CAN（Controller Area Network）を利用した論理的通信モデル実現のためのプロトコル

佐藤健哉[†] 小板隆浩[†] 井上博之[††]

組み込みリアルタイムシステムは，自動車，家電製品，産業用機器など幅広い分野に適用されている．CAN（Controller Area Network）は，自動車用途に開発され，高速応答性，高信頼性が必要なアプリケーション分野におけるセンサやアクチュエータを制御するための，標準データリンクプロトコルとして利用されている．本研究では，CAN を利用して組み込みアプリケーションを容易に実現するための論理的通信モデルの構築と，優先度制御をサポートしたデータ転送プロトコルの設計・評価を行った．論理的通信モデルとデータ転送プロトコルを利用することで，下位のネットワークをほとんど意識することなく，効率的に組み込みシステム向けアプリケーションを実現することができる．CAN 仕様のデータフレームは，29 ビットのヘッダと最大 8 バイトのデータから構成され，上位のプロトコルを構築するには制約の大きい構造となっているため，本論文では，データ転送プロトコルを効率的に CAN のデータフレームへのマッピングする方法も記述した．

Network Protocols for Logical Communication Model on CAN (Controller Area Network)

Kenya Sato,[†] Takahiro Koita[†] and Hiroyuki Inoue[††]

Embedded real-time systems are deployed in a wide range of application domains such as transportation systems, automated manufacturing, home appliances and telecommunications. Originally developed for use in automotive applications, CAN (Controller Area Network) has been adopted by industry as the standard network technology to transmit data for sensors and actuators, due to its fast response and high reliability for applications. We design and implement a logical communication model and a real-time data transmission protocol for embedded systems with CAN. This logical communication model with prioritized data transmission is effective for developing application programs in an embedded system, which results in less consideration of underlying network mechanisms and the high portability of applications. Owing to the small size of the CAN frame format, there are many restrictions on the implementation of a network protocol on CAN. In an attempt to solve this problem, we describe a concrete implementation of the network protocols in detail to realize the logical communication model a CAN 8-byte data frame with a 29-bit identifier.

1. はじめに

自動車，家電製品，産業用機器など幅広い分野において組み込みリアルタイムシステムが適用されている．一般に，組み込みシステムは多種多様であるが，半導体技術の進歩とデジタル化，ネットワーク化にともない，その構成は複雑になり，高度なものは一種の分散システムの様相を呈している．これにともない，ソフトウェア設計も大規模，複雑化する傾向にあり，モジュール化，コンポーネント化を行うことで，開発工程の短縮，信頼性の向上が行われている[1]．CAN（Controller Area Network）[2] は，自動車用途に開発され，高速応答性，高信頼性が必要なアプリケーション分野におけるセンサやアクチュエータを制御するためのネットワークの標準として利用されている[3]．CAN の普及にともない，CAN コントローラがマイコンに内蔵され，安価に実現できるため，従来の制御用途に加え，自動車用マルチメディアやテレマティクスのアプリケーションにも利用され始めている[4),5]．

本研究では，マルチメディア，テレマティクスを対象とした組み込みシステム向けの論理的通信モデルを構築し，CAN のデータリンク層の上に論理的通信モデルを実現するためのデータリンク層とトランスポー

† 同志社大学工学部情報システムデザイン学科
Deptartment of Information Systems Design, Doshisha University
†† 株式会社 IRI ユビテックユビキタス研究所
Ubiquitous Laboratories, IRI Ubiteq, Inc.

ト層の通信プロトコルを新たに設計し，実装と評価を行う．論理的通信モデルでは，アプリケーションの概念として，論理ユニットを定義し，論理ユニットの識別子をもとに，アプリケーションが通信を行う形態をとる．これにより，アプリケーションは，どの機器（物理ユニット）に搭載されていても，機器を意識することなく通信を行うことが可能となる．CAN のフレームフォーマットは 29 ビットの ID と最大 8 バイトのデータで構成されるため，一般的に，仕様の複雑なプロトコルをこの上に構成することは困難である．そのため，上位のプロトコル仕様をコンパクトにし，CAN のフレームフォーマットの上に効率良く搭載することが必要となる．本論文では，この論理的通信モデルを CAN 上に実装するための，具体的なフレームフォーマットについても記述する．

2. 関 連 研 究

CAN を利用した組み込み向けの分散システム[6),7)] およびプロトコル[8)] などが提案されているが，ここでは，特に汎用機器接続を目的とした通信プロトコルという観点から，IP over CAN と，IDB-C について取り上げる．

2.1 IP over CAN

IP over CAN[9)] は，IP データグラムを CAN 上で伝送するための仕様であり，INTERNET DRAFT として提案された経緯がある．IP over CAN では，(1) CAN 上のデバイスへのアドレッシング，(2) CAN ID の構造，(3) CAN ノード間の IP アドレスの割付け，(4) IP データグラムのフラグメントを記述している．

CAN 上に IP を効率良く実現できれば，インターネットにおけるアプリケーションのフレームワークが利用でき，用途が広がる．しかし，CAN のデータフレームの PDU（Protocol Data Unit）は 128 ビットであり，一方，IP パケットはヘッダのみで 20 から 60 バイトになり，効率良く伝送することができない．また，IP over CAN では，通信開始時に情報交換フレームが流れるため，これらを総合すると，データ転送の実効速度は 10%以下となる．

2.2 IDB-C

自動車用マルチメディア，テレマティクスのアプリケーションとして，ITS データバス（IDB）が SAE（Society of Automotive Engineers）の標準として検討されている[10)]．**図 1** に IDB のネットワーク構成を示す．自動車メーカ独自のバスと IDB がゲートウエイで接続されており，IDB のネットワーク側に IDB デバイスが接続される．IDB デバイスは，標準化されたバスインタフェースを持つため，どの自動車メーカのネットワークに対しても接続が可能となる．自動車メーカ独自のバスは，自動車メーカごとに仕様が異なるが，ゲートウエイを通して，IDB 側から標準化された手順でアクセスすることが可能となっている．

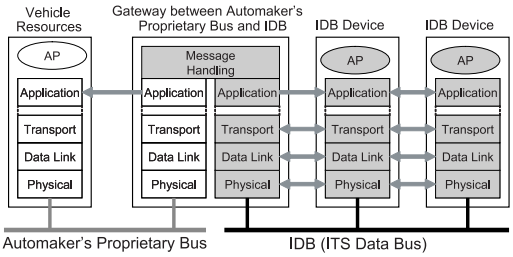

図 1 IDB ネットワークアーキテクチャ
Fig. 1 Network architecture for IDB.

