LCM 内に存在する対象メッセージとそれに影響を与える先行メッセージの各送信要求時刻をクリティカルインスタントとし，手順 (3) に戻りすべての場合の最大遅れ時間の候補を求める．

(6) 自 ECU に複数の送信タイムスロットがあり，未だ初回送信タイムスロットとして解析されていないものがある場合，それを初回送信タイムスロットとし，手順 (2) に戻り，全クリティカルインスタントに対し解析を行う．その結果，すべての最大遅れ時間の候補のうち，最も大きい遅れ時間をもつ候補が最大遅れ時間となる．

6. 評　　価

6.1 評 価 対 象

本評価では，前述するスケーラブル CAN の最大遅れ時間解析手法を用いて，CAN とスケーラブル CAN プロトコルの伝送能力の違いを比較する．本評価では，まず，伝送効率の違いを比較するため，伝送速度とメッセージ量が同じ場合の比較を行う．次に，スケーラブル CAN では CAN と比べて最大伝送速度が改善されることから，伝送速度に応じたデータ量を送信した場合の比較を行う．尚，本論で比較対象として用いる CAN メッセージの最大遅れ時間解析手法は，文献 9) で提案されたオフセット付き CAN メッセージの最大遅れ時間の解析手法に対し，スケーラブル CAN と同等の解析条件とするため，ECU 内の優先度逆転の考慮を削除した解析方法を用いる．

本評価では既存する CAN バス上の ECU がスケーラブル CAN へと移行する場合を想定し，実際の車両で使われているメッセージセットから送信メッセージの負荷率の合計が大きい 6 つの ECU を抽出し，それら 6 つの ECU が 1 つのバス上に配置されるものを用いる．尚，送信メッセージの負荷率とは，ある送信メッセージ τ_i の最大送信時間 ECU_i をその送信周期 C_i で割り百分率で表した値である．この抽出されたメッセージセットは全 48 メッセージ，CAN500kbps におけるバス負荷率は約 55%であり，このメッセージセットのメッセージ量を 1 倍量と呼ぶ．本評価では各メッセージにオフセットを付与する場合と付与しない場合 (つまり，オフセットが 0 の場合) の 2 パターンのメッセージセットを用いて評価を行う．

また，評価指標は全メッセージの最大遅延率の平均とする．最大遅延率とは，最大遅れ時間解析からえられる最大遅れ時間をそのメッセージの送信周期で割り百分率で表した値である．この最大遅延率を使う理由は，各メッセージの送信周期はメッセージ毎に様々な値に設定されるため，メッセージセット全体を評価す

るためには，全メッセージの最大遅延率の平均を用いる方法が適切と考えるためである．

6.2 CAN とスケーラブル CAN の比較

6.2.1 スケーラブル CAN のスロット割り当て方法

本評価では，以下の 3 種類の送信スロットの割り当て方法を用いた．尚，本評価で用いた具体的なスロットの配置については以下の表 2 に示すとおりである．

(A) 各 1 スロット式 各 ECU あたり 1 スロットずつ割り当てる方法であり，1 サイクルあたりのスロット数は ECU 数と同様，6 スロットとなる．

(B) ドント式 1 サイクルあたりのスロット数を 16 とし，各 ECU に 1 スロットずつ割り当てた上で，残りの 10 スロットは各 ECU の送信メッセージの負荷率の合計に応じて分配する．

(C) 最短送信周期のメッセージ数式 メッセージセットの中で送信周期の短いメッセージの最大遅延率が大きくなることが予想できる．このため，メッセージセット中の最短周期となる送信メッセージの数に応じて，各 ECU にスロットを付与する．尚，1 サイクルあたりのスロット数は 22 スロットとなる．

表 2 本評価で用いるスロット割り当ての設定内容
Table 2 Settings of assigned time-slot for Scalable CAN.

