Vol.2019-ARC-235 No.3
Vol.2019-SLDM-187 No.3
Vol.2019-EMB-50 No.3
2019/3/17

時刻配送型CANネットワークの提案

倉地 亮[1,a)]　高田 広章[1,b)]　足立 直樹[2,c)]　上田 浩史[2,d)]　宮下 之宏[2,e)]

概要：現在，Controller Area Network(CAN) は車載制御システムで広く使用されているネットワークプロトコルである．CAN プロトコルでは，CAN-ID を用いた優先度ベースの送信権の調停方式であるため，遅れ時間を見積もることは容易であるものの，各 CAN メッセージの生成時刻を知る術はない．このため，本論文では各ノード間でグローバル時刻の同期をすることなく，各メッセージの生成時刻を通知する方式を提案する．

Delivery Delay Mechanisms for Controller Area Network

1. はじめに

自動車の電子制御システムは車載制御システムと呼ばれ，1 台の自動車に 100 個近い Electronic Control Unit(ECU) と呼ばれる制御用のコンピュータが配置されており，車載制御ネットワークを介して，各センサ情報が共有されるような分散制御型のシステムである．車載制御ネットワークでは，Controller Area Network(CAN) プロトコルが広く使用されているが，今後は CAN with Flexible Data-Rate(CAN-FD) などのプロトコルが搭載されることが予想されている [1], [2], [4], [5].

CAN プロトコルの最大の特徴として，1 つの CAN バス上に配置される複数の ECU から同時に送信したいメッセージが存在する場合，CAN メッセージに付与される CAN-ID を用いて，送信権の調停が行われる点が挙げられる [6]．この送信権の調停の結果，優先度の最も高いメッセージが必ず 1 つ送信されることが保証される一方，優先度の低いメッセージの送信は遅延される．

特に自動車の制御システムでは，より高度な最適制御を実現するには，CAN メッセージの受信タイミングの揺らぎが制御の安定性や予測可能性に影響を与えていることが課題となる．さらに，制御システムのセキュリティ上の要件として，各制御メッセージのフレッシュネス (鮮度) を保証することが考えられている [7], [8]．しかしながら一方で，CAN のようなイベントトリガ型のネットワークプロトコルを使用する場合には，各メッセージのフレッシュネスを保証することが難しいという課題がある．そこで本論文では，CAN メッセージのフレッシュネスを保証するための手法として，遅延時間の配送を実行するための CAN 通信コントローラを提案する．この提案する CAN コントローラを用いて，各メッセージの滞留時間を各送信メッセージに付与して通知することにより，フレッシュネスを保証することを実現する．

本論文の構成は，以下のとおりである．まず，第 2 章では，研究対象である車載制御ネットワークやその課題について説明し，第 3 章では提案手法について述べる．つづく，第 4 章では実装された CAN コントローラについて述べ，第 5 章では評価について述べる．最後に，第 6 章で本論をまとめる．

2. 車載制御ネットワーク

2.1 Controller Area Network(CAN)

Controller Area Network は Bosch により提案され，ISO11898 により国際標準化されたプロトコルである．**図 1** に示すように，車載制御ネットワークは，1 つの CAN バス上に複数の制御用コンピュータである Electric Control Unit (ECU) が接続されるバス型ネットワークで構成されることが多い．さらに，これらの CAN バスを複数束ねる

[1] 名古屋大学大学院情報学研究科
Nagoya University, Graduate School of Informatics
[2] 株式会社オートネットワーク技術研究所
AutoNetworks Technologies ltd.,
[a)] kurachi@nces.i.nagoya-u.ac.jp
[b)] hiro@ertl.jp
[c)] adachi-naoki@sei.co.jp
[d)] ueda-hiroshi@sei.co.jp
[e)] miyashita-yukihiro@sei.co.jp

セントラルゲートウェイ (CGW) と呼ばれる ECU が CAN バス間のメッセージのルーティングを実施することにより実現されることが多い.