この IDB を CAN 上で実現した仕様が IDB-C である．IDB-C は，SAE において，物理層（J2366-1）[11)]，データリンク層（J2366-2）[12)]，トランスポート層（J2366-4）[13)]，アプリケーション層（J2366-7）[14)] が標準化されているが，(1) レスポンス時間が長い，(2) データ転送効率が悪い，(3) CPU の負荷が高い，(4) 論理的識別子を利用したアプリケーションの構成が困難，といった問題点がある．

3. 通信モデル

本研究におけるネットワークアーキテクチャとなる基本的な通信モデルは，図 1 で示す構成と同様である．それぞれの物理的なコンポーネント（たとえば図中の IDB デバイス）は，単一のネットワークインタフェース，物理層，データリンク層，トランスポート層，アプリケーション層のプロトコルスタック，および，その上で動作するアプリケーションプログラム（AP）から構成される．ここでは，アプリケーションプログラムモジュールの抽象化を論理ユニット，物理的なコンポーネントを物理ユニットと定義する．論理ユニットはその論理識別子としての論理アドレスを持ち，物理ユニットは物理的な識別子として物理アドレスを持つ．図 1 では，1 つの物理ユニット内に 1 つの論理ユニットが存在している例となっているが，たとえば，1 台のオーディオユニット内に，ラジオ，CD，MD の 3 つのアプリケーションが動作する場合のように，1 つの物理ユニット内に複数の論理ユニットが存在する場合もある．

プロトコルスタックにネットワーク層が含まれていないのは，一般的な組み込みシステムでは，複雑なネットワーク構成をとることがなく，かつ，セキュリ

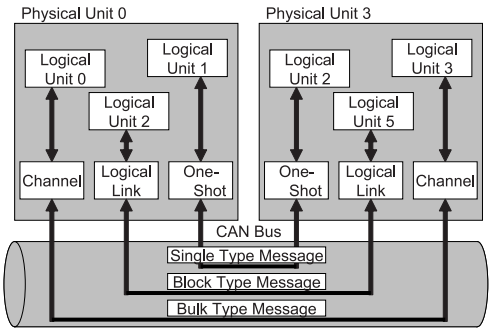

図 2 論理的通信モデルの構成
Fig. 2 Structure of logical communication model.

ティの確保が必要であり，複数のネットワークをルータで接続するのではなく，プロキシ（アプリケーションゲートウエイ）を通して行うためである．

3.1 論理ユニット

論理的通信モデルでは，論理アドレスにより，他のユニット内のアプリケーションと，直接通信を行うことができる．それぞれのアプリケーションは，それ自身が搭載されているコンポーネント，あるいは，下位のネットワークの階層を意識することなく設計が可能となり，アプリケーションを他のコンポーネントに容易に移動させることもできる．また，論理アドレスを利用することで，(TCP/UDP におけるウェルノウンポートのように）車輪速センサ，エアバッグ用加速度センサ，ドアロックアクチュエータなどといったサービスの種類を特定することにも利用でき，特別なサービスディスカバリ機構がなくても，アプリケーションはサービスを利用することが可能となる．

論理ユニットと物理ユニットの関係，および，通信形態を**図 2** に示す．この例では，2 つの物理ユニットが CAN によるネットワークで接続されている．それぞれの物理ユニット内に 3 つの論理ユニットが存在する．物理ユニット 0 内には，論理ユニット 0，1，2 があり，物理ユニット 3 内には論理ユニット 2，3，5 がある．物理ユニット 0 内の論理ユニット 1 は，物理ユニット 3 内の論理ユニット 2 と，仮想的に接続されており，アプリケーションは，逐一，相手方の物理アドレスを確認することなく通信することが可能である．論理的通信モデルを用いてメッセージの優先度制御を実現するためには，アプリケーション層の通信プロトコルばかりでなく，トランスポート層やデータリンク層のプロトコルにおいても，論理ユニット間の仮想的通信を認識し，制御する必要がある．

3.2 伝送方式

一般に，組み込みリアルタイムシステムでは，ネットワーク上を伝送するメッセージの緊急性がそれぞれ異なり，緊急性の高いメッセージは小さなレイテンシで伝送されることが求められる．たとえば，自動車のアプリケーションにおいては，ハンドル操作，アンチロックブレーキ，エアバッグなどの情報を送るメッセージは，他のメッセージ（たとえば，窓の開閉や，ヘッドライトのオン・オフを制御するためのメッセージ）と比較して緊急性が高いといえる．また，運転記録や故障診断のログデータなどは緊急に転送する必要がないが，メッセージのサイズが大きくなるため，効率的に伝送することが求められる．したがって，メッセージの伝送は，アプリケーションごとに要求が異なり，通信の機構において，異なる要求のメッセージを分類して低レイテンシ，高スループットをサポートする必要がある．