	(A) 方式 各 1 スロット式	(B) 方式 ドント式	(C) 方式 最短送信周期のメッセージ数式
ECU1	1	1,7,11,15	1,7,11,15,19
ECU2	2	2,8,12,16	2,8,12,16
ECU3	3	3,9,13	3,9,13,17,20,22
ECU4	4	4	4
ECU5	5	5	5
ECU6	6	6,10,14	6,10,14,18,21
合計☆	6 スロット	16 スロット	22 スロット

6.2.2 CAN とスケーラブル CAN の比較結果

ここで，オフセットを付けない場合と付けた場合の評価結果を表 3 と表 4 に示す．

表 3 オフセットが無い場合の従来 CAN とスケーラブル CAN の最大遅延率の平均の比較
Table 3 Comparisons of the worst-case response delay avarage for CAN messages and Scalable CAN messaes without offsets.

伝送速度	メッセージ量	オフセット	CAN	スケーラブル CAN		
				(A) 方式	(B) 方式	(C) 方式
500kbps	1 倍量	なし	20.80(%)	32.89(%)	31.68(%)	32.68(%)
5Mbps	10 倍量	なし	19.16(%)☆☆	27.83(%)	21.04(%)	24.97(%)

表 3 の結果より，オフセットの無い場合には，各プロトコルで伝送速度が同じ場合を比較すると，スケーラブル CAN の方が巡回するためのオーバーヘッドや

☆ 1 サイクルあたりのスロット数を表す．
☆☆ 実際には CAN を 5Mbps で実行することはできないが，仮に 5Mbps で実行できるものと仮定した場合の参考値である．

表 4 オフセットがある場合の従来 CAN との最大遅延率の平均の比較
Table 4 Comparisons of the worst-case response delay avarage for CAN messages and Scalable CAN messaes with offsets.

伝送速度	メッセージ量	オフセット	CAN	スケーラブル CAN		
				(A) 方式	(B) 方式	(C) 方式
500kbps	1 倍量	あり	10.83(%)	17.71(%)	14.68(%)	16.61(%)
5Mbps	10 倍量	あり	1.84(%)☆☆	2.21(%)	2.24(%)	2.52(%)

フレームの拡張により，CAN と比べて相対的に最大遅延率の平均は長くなる．一方，伝送速度に応じてメッセージ量を増やした場合の比較として，CAN の伝送速度を 500kbps，メッセージ量を 1 倍量とした場合の結果である 20.80(%) と，スケーラブル CAN の伝送速度を 5Mbps，メッセージ量を 10 倍量とした場合の結果を比較すると，スケーラブル CAN の (B) 方式でスロットを割り付けた場合には 21.04(%) であり，ほとんど CAN と同程度となる．これは伝送速度が上昇したことにより優先度逆転の影響が減少するため，最大遅延率が改善されたことを示している．

また，表 4 の結果より，オフセットがある場合には，伝送速度が同じ場合を比較すると，オフセットが無い場合と同様に CAN の方がスケーラブル CAN よりも最大遅延率の平均は良い結果となる．一方，伝送速度に応じてメッセージ量を増やした場合の比較では，メッセージ量が 10 倍量に増加した場合でも，各メッセージがオフセットにより分散して送信要求されるため，各メッセージが対象メッセージの送信を阻害する時間を大きく減少することができる．このため，伝送速度 5Mbps，メッセージ量 10 倍量のときには CAN の伝送速度 500kbps，メッセージ量 1 倍量と比べ，大きく最大遅延率を削減する結果となった．この結果より，我々のプロトコルは従来 CAN を高速化した場合とほぼ同程度の高い能力を示しているといえる．

6.3 スロット割り付け方法とオフセットの効果

本節では，スケーラブル CAN におけるオフセットとスロット割り当て方法の効果について考察する．

まず，オフセットの効果は，表 3 と表 4 のオフセットありなし時の各結果を比較すると，どの設定においてもオフセットを付与した方が明らかに遅延率が低減できているといえる．

次に，3 種類のスロット割り当て方法の有効性は，表 3 のオフセット無し時においては，500kbps での結果が示すように多少の有効性は確認できるもののあまり効果は確認できなかった．一方，表 4 のオフセットあり時においては，500kbps での結果が示すように，単純に各 ECU に 1 タイムスロットずつ割り当てた (A) 方式よりも，(B) あるいは (C) 方式の方が，全体とし