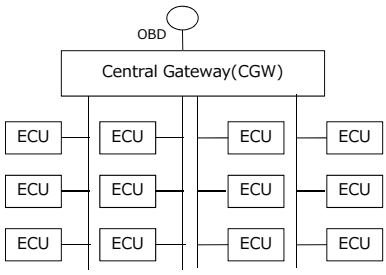


図 1 想定する車載制御ネットワーク

2.2 CAN-FD

CAN-FD は，2015 年に Bosch により提案された CAN の改良プロトコルであり，その特徴は 2 つある．まず 1 つ目に，1 メッセージの最大ペイロード長が拡張され，最大 64 バイトまで転送できるようになった．次に 2 つ目として，送信権の調停後の転送速度を向上するビットレート・スイッチが採用された点である．

これらの 2 つの技術を適用することにより，複数の CAN メッセージにより配送されていた CAN メッセージを 1 つの CAN-FD メッセージにて実現することができる．さらに，ビットレート・スイッチにより，最大転送速度を引き上げることにより，拡張されたペイロード領域を用いても，十分に短い転送時間で配送できることになった．

2.3 フレッシュネスに関するセキュリティ脅威

近年，実際に販売される自動車に対するセキュリティ上の脅威が報告されており，特に深刻な事例として，CAN バス上になりすましメッセージを転送することにより，車載電子制御システムの制御を乗っ取る方法が指摘されている．

このとき，攻撃者は，CAN バス上に流れる CAN メッセージを取りためておき，以降の任意の時刻に送り付ける再送攻撃 (リプレイ攻撃) や偽メッセージを偽造して転送することにより実現する．このような手法により，ブレーキの無効化などが実現できることが指摘されている [9]．このため，CAN のセキュリティ強化手法が様々提案されている [10], [11], [12], [13] ものの，フレッシュネスを保証する手法については存在しない．

2.4 関連研究

CAN のようなイベントトリガ型プロトコルにおいて，各メッセージの遅延時間を知ることは難しい．また，各ノードで時刻同期を実施する場合には，時刻同期をする場合，通常は IEEE1588 のようなアプリケーション層での時刻同期プロトコルを用いて時刻を同期する方法がある．この場合には，アプリケーション間で同期を実行するためのメッセージを用いて同期することが考えられる．

一方，ネットワークプロトコルにおいて時刻同期を実行するとなると，時刻同期型のネットワークプロトコルである．Time-Triggered Protocol (TTP) [14] や FlexRay[15] などのプロトコルに変更する必要がある．CAN でも Time-Triggered CAN (TT-CAN) プロトコル [16] が提案され標準化されたものの，現在では通信コントローラを入手することは難しい．

さらに CAN 通信コントローラを使いつつ，アプリケーション側で同期をとる仕組みとして，FTT-CAN[17] が提案されている．しかしながら，タイムトリガプロトコルと同様，タイムマスターノードからの同期フレームを受信すると各スレーブノードがそのスケジュールに合わせてメッセージを転送する方式を採用している．

CAN メッセージの最大遅れ時間解析手法としては，これまでに多数の研究がなされている [18], [19], [20]．しかしながら一方で，実システム上で各 CAN メッセージの遅延時間を収集する手法については現在までに提案されていない．

3. 提案手法

本章では，まず CAN メッセージの生成時刻を生成するための課題を整理し，アイデアと提案手法の詳細について述べる．

3.1 生成時刻取得の課題

CAN では，各メッセージの優先度を示す CAN-ID により，CAN ネットワーク上で優先度順に送信される．このため，前述するように優先度の低いメッセージが遅延された結果，いつメッセージが生成されたのかがわからなくなるという課題がある．

より具体的に，メッセージモデルで説明すると，以下の図 2 の通りである．CAN プロトコルでは，ある送信メッセージ m_i は任意の時刻に送信要求時刻 r_i に送信要求を実行する．そして，CAN ネットワーク上に流れる送信開始時刻 t_i^{start} に送信開始され，時刻 t_i^{fin} に送信完了する．

このため，各時刻に送信要求されたメッセージ m_i の遅延時間 ΔT_i は，$t_i^{fin} - r_i$ と表すことができる．各送信メッセージは，遅延なく送信できる場合がある一方，バス上が込み合う場合には最大遅れ時間まで遅延するため，各メッセージの送信要求時のネットワークの状況によりばらつくものである．

3.2 生成時刻取得のアイデア

送信ノードにて遅延時間を知るためには，$t_i^{fin} - r_i$ の時刻を知る必要がある．仮にこの時刻をアプリケーションで知るためには，例えば時刻 r_i にハードウェアカウンタを起