図 2 で示す通信モデルでは，メッセージの伝送を，それぞれの要求に応じて，「シングル」，「ブロック」，「バルク」の 3 つのタイプに分類する．シングルタイプのメッセージはワンショットとして，単一のフレームで伝送されるが，ブロックタイプメッセージおよび，バルクタイプメッセージはそれぞれ論理的なリンクとなる通信路を構成し，ブロックタイプメッセージの論理的通信路を「論理リンク」，バルクタイプメッセージの通信路を「チャネル」と呼ぶ．ワンショットとしてのシングルタイプメッセージは，メッセージサイズが小さく，最も高い優先度を持ち，最低のレイテンシで伝送される．論理リンクを通るブロックタイプメッセージは，中程度の優先度で，中程度のレイテンシを伝送するメッセージである．チャネルを通るバルクタイプメッセージは，優先度が低くレイテンシは大きいが，効率良くメッセージを伝送することができる．また，チャネルごとに優先度を設定することができる．

4. データリンク層

論理的通信モデルにおいて，優先度を持ったメッセージの伝送を実現するための CAN 上に構築するデータリンクプロトコルについて解説する．データリンク層は，MAC（Media Access Control）副層と LLC（Logical Link Control）副層から構成される．MAC 副層はメディアアクセス制御を通して物理ユニットどうしを接続し，LLC 副層は論理ユニットどうしの接続を抽象化するものである．これより，1 つの物理ユニット内に複数の論理ユニットが存在可能となり，論理アドレスで通信が可能となる．本研究におけるデータリンク層のプロトコルの特徴は，メッセージの優先度制御を行うために，LLC 副層において，CAN の

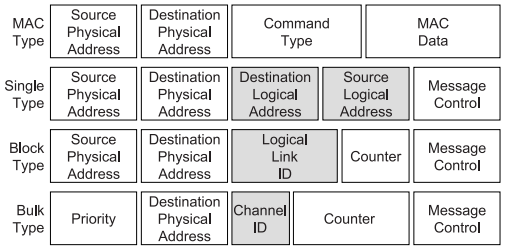

図 3 データリンク層ヘッダフォーマット概要
Fig. 3 Outline of header format for LLC sub-layer.

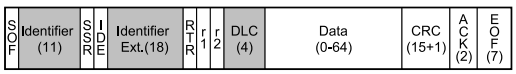

図 4 CAN 2.0B フレームフォーマット
Fig. 4 Frame format for CAN 2.0B.

Identifier（ID）フィールドの特徴を活かしてアービトレーションを行うところである．

データリンク層のヘッダフォーマット概要を**図 3** に示す．これらのヘッダに続いてペイロードが伝送される．このデータリンク層においては，(0) MAC タイプ，(1) シングルタイプ，(2) ブロックタイプ，(3) バルクタイプの 4 つのメッセージ（フレーム）タイプを定義する．MAC タイプは，ネットワーク上位の階層（LLC 副層）のメッセージを伝えるのではなく，物理ユニットの管理に利用し，シングルタイプ，ブロックタイプ，バルクタイプは，LLC 副層においてのメッセージ伝送の役割を担う．

4.1 MAC 副層

図 4 に CAN 2.0B 拡張フレームフォーマットを示す．CAN の MAC 副層は，CSMA/CR（Carrier Sense Multiple Access with Collision Resolution）を採用しており，複数の物理ユニットが同時にフレームの送信を始めた場合のアービトレーションを行う．バス上の信号はドミナントレベルとレセッシブレベルがあり，ビットとしてそれぞれ 0 と 1 で表される．物理ユニットは，バスを確認し，信号が出されていなければ，ビット単位で図 4 の MSB 側である SOF（Start Of Frame）からバス上に信号を出す．信号を出すと同時にバス上の信号のモニタも行う．今，仮に，2 つのユニットが同時に送信を開始したとする．両方のユニットが SOF の後に，ID として同時に 0 を出した場合，バス上は 0 となり，両方のユニットが 1 を出した場合，バス上は 1 となる．一方のユニットが 0 を出し，他方のユニットが 1 を出した場合，バス上は 0 となり，1 を出したユニットは 1 を出したがバス上が 0 となっていることをモニタし送信を停止する．このとき，0 を出したユニットはバス上の信号が 0 であるので，その後も信号を出し続ける．これにより，複数のユニットが同時に送信を開始した場合，ID が小さいフレームを出したユニットがアービトレーションに勝つこととなる．CAN の仕様では，ユニットごとの ID フィールドを規定しておらず，したがって，ID はフレーム（メッセージ）の送り先を示すものではなく，フレームの優先順位を示すだけのものとなる．一方，受信に際しても，各ユニットが受信すべき ID フィールドの規定はなく，ソフトウェアが登録した ID のフレームがバスに出されれば，そのフレームを受信し，そうでなければ受信しない．また，CAN では，ハードウェアにおいてエラーを検知した場合，再送を行う機能がある．

この CAN のフレームフォーマット上に，本ネットワークで定義するデータリンクプロトコルのフレームを定義する．図 4 に示した CAN のフレームフォーマットのうち，ここで利用するのは，11 ビットの Identifier，18 ビットの拡張 Identifier，データサイズを示す 4 ビットの DLC のフィールド，0 から 64 ビットの Data フィールドである．

CAN のフレームフォーマットに割り当てたデータリンクプロトコルのヘッダ構成を**表 1** に示す．29 ビットの CAN ID の上位 21 ビットをヘッダとし，下位 8 ビットおよび CAN データフィールドがデータ用となる．CAN の DLC フィールドは，CAN のフレームサイズを示すためのフィールドであるが，フレームの最大 8 バイトの場合，DLC の最上位ビットが 1 となり，下位 3 ビットを利用しないため，この 3 ビットを利用して，LLC 副層のメッセージシーケンスカウンタ（順序番号）とする．これにより，CAN の標準仕様を守りながらフィールドを有効利用することが可能となる．すべてのタイプで送信元物理アドレスを設定するが，送信されるフレームが物理ユニットごとに必ず異なることを保証するためである．

MAC タイプメッセージは，ネットワーク上位の階層のメッセージを伝えるのではなく，物理ユニットの電源制御やネットワークへの接続/離脱の管理，物理アドレスの割当て，バスのリセットなどのネットワーク管理を行うため，論理ユニットを特定する ID を含まない．このメッセージタイプを構成する送信先物理アドレス，送信元物理アドレス，コマンドタイプ，および MAC データは，CAN の ID フィールドのみに収まり，CAN フレームとしてのデータサイズを 0 バイトとすることで，効率を上げている．

表 1 CAN フレーム上のデータリンクプロトコルヘッダ詳細
Table 1 Detail of data link protocol header on CAN frame.