て明らかに遅延率を低減できているといえる．一方，5Mbps の場合には，オフセットが付与されていれば，ほとんど送信タイムスロットの割り付け方法に関係なく，最大遅延率が低減できている．

これらの結果より，スケーラブル CAN においては，送信タイムスロットの割り付け方法により多少最大遅れ時間が低減されるものの，より最大遅れ時間を低減するためには，オフセットと送信タイムスロットの割り付け方法の両方を用いて最適化する必要があるといえる．

7. ま と め

我々は新しい高速な車載ネットワーク向けプロトコルとして，スケーラブル CAN を提案している．本論では，スケーラブル CAN の性能評価として，最大遅れ時間解析手法を提案し，CAN との伝送能力の差を比較し評価した．本論での評価結果より，スケーラブル CAN では従来 CAN を高速にした場合と同程度の遅延率で伝送速度に見合う十分なメッセージ量の送信を行えることを示した．

参 考 文 献

1) International Organization for Standardization, Road vehicles. Controller area network (CAN). Part1:Data link layer and physical signaling, ISO IS11898-1, (2003).
2) Thomas Nolte, Hans Hansson, and Lucia Lo Bello : Automotive communications - past, current and future, Proceedings of 10th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA '05), Vol.1, pp. 985-992, (2005).
3) G. Cena, A. Valenzano: FastCAN: a high performance enhanced CAN-like network, IEEE Trans. Ind. Electron., vol.47, No.4, pp. 951-963, (2000).
4) B.Terry Compton: Goldilocks Serial Communication Protocol, SAE World Congress, (2008).
5) 倉地亮，高田広章，手嶋茂晴，宮下之宏：10MbpsCAN プロトコルの設計と評価，情報処理学会論文誌：Vol.50，No.11，pp.2643-2653，(2009)．
6) Ryo Kurachi, Hiroaki Takada, Masanobu Nishimura and Satoshi Horihata: A New High-Speed Bus Topology LAN Protocol Compatible with CAN, SAE 2011 World Congress, (2011).
7) FlexRay Consortium. http://www.flexray.com/.
8) Takada, H. and Sakamura, K.: Schedulability of generalized multiframe task sets under static priority assignment, Proc. Real-Time Computing Systems and Applications, pp. 80-86, (1997).
9) 飯山真一，冨山宏之，高田広章，城戸正利，細谷伊知郎： オフセット付き CAN メッセージの最大遅れ時間解析，情報処理学会論文誌：コンピューティングシステム，Vol.45， No.SIG11(ACS 7), pp.455-464，(2004)．

付　　録

A.1 従来 CAN とスケーラブル CAN の最大送信時間の差

従来 CAN とスケーラブル CAN のデータフレームの差は文献 6) で示したとおりである．このため，従来 CAN とスケーラブル CAN ではフレームの最大送信時間が異なる．

ここで 1 ビット当たりの送信時間を τ_{bit}，データ長 (1〜8 バイト) を s_i とすると，CAN メッセージの最大送信時間は以下の式 3 を用いて導出できる．

$$C_i = \left(\left\lfloor \frac{35 + 8s_i - 1}{4} \right\rfloor + 46 + 8s_i \right) \tau_{bit} \quad (3)$$

一方，スケーラブル CAN の最大送信時間は，メッセージが送信されるタイムスロットの最大時間とみなして解析するため，ACK 情報フィールド長を $acklen$，スロット番号フィールド長を $slotlen$，スロット境界からの送信開始時刻を $acplen$，伝送路上の許容する伝送路遅延ビット数を $delay$ とすると，1 フレームの送信にかかるタイムスロットの最大時間は以下の式 4 を用いて導出する．

$$C_i = \left(\left\lfloor \frac{34 + acklen + slotlen + 8s_i - 1}{4} \right\rfloor + 45 + 8s_i + acklen + slotlen + acplen + delay \right) \tau_{bit} \quad (4)$$

本論の評価では，$acklen$ を表 2 の 1 サイクルあたりのスロット数で設定し，$acplen = 5$ ビット，$slotlen = 5$ ビット，$delay = 2$ ビットとして計算している．また，ACK フレームの最大送信時間は式 4 を $s_i = 0$ として導出するものである．