Vol.2019-ARC-235 No.3
Vol.2019-SLDM-187 No.3
Vol.2019-EMB-50 No.3
2019/3/17

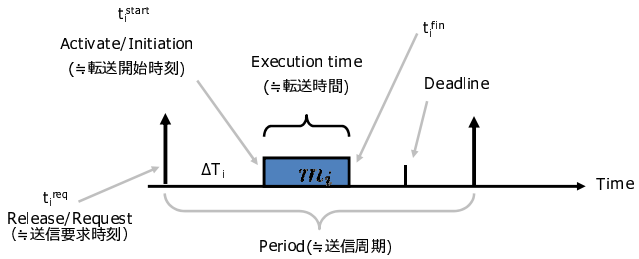


図 2 CAN メッセージの送信モデル

動し，送信完了割り込み時刻に遅延時間をキャプチャする方法がある．しかしながら，この方法では送信しているフレームではなく，別のフレームで時刻を転送する必要があるため，制御としては役に立たない[*1]．

このように，実際に転送されるメッセージの遅延時間を通知する場合，ソフトウェアの処理で実行するには限界がある．

このため，提案手法は，CAN コントローラを改造することにより，実際に遅延している時刻を配送する手法を提案する．より具体的には，前述するような送信要求時刻 r_i から転送完了する時刻 t_{start} までの時刻をハードウェアカウンタにより計測し，実際に転送されている時刻 $t_{fin} - t_{start}$ は，事前に計算された時刻により生成する方法により時刻を生成し通知することを検討する．

図 3 に示すように，遅延時間計測のため，CAN 通信コントローラの送信メールボックスに対して，遅延時間計測用カウンタを追加する．より具体的には，このカウンタには CAN バス上の 1 ビット時間単位のクロックが与えられ，各メッセージの送信権確定までの遅延時間を計測するものであり，以下の 3 ステップの処理で実現される．

(1) 送信要求時に遅延時間計測用カウンタを起動
(2) 送信権獲得時にカウンタ値を取得し，送信フレーム内のペイロードの指定された位置に遅延時刻を付与
(3) 転送完了時にカウンタを停止

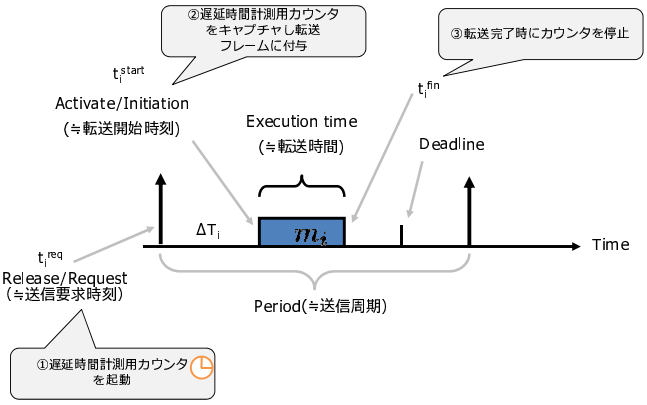


図 3 遅延時間計測のアイデア

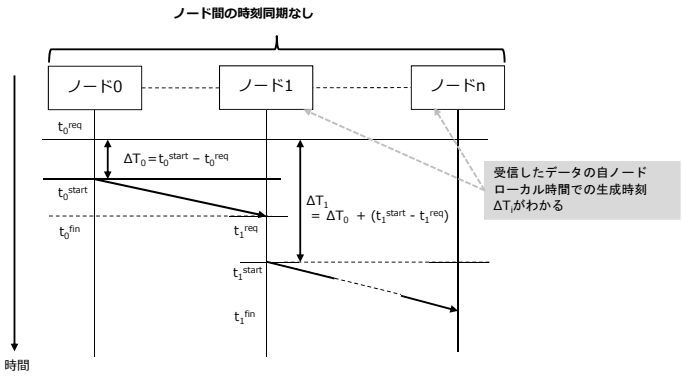
[*1] 例えば，リアルタイム制御が必要なアプリケーションでは即時性も要求されることが多い

3.3 中継先ネットワークにて生成時刻を知る方法

前節では，各 CAN メッセージの遅延時間の計測手段について提案した．本節では，前述するように，図 4 を用いて，CAN ネットワーク全体に遅延時間を伝播する手段について述べる．

図 1 に示すように，車載制御システムでは複数の CAN バスにまたがり CAN メッセージが中継される．前述する我々の提案手法を改良し，ゲートウェイが受信した遅延時間を中継処理に要した時間だけ追加することができれば，中継先ネットワークにおいても遅延時間を知ることができる．