	CAN ID (11)				
Type	28	27	26	25 - 22	21- 18
MAC	0	0	Bcast/ P2PP	Src Phy Addr	Dest Phy Addr
Single	0	1			
Block	1	0	1		
Bulk	1	1	1	Priority	
Reserve (Stream)	0	0	1	Src Phy Addr	

	CAN ID (18)				
Type	17 - 14	13	12	11 - 8	7 - 0
MAC	Cmd Type	Reserve			MAC Data
Single	Src Logical Addr		Dest Logical Addr		Data
Block	Logical Link ID				
Bulk	Channel ID			Msg High Count	
Reserve (Stream)	1111	Stream ID		Msg High Count	

	CAN DLC (4)	
Type	3	2-0
MAC	0	0
Single	1 / 0	DLC
Block	1	Msg Count
	1 / 0	DLC
Bulk	1	Msg Low Count
	1 / 0	DLC
Reserve (Stream)	1	Msg Low Count
	1 / 0	DLC

4.2 LLC 副層

LLC 副層としてのメッセージは，(1) シングルタイプ，(2) ブロックタイプ，(3) バルクタイプの 3 タイプある．これに，将来，音声ストリームデータ伝送に利用するための予約も含めて，合計 4 種類に分類する．それぞれのメッセージはヘッダ内に物理アドレスのフィールドがあり，この部分において物理ユニットの特定を行う．また，論理ユニットの各メッセージをデータリンク層で優先度制御するため，IPv6 におけるフローラベルのような役割として，論理アドレス，論理リンク ID，チャネル ID といった論理ユニットを特定するための ID をヘッダ内に入れている．

4.2.1 メッセージタイプ

シングルタイプメッセージのヘッダにおいて，送信元論理アドレスで特定する物理ユニット内の論理ユニットから，送信先となる他の物理ユニット内の論理ユニットへの伝送であることを示し，CAN ID フィールドの下位 8 ビット（1 バイト）と，CAN データフィールドの 8 バイトを合わせて，合計で最大 9 バイトのデータを伝送する．

ブロックタイプメッセージは，論理リンクを利用して送信先の論理ユニットを指定するが，論理リンクは，送信元物理アドレス，送信先物理アドレス，送信元論理アドレス，送信先論理アドレスの組合せで設定される．論理リンクは，双方向の通信として利用可能である．DLC フィールドの 3 ビットのメッセージシーケンスカウンタにより，最大メッセージサイズは，64 バイトとなる．

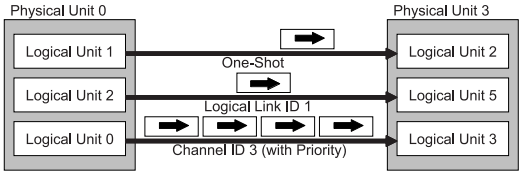

図 5 メッセージタイプの優先度（単一コネクション）
Fig. 5 Priority of message type (Single connection).

バルクタイプメッセージで利用されるチャネルも，送信元物理アドレス，送信先物理アドレス，送信元論理アドレス，送信先論理アドレスの組合せで設定され，送信元から送信先への片方向のメッセージ送信となる．CAN ID フィールド内の 4 ビット，DLC フィールドの 3 ビットのメッセージシーケンスカウンタにより，最大メッセージサイズは，1 k バイトとなる．

たとえば，シングルタイプメッセージは，ユーザの操作や機能の停止などの短いメッセージを想定し，ブロックタイプメッセージはユーザインタフェースへの文字表示などの 1 バイト以上のメッセージを，バルクメッセージは運転記録や電話帳などの優先度の低い大きなデータを想定している．

4.2.2 メッセージ優先度

表 1 に示すように，MAC タイプ，シングルタイプ，ブロックタイプ，バルクタイプの識別は，CAN ID の 28-26 ビットで区別し，それぞれ，(00X)，(01X)，(101)，(111) と割り当てている．CAN において，ビットの 0 の値が 1 の値より優先度が高いため，これらのメッセージの優先度は，MAC タイプ，シングルタイプ，ブロックタイプ，バルクタイプの順で設定される．CSMA/CR の機能により，物理ユニットを操作する MAC タイプメッセージが最優先で伝送される．また，サイズの大きいバルクタイプメッセージ伝送中であっても，シングルタイプメッセージを伝送することが可能となる．

図 5 にシングルタイプメッセージ（ワンショット），ブロックタイプメッセージ（論理リンク），バルクメッセージ（チャネル）について，それぞれの優先度を持ったメッセージの送信モデルを示す．シングルタイプメッセージは，送信元と送信先の論理アドレス間で特定され，物理ユニット 0 内の論理ユニット 1 から，物理ユニット 3 の論理ユニット 2 へ送信される．ブロックタ

イプメッセージの場合，識別子（この場合は ID1）を持った論理リンクを利用して送信先の論理ユニットを特定する．論理リンクは，送信元物理アドレス，送信先物理アドレス，送信元論理アドレス，送信先論理アドレスの組合せで設定するため，双方の論理ユニットから利用可能となり，双方向の通信を行うことができる．図中では，物理ユニット 0 内の論理ユニット 2 から，物理ユニット 3 内の論理ユニット 5 に論理リンクが設定されている．バルクタイプメッセージは，チャネル（この場合は ID3）を利用して転送される．チャネルは，送信元物理アドレス，送信先物理アドレス，送信元論理アドレス，送信先論理アドレスの組合せで設定し，送信元から送信先への片方向のメッセージ送信となる．チャネルごとに優先度を設定可能な点が論理リンクと異なる．

IDB-C のデータリンク層では，CAN が本来持つ CSMA/CR の機能の上位にトークンパッシングを構築している．これはメディアアクセス制御として冗長であり，最長遅延が 10 ms とレイテンシ増大の要因となる．また，CAN のスループットの約 1/3 をこのトークンパッシングに利用するため，低スループットとなる．さらに，IDB-C フレームは通常の CAN コントローラで処理できず，不必要なフレームであっても CPU で処理するため，CPU の負荷が高くなる要因となる．IDB-C のデータリンク層では，論理的通信モデルを実現する機能がないため，アプリケーションの設計を困難にし，また，ゲートウエイの処理を複雑にする要因ともなる．

4.2.3 アドレス，ID の割当て

組み込みシステムの場合，ネットワーク設計時にあらかじめアドレスや ID を静的に割り当てる場合も多いが，ここでは動的割当ても検討対象に含める．

物理アドレスを 4 ビットで構成するため，1 つのネットワークに接続可能な物理ユニットの数は 16 となり，IDB-C（J2366-2）において接続可能な物理ユニット数と同じである．物理アドレスの割当て方法として，J2366-2 で行っているように，ブロードキャストを多用し，すべての物理ユニットを動的に割り当てる方法も可能であるが，ネットワークのブート時にトラフィックが集中し，各ユニットが立ち上がるまでに時間がかかる．そのため，図 1 に示したネットワーク構成を前提に，ネットワーク内に唯一存在するゲートウエイに対してあらかじめ固定的に物理アドレス（たとえば 0）を割り当て，それ以外の物理ユニットは，ゲートウエイに問い合わせて割り当てる方法が簡単である．

論理リンク，チャネルの ID の割当てに関して，ブロックメッセージ，バルクメッセージ内に送信元の物理アドレスが必ず含まれているため，送信元の物理ユニットにおいて管理を行うことで，一意性を確保することができる．割当ては，送信元の物理ユニットで行い，その情報を送信先に通知することで可能となる．

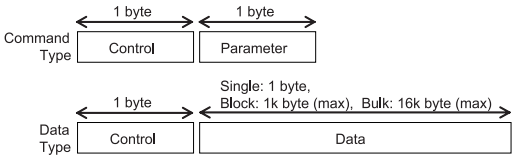

図 6 トランスポート層フレームフォーマット
Fig. 6 Frame format for transport layer.

論理アドレスを 5 ビットで示すように，物理ユニットあたり最大設置ユニット数（実行アプリケーション数）は 32 となる．論理アドレスは，アプリケーションの機能を示すファンクションの ID と，インスタンスを示す ID の組合せで構成し，特定のアプリケーションを容易に発見できる仕組みを利用することが可能である．ファンクションを示す ID はあらかじめシステム設計時に設定しておき，インスタンスを示す ID を，ゲートウエイで管理し動的に割り振ることができる．

これらのアドレス，ID 割当ての機能は，それぞれのネットワークシステムで設定されるべき要項であり，本論文では詳細について記載しない．割り当てられるアドレスは CAN の ID フィールドを利用するため，優先度に影響を与えるため，優先度の高いメッセージを送信する物理ユニットに小さな ID を割り当ててあるとネットワークシステム全体として効率が良いが，メッセージごとの優先度がより高いため，自動的にアドレスを割り当てた場合でも特に問題は発生しない．

5. トランスポート層

論理的通信モデルを実現するにあたり，データリンク層と同様に，トランスポート層も重要な役割をはたす．本研究において設計したトランスポート層においては，エラーチェックおよび再送機能を含むエンド・トゥ・エンドのデータ到達確認を行い，分割されて送信されたメッセージをまとめ，アプリケーション層に対して適切な形式でデータを伝える．

図 6 にトランスポート層のフレームフォーマットの基本構成を示す．トランスポート層のメッセージは，コントロール部の 8 ビット（1 バイト）ヘッダと，データ部の可変長ペイロードから構成される．コントロール部を 8 ビットとすることで，LLC 副層のメッセージフォーマットにおいて，CAN ID の 7-0 ビット（Data）に格納されるため，効率の良い伝送を行うことが可能

表 2 トランスポート層フレームのマッピング
Table 2 Transport frame mapping.

	Header (8)							
Frame Type	7	6	5	4	3	2	1	0
Setup	0	1	1	res.	Link Setup			
Ack for Single	0	0	0	res.	Packet ID Number			
Ack for Block/Bulk	0	1	0	res.	TPL Packet Count			
Nack for Block/Bulk	0	0	1	res.	TPL Packet Count			
Single Data	1	0	0	Ack Request	Packet ID Number (count)			
Block/Bulk Data (Start)	1	1	1	Re-sync.	TPL Packet Count			
Block/Bulk Data (Cont)	1	1	0	res.	Same as "Start"			
Block/Bulk Data (End)	1	0	1	res.	Same as "Start"			

	Data (64 bit - max 128k bit)		
Frame Type	0 - 1	2 - 7	8 - 63
Setup	Ack Mode - No Ack - Transport Block Ack - Application Frame Ack - Complete Ack only	Link ID - Logical Link ID (6bits) - Channel ID (4bits)	N. A.
Ack for Single	N. A.		N. A.
Ack for Block/Bulk	Ack Attribute - Transport Block Ack - Application Frame Ack - Complete Ack	Link ID	N. A.
Nack for Block/Bulk	Ack Attribute - Transport Block Ack - Application Frame Ack - Complete Ack	Link ID	N. A.
Single Data	Data		
Block/Bulk Data (Start)	Data		
Block/Bulk Data (Cont)	Data		
Block/Bulk Data (End)	Data		