より具体的には，ゲートウェイに到着するメッセージをゲートウェイの中継先も前節で本論文が提案する CAN 通信コントローラが搭載されており，中継先ネットワークへの送信要求時に遅延時間計測用カウンタの初期値を設定することにより実現する．この初期値の設定方法は，CAN メッセージの送信要求時刻において，(a) CAN-ID, (b) メッセージのペイロード長を示す DLC, (c) ペイロードの 3 つを設定した上で送信要求することから，ペイロード内の遅延時間を配送する該当位置 (例えば，ペイロードの 1 バイト目など) に初期値として設定することにより遅延時間が引き継がれる．



t_i^{req}	データ生成時刻（各ノードのローカル時間）
t_i^{start}	送信開始時刻（送信権獲得時の各ノードのローカル時間．）
t_i^{fin}	送信側での送信完了時刻（送信ノードのローカル時間だが，CANプロトコル上は受信側の t_i^{req} 時刻と同じ
ΔT_i	送信要求開始からの経過時間．データ生成時刻（＝送信要求時刻から送信監視までの経過時間．送信権獲得後，この時刻を転送フレーム内に埋め込み転送する）

図 4 中継先メッセージへの遅延時間の伝播方法

4. 実装方法

実現方法としては，CAN 通信コントローラの各送信メールボックスに対して，前述する遅延時間計測用カウンタの値を付与することにより実現した．

5. 評価

本評価では，前節で実装された CAN 通信コントローラ

Vol.2019-ARC-235 No.3
Vol.2019-SLDM-187 No.3
Vol.2019-EMB-50 No.3
2019/3/17

が正しく動作することを確認する実験を行った．さらに，時刻同期プロトコルとの比較について考察を行った．

5.1 実装評価

実装された CAN 通信コントローラの評価として，以下の 2 つの実験を行った．まず，1 つ目の実験として，シミュレータ上で遅延時間計測用カウンタが正しく実装されていることを確認した．次に 2 つ目の実験では，実機上へ実装された CAN 通信コントローラにより，実環境において遅延時間が伝播されていることを確認した．

5.1.1 実験 1: シミュレータ上での動作確認

まず 1 つ目の実験については，実装された CAN 通信コントローラが正しく動作することを確認するために，シミュレータ上で，遅延がない場合と遅延される場合の 2 つの場合についてそれぞれ確認した．図 5 は遅延しない場合の結果であり，遅延しない場合でも送信要求時刻から送信権確定までの時刻 (この図の場合，20BT) がペイロードの 0 バイト目と 1 バイト目に配送されていることを確認した．また，図 6 は遅延する場合の結果であり，2 ノードが同時にメッセージを送信要求した場合を示している．この場合，先に送信権を得たノードは，33BT の遅延時間，また遅延されたメッセージでは 133BT の遅延時間をペイロードに付与して転送することを確認した．