である．トランスポート層のメッセージはコマンドとデータの 2 種類に分類される．コマンドメッセージは，LLC 副層のシングルタイプとのメッセージとして伝送する．データメッセージはシングルデータ，ブロックデータ，バルクデータの 3 種類に分類され，それぞれ，LLC 副層のシングルタイプ，ブロックタイプ，バルクタイプのメッセージとして伝送する．

トランスポート層フレームフォーマットを**表 2** に示す．コマンドメッセージは，シングルデータ伝送のための ACK，ブロックデータおよびバルクデータのための ACK，また，ブロックデータおよびバルクデータのための NACK がある．また，ブロック/バルクデータのコネクションの設定が必要な場合も，コマンドメッセージが利用され，設定を行う．

コマンドメッセージに加えて，データメッセージのシングルデータも，LLC 副層のシングルタイプメッセージとして送信される．データサイズは最大 8 バイトであり，CAN の 1 フレーム内に収まる．シングルデータの場合，送信元がアクノリッジ（ACK）を必

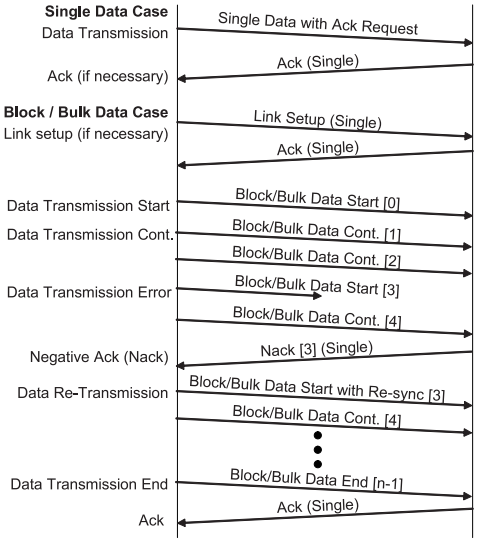

図 7 トランスポート層メッセージシーケンス例
Fig. 7 Example of transport layer message sequence.

要とする場合，Ack Request フラグをセットすることで，メッセージを受け取った側がコマンドメッセージとして ACK を返す．

ブロックデータおよびバルクデータは，それぞれ，LLC 副層のブロックタイプメッセージ，バルクタイプメッセージで送信される．データメッセージのヘッダ部には，4 ビットのメッセージシーケンスカウンタがあり，分割されたメッセージのシーケンス番号を入れる．ブロックデータの PDU のサイズは最大 1 k（64×16）バイトであり，バルクデータは最大 16 k（$1{,}024 \times 16$）バイトとなる．論理リンク，チャネルが確立していない場合は，ID を管理している送信元ユニットにおいて利用する ID を割り当て，Setup コマンドを利用して送信元ユニットから送信先ユニットに対して通知を行う．このリンク確立時にそれぞれのメッセージに対して Ack の必要/不要も通知する．複数パケットを送信する場合，先頭パケットは Start，最終パケットは End，途中のパケットは Cont の設定となる．Ack を必要とした場合，複数パケット送信中に，特定のパケットがエラーとなった際に，送信先ユニットで受信するはずのメッセージシーケンスカウンタの値を入れて送信元ユニットに Nack として返す．送信元ユニットは，この Nack パケットのメッセージシーケンスカウンタのパケットから再送信を開始する．その際，先頭パケットに Re-sync フラグをセットすることで，途中からの送信であることを意味する．メッセージシーケンス概要を**図 7** に例示する．

IDB-C のトランスポート層では，同時に複数の論理

ユニットの通信をサポートできないため，2 つの物理ユニット間のそれぞれの論理ユニットがデータ伝送中に，同一物理ユニットに搭載される複数の論理ユニットが同時にメッセージを送信し，応答を受信することが困難となる．たとえば，車載ユニットとしてのナビゲーションから，ネットワークで接続された携帯電話へ，電話帳データを転送している際，ユーザが車載ユニットを通して携帯電話を操作できないという問題がある．

6. その他の階層

本研究では，アプリケーション層におけるメッセージのフレームフォーマットについて，特に規定していない．ここでは，アプリケーション層のメッセージサイズは，トランスポート層の最大値と設定して，ブロックタイプメッセージの場合 1 k バイト，バルクタイプメッセージは 16 k バイトとしている．これ以上のサイズのデータの場合は，アプリケーションにおいて分割して送信する必要がある．物理層に関しては，CAN の仕様から変更する点はないが，プラグ・アンド・プレイを実現するためには，機器接続時に，IDB-C で規定している ID 割当てのための追加仕様が必要となる．その場合，バスのインピーダンスの関係から，最長伝送速度は 250 kbps となる．

7. 実装と評価

7.1 プロトコル実装

本研究で検討した論理的通信モデルを実現するための優先度制御を行うデータ転送プロトコルの実装を行った．下位ネットワークは CAN 2.0B とし，マイクロプロセッサとして CAN コントローラを内蔵している三菱電機（ルネサス・テクノロジ）製の M16C（16 MHz）を利用した．評価においては，CAN の速度は，IDB-C 物理層仕様と比較するために，250 kbps と設定したが，処理能力的には 1 Mbps でも伝送可能である．実装したシステムは小さなモニタ上で動作し，より下位のプロトコルスタックが優先的に起動するように設定しているが，実際のバス上のデータ伝送による実時間性が支配的となるため，バスの動作と同期してスタックを動かすような構造にはなっていない．