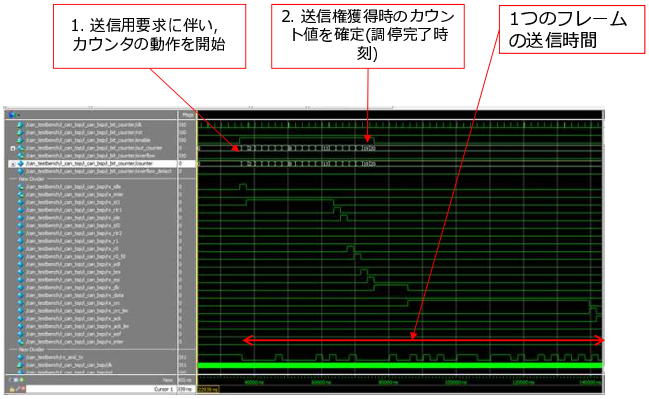


図 5 遅延時間がない場合

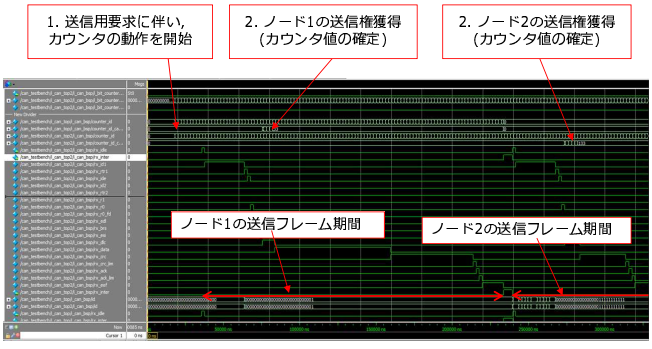


図 6 同時に送信要求し，1 つのメッセージに遅延される場合

5.1.2 実験 2: FPGA ボード上での動作確認

前節で実装された CAN 通信コントローラを FPGA ボード上に実装した．FPGA ボードはインテル社の DE0-NANO を用いており，FPGA ボード Quartus13.0sp1 を使用した．リソースの消費量は，図 7 に示すとおりであり，最小セットの構成ではロジック数が 67LEs，レジスタ数が 32 個増加するのみで実装可能であった．

	合成結果（追加前）	合成結果（追加後）	増加量
Total Logic Elements (LEs)	19073	19140	67
Total registers	12460	12492	32

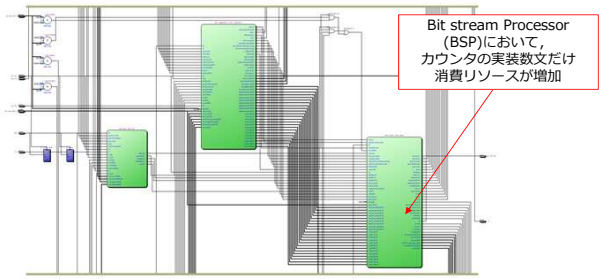


図 7 改造された CAN 通信コントローラの合成結果と最小セットでの増加分

実機上の評価では，市販するアナライザを使用し，エラー環境においても正しく遅延時間が配送されるかを評価した．この結果，遅延のない場合には通常 15BT で遅延するフレームに対して，1 ビット目にてビットエラーを発生させると，(i) エラーフレーム送信時間 (14BT) と (ii) インターミッション時間 (2BT) だけ増加された 25BT で転送されることを確認した．

これらの結果より，正しく生成時刻を配送できることを確認した．

5.2 各方式の比較

次に，本方式の優位性を示すために，他の時刻同期プロトコルへの実装方法との違いを表 1 に示す．本比較では，関連研究で挙げたタイムトリガプロトコル (TT-CAN/FlexRay, FTT-CAN) への移行を想定する場合と，本論で提案手法を適用した場合の比較を想定する．このとき，いずれの方式も各メッセージの生成時刻を取得できるものの，プロトコル，ハードウェア，ソフトウェアなどを変更する必要があることが課題である．特に，本提案手法では他のプロトコルへの移行に比べて，通信スケジュールの変更規模やソフトウェアの改変規模が小さい一方で，改造された CAN 通信コントローラが必要であることが課題である．一方，増加するオーバーヘッドについては，いずれの方式でもスケジューリングに依存したり，基準時刻や時刻配送が必要となるため，増加することは避けられないといえる．

表 1 関連研究との比較

Items	TT-CAN, FlexRay	FTT-CAN	本提案手法
CAN プロトコルの維持	× (変更必要)	○ (変更不用)	○ (変更不用)
既存ハードウェアの改造	× (必要)	○ (不要)	× (必要)
ソフトウェアの改変 (スケジューリング再設計)	× (必要)	× (必要)	○ (不要)
ノード間の同期 (マスターノードの有無)	× (必要)	× (必要)	○ (不要)
増加する通信オーバーヘッド	△ (スケジューリング次第)	△ (時刻同期用通信)	△ (付与する遅延時刻)

6. まとめ

本論文では，CAN ネットワーク内にて正確なメッセージ生成時刻を配送するための手法について提案した．今後の課題として，周期送信メッセージの揺らぎがどれぐらいであるかの誤差の評価やゲートウェイを超える中継メッセージの遅延時間配送の評価が挙げられる．さらに，配送できる遅延時間の上限はペイロード内での遅延時間の上限値に依存するため，通信オーバーヘッドと遅延時間の上限のトレードオフの決定方法を議論する必要がある．