図 8 に本プロトコル実装の概要を示す．プロセッサ内の CAN コントローラを制御するデバイスドライバの上に，データリンク層，トランスポート層，アプリケーション層があり，その上位にアプリケーションを搭載している．1 つのユニットにシングルタイプメッセージ送信アプリケーション，バルクタイプメッセージ送信アプリケーションを搭載し，他のユニットに，シングルタイプメッセージ受信アプリケーション，バルクタイプメッセージ受信アプリケーションを搭載した．この 2 つのユニットを CAN ネットワーク経由で接続し，データの送受信テストを行った．メッセージの送受信が正常に行われるかはもちろんであるが，ここでは，メッセージの優先度が認識され，優先度の低いメッセージを送信中に，優先度の高いメッセージが遅れなく到達するかを確認することに主眼を置いた．性能を評価するために，内部に 10 μs のソフトウェアタイマを設定し，これにより時間の計測を行う．

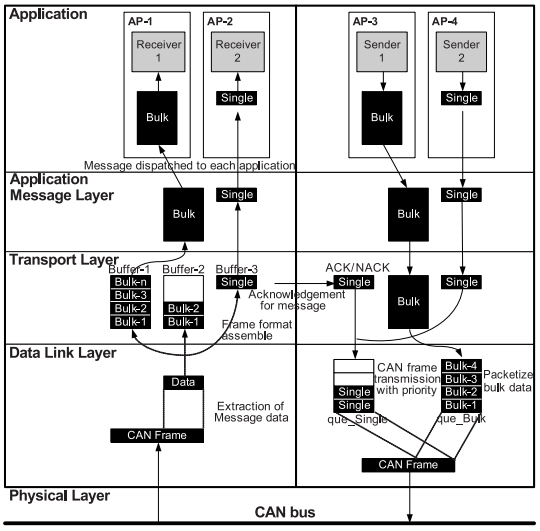

図 8 プロトコル実装概要
Fig. 8 Software architecture for protocol implementation.

7.2 性能評価

実装したネットワークプロトコルについての優先度に基づいたメッセージ伝送の評価を行うため，シングルタイプメッセージ伝送，バルクタイプメッセージ伝送それぞれの個別送信，およびバルクタイプメッセージの送信中にシングルタイプメッセージ伝送する複合送信の時間を計測した．これにより，IDB-C で問題となっている優先度の低いデータ伝送中に，優先度の高いデータ伝送の実現について，確認することができる．評価条件として，シングルタイプメッセージは最大の 8 バイトとし，バルクタイプメッセージは，送信中にシングルタイプを送るために十分なサイズの 1 k バイトと設定した．計測する送信時間は，アプリケーションがメッセージ送信のための関数を呼んだ時点から，メッセージ送信完了の割込み関数が完了するまでを計測している．システムとして組み込んだ 10 μs のソフトウェアタイマで計測するとともに，Vector 社の

表 3 メッセージ送信時間
Table 3 Message transmission time.

	Single Type	Bulk Type
Individual Transmission	1.59 ms	106.1 ms
Complex Transmission	1.71 ms	107.4 ms

CAN プロトコルアナライザである CANoe [15] を利用して，伝送されるフレームの計測も実施した．

表 3 にそれぞれのメッセージの送信時間の測定結果を示す．シングルタイプメッセージの個別の送信時間の実測値は約 1.59 ms で，その中で，アプリケーションが送信のための関数を呼び出してからデータリンク層がデータを送出するまでの時間が約 1 ms，1 つの CAN フレームがバスに完全に送出されるまでが約 670 μs かかっていることが計測された．バルクタイプメッセージの個別の送信時間は約 106.1 ms で，その中で，アプリケーション層からデータリンク層までメッセージが送られる時間はおよそ 20 ms である．バルクタイプメッセージ送信中にシングルタイプメッセージを送信する複合送信では，シングルタイプメッセージの送信時間は 1.71 ms であり，バルクタイプメッセージの送信時間は 107.4 ms となった．

1 つの CAN のフレームをネットワーク上で伝送するために必要な時間は 51 μs（128 bits/250 kbps）であり，次の CAN フレームは前の CAN フレームの伝送が完了した後に送信される．バルクタイプメッセージ送信中にシングルタイプメッセージを送信するためには，最長でこれだけの遅延が発生する場合がある．実際，シングルタイプメッセージを送出するまでの遅延時間を計測すると約 120 μs であり，このことから，ここでのネットワークプロトコルは，優先度に基づいたメッセージ伝送において正常に機能していることが分かる．バルクメッセージの送信時間は，個別送信より複合送信のほうが約 1.3 ms 長い．これはシングルタイプメッセージの送信フレームと ACK フレームの伝送に必要なバス占有時間が約 1.2 ms であり，それに追加して，ACK フレーム受信処理時間がおよそ 300 μs 必要となり，この計測された遅延時間はほぼ妥当な値と考えられる．また，CANoe を利用して，メッセージ送受信の経緯を測定すると，バルクタイプメッセージ送信中にシングルタイプメッセージが送信されており，本ネットワークプロトコルで設計した優先度に基づいたメッセージ伝送が正常に動作していることが確認できた．

スループットについては，本トランスポート層のプロトコルのヘッダ部を，CAN の ID フィールドにおいて伝送しているため，CAN のデータ伝送としてのスループットと同等となる．

IDB-C の場合，バルクタイプ形式のメッセージ送出完了まで，シングルタイプ形式のメッセージが送出できない．バルクメッセージ送信中におけるシングルタイプメッセージの送出タイミングにより，送信時間が異なるが，最長送信時間は 106.1 ms となる．また，伝送効率に関しては，CAN のスループットの約 2/3 となる．これは，擬似的なトークンパッシングを利用しているため，データ伝送を行うユニットがなくても，最長 1.5 ms ×(ユニット数 − 1) の時間を待つ必要があり，また，最長トークン保持時間を超えると，データ送信を中止しなければならないためである．

IP over CAN の場合，一般に IP ヘッダサイズは 20 から 60 バイトであり，仮に 24 バイトとすると，IP ヘッダを伝送するだけで，CAN の 3 フレームが必要となる．したがって，8 バイトのデータを伝送するのに 4 フレームが必要となる．また，IP over CAN の仕様によると，通信開始前に 2 フレームの情報交換フレームが流れるので，最悪の場合，68 バイトのデータを送るのに 11 フレーム必要となり，データの伝送効率は，4.5%にまで低下する．トランスポート層におけるセグメントのサイズが大きければ，相対的に効率は良くなるが，データよりもコマンドの伝送が多いシステムにおいては，さらに効率が悪くなる．

8. ま と め

本研究では，論理的通信モデルの検討を行い，CAN 上において優先度設定可能で最長伝送時間を保証できるようなデータ伝送のためのデータリンク層，トランスポート層のプロトコルを開発した．CAN（2.0 B）メッセージは，29 ビットの ID と最大 8 バイトのデータフレームから構成されており，上位に通信モデルを構築するには制限が大きい．そのため，本論文では，各階層のプロトコルのフレームフォーマットを，効率的に CAN フレームにマッピングする方法についても具体的に記述した．そして，このプロトコルを，CAN コントローラ内蔵の M16C マイコンに実装し，評価を行った．その結果，IDB-C において実現できていない論理的通信モデルの構築に加え，応答時間，データ転送効率，CPU 負荷についての問題について改善されていることを立証することができた．

参 考 文 献

1) Nossal, R.: Meeting the Challenges in a Collaborative OEM-Supplier Development of Distributed Embedded Systems, *Distributed Em-*

bedded Systems Engineering SP-1885 SAE International, pp.21–27 (2004).
2) International Organization for Standardization: ISO 11898-1, Road Vehicles—Controller Area Network (CAN)—Part 1: Data Link Layer and Physical Signalling (2003).
3) 後藤正博，秋山 進：自動車用ネットワークの技術動向，デンソーテクニカルレビュー，Vol.6, No.1 pp.82–89 (2001).
4) Staszel, M., Emaus, T. and Erichson, B.: MP3 on CAN—CD Quality with Wiriting Reduction, SAE Technical Paper, 2003-01-1202 (2003).
5) Meier, R., Kaiser, J., Hughes, B., Brudna, C. and Cahill, V: An Event Model for Real-Time Systems in Mobile Environments, *Proc. 2nd Workshop on Software Technologies for Fugure Embedded and Ubiquitous Systems*, pp.29–34 (2004).
6) Fredriksson, L.: A CAN Kingdom Rev. 3.01, Kvaser (1996).
7) Zuberi, K.M. and Shin, K.G.: Real-time Decentralized Control with CAN, *Proc. IEEE Conference on Emerging Technology and Factory Automation*, pp.93–99 (1996).
8) Neilsen, M.L.: A Flexible Real-time Transport Protocol for Controller Area Networks, *Proc. International Conference on Parallel and Distributed Processing Techniques and Applications*, pp.25–28 (2001).
9) Cache, P., and Fiedler, P.: IP over CAN, Internet-Draft of the Internet Engineering Task Force, draft-cafi-can-ip-00.txt (2001).
10) Society of Automotive Engineering: J2355 ITS Data Bus Architecture Reference Model Information Report (1997).
11) Society of Automotive Engineering: J2366-1 ITS Data Bus—IDB-C Physical Layer (2001).
12) Society of Automotive Engineering: J2366-2 ITS Data Bus—IDB-C Data Link Layer (2001).
13) Society of Automotive Engineering: J2366-4 ITS Data Bus—IDB-C Thin Transport Layer (2002).
14) Society of Automotive Engineering: J2366-7 ITS Data Bus—IDB-C Application Message Layer (2002).
15) Vector Informatik Web Page. http://www.vector-informatik.de/english/products/buses.php

(平成 16 年 12 月 6 日受付)
(平成 17 年 6 月 9 日採録)

佐藤 健哉（正会員）

1984 年大阪大学工学部電子工学科卒業．1986 年同大学院工学研究科電子工学専攻修士課程修了．同年住友電気工業（株）情報電子研究所入社．1991〜1994 年スタンフォード大学計算機科学科客員研究員．2000 年奈良先端科学技術大学院大学情報科学研究科博士後期課程修了．2001 年米国 Automotive Multimedia Interface Collaboration, Inc. Chief Technologist．2004 年同志社大学工学部情報システムデザイン学科助教授．博士（工学）．組み込みシステム，通信プロトコル，ITS に関する研究に従事．ACM，IEEE-CS，SAE 各会員．

小板 隆浩（正会員）

1993 年同志社大学工学部電気工学科卒業．1995 年同大学院工学研究科電気工学専攻博士前期課程修了．1999 年奈良先端科学技術大学院大学情報科学研究科博士後期課程単位認定退学．1999 年大阪産業大学工学部助手．2001 年同学部講師．2004 年同志社大学工学部情報システムデザイン学科専任講師．博士（工学）．分散処理，Grid コンピューティング，スケジューリング，ゲノムアプリケーションに関する研究に従事．電子情報通信学会，IEEE-CS 各会員．

井上 博之（正会員）

1987 年大阪大学工学部電子工学科卒業．1989 年同大学院工学研究科電子工学専攻修士課程修了．同年住友電気工業（株）システムエレクトロニクス研究開発センター入社．1998 年奈良先端科学技術大学院大学情報科学研究科博士後期課程修了．2000 年（株）インターネット総合研究所入社．2003 年（株）同社ユビキタス研究所主幹研究員．2004 年（株）IRI ユビテックユビキタス研究所第二研究部長．博士（工学）．分散処理，組み込みシステム，インターネットプロトコルの研究に従事．電子情報通信学会，IEEE-CS 各会員．