謝辞 本研究は JSPS 科研費 16K16025, 18K11212 の助成を受けたものである．

参考文献

[1] Leohold, J.. Communication Requirements for Automotive Systems, 5th IEEE Workshop on Factory Communication Systems, 2004.
[2] S. Saidi, S. Steinhorst, A. Hamann, D. Ziegenbein, M. Wolf, "Special Session: Future Automotive Systems Design: Research Challenges and Opportunities", 2018 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), Oct 2018.
[3] A. Ferre, J. Fontanilles, R. Serret, CarlosFernandez , "E-Systems for Automated Driven Vehicles" , Electric/Electronic in Hybrid and Electric Vehicles and Electrical Energy Management, May 2017.
[4] M. Lunt. E/e architecture in a connected world, 2017.
[5] DI STEFANIE PYKA, ROBERT BOSCH AG WIEN, future of mobility, https://www.a3ps.at/sites/default/files/conferences/2014/papers/5_bosch_pyka.pdf, Bosch, 2014.
[6] International Organization for Standardization, Road vehicles - Controller area network (CAN) - Part 1: Data link layer and physical signaling, ISO11898-1, 2003.
[7] Benjamin Glas, Jens Gramm, Priyamvadha Vembar, Security Levels and System Security Requirements for System Integrity of Automotive ECUs, Embedded Security in Cars Conference (escar EU 2014), pp.1-9, Nov 2014.
[8] Specification of Module Secure Onboard Communication (SecOC) AUTOSAR Release 4.2.2, http://www.autosar.org/fileadmin/files/releases/4-2/software-architecture/safety-and-security/standard/AUTOSAR_SWS_SecureOnboardCommunication.pdf, 2014.
[9] C. Valasek, C. Miller, " Adventures in Automotive Networks and Control Unit" , http://www.ioactive.com/pdfs/Ioactive_Adventures_in_Automotive_Networks_and_Control_Unts.pdf, 2014
[10] O. Hartkopp, C. Reub, R. Schilling. MaCAN - Message Authenticated CAN. In Embedded Security in Cars Conference, escar 2012, Berlin, Germany, November 28-29, 2012.
[11] J. Pieprzyk, A. Sadeghi, M. Manulis. LiBrA-CAN: A Lightweight Broadcast Authentication Protocol for Controller Area Networks. pages 184-200. In 11th International Conference, CANS 2012, Darmstadt, Germany, December 12-14, 2012.
[12] Kurachi, R., Matsubara, Y., Takada, H., Adachi, N., Miyashita, Y., and Horihata, S., " CaCAN - Centralized Authentication System in CAN" , Proceedings of the escar 2014 Europe Conference, Hamburg, Germany, Nov 2014.
[13] Ryo Kurachi, T.David Pyun, Shinya Honda, Hiroaki Takada, Hiroshi Ueda, Satoshi Horihata, "CAN Disabler: Hardware-based Prevention method of Unauthorized Transmission in CAN and CAN-FD networks" , Embedded Security in Cars Conference (escar US 2016), pp.1-7, Detroit, Jun 2016.
[14] H. Kopetz and G. Grunsteidl, "TTP - A time-triggered protocol for fault-tolerant real-time systems," FTCS-23 The Twenty-Third International Symposium on Fault-Tolerant Computing, Toulouse, pp. 524-533, 1993.
[15] FlexRay Communications System Protocol Specification Version 3.0.1, https://svn.ipd.kit.edu/nlrp/public/FlexRay/FlexRay%E2%84%A2%20Protocol%20Specification%20Version%203.0.1.pdf, 2019.
[16] International Organization for Standardization, Road vehicles – Controller area network (CAN) – Part 4: Time-triggered communication, ISO11898-4, 2004.
[17] L. Almeida, P. Pedreiras and J. A. G. Fonseca, "The FTT-CAN protocol: why and how," in IEEE Transactions on Industrial Electronics, vol. 49, no. 6, pp. 1189-1201, Dec. 2002.
[18] Tindell, K. and Burns, A.: Guaranteed Message Latencies For Distributed Safety-Critical Hard Real-Time Control Networks (1994).
[19] Davis, R., Burns, A., Bril, R. and Lukkien, J.: Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised, Real-Time Systems, Vol. 35, pp. 239-272 (2007), 10.1007/s11241-007-9012-7.
[20] Yong Xie, Gang Zeng, Yang Chen, Ryo Kurachi, Hiroaki Takada, Renfa Li "Worst Case Response Time Analysis for Messages in Controller Area Network with Gateway", IEICE Transactions 96-D(7), pp.1467-1477 Aug 2013.