

分散制御用リアルタイム通信規格 *Responsive Link*

山崎 信行（慶應義塾大学） 松井 俊浩（産業技術総合研究所）

概要 *Responsive Link* は分散制御に必要なリアルタイム通信を実現した通信規格であり、ISO/IEC で国際標準化されている。*Responsive Link* はリアルタイム通信を実現するために、通信のプリエンプション（横取り）をパケット追いつきによって実現している。さらには柔軟なリアルタイム通信を実現するために、ソフトリアルタイム通信（データリンク）とハードリアルタイム通信（イベントリンク）の分離、異なる優先度を持つパケット毎に別経路（専用回線や迂回路等）を実現、ノード毎に優先度を付け替えることで分散管理型でパケットの加減速を制御、ハードウェアによる 1 ホップ毎のエラー訂正等の様々な機能を実現する。*Responsive Link* は並列分散制御用の SoC (System-on-Chip) である *Responsive Processor* や *Responsive Multithreaded Processor (RMTP)* 等実装されており、これらを用いた実機応用、検証・試験、及び標準化作業等についても述べる。

1. はじめに

近年、組込みシステム、特に各種ロボットやメカトロニクス等の分野において、分散制御や分散リアルタイム処理が行われている。そのためには、プロセッサやコントローラ間のリアルタイム通信が必要であるが、分散制御用途の実用的なリアルタイム通信規格はほとんど存在していなかった。そこで、我々は分散制御を実現するための実用的なリアルタイム通信規格である *Responsive Link* を提案・設計・実装し、さらには国内外で標準化を行った。*Responsive Link* はリアルタイム通信の標準規格となるべく情報処理学会試行標準 IPSJ-TS0006:2003[1]として国内の標準化を、ISO/IEC JTC1 SC25 において ISO/IEC24740:2008 として国際標準化を行っている。本稿では、まず実用的なリアルタイム通信規格を設計するに至った背景や設計思想に焦点を当て、*Responsive Link* の設計・実装、*Responsive Link* を用いた SoC/SiP、実機応用、検証・試験、及び標準化について順に述べる。

2. 分散制御システム

2.1 開発の背景及びターゲットアプリケーション

Responsive Link のターゲットアプリケーションは、組込みシステムであり、特にその中でも分散制御システム（各種ロボットやメカトロニクスシステム等）をメインターゲットとしている。分散制御システムをトップダウンに設計・実装するためには、信頼性が高く時間粒度の細かいノード（分散コントローラ）間リアルタイム通信が必要である。

Responsive Link の研究開発は 1997 年から行っているが、

当時、我々は分散制御型ロボットをトップダウンに設計を行うために既存の各種通信規格（イーサネット、ATM、CAN、USB、IEEE1394、I²C 等）の調査を詳細に行った。その結果、2.4 節で述べるように分散制御用途に耐えるリアルタイム通信は存在しておらず、分散制御型ロボットを容易に開発することは困難であることが分かった。そこで我々は分散制御用途に耐えうるリアルタイム通信規格 *Responsive Link* の新規設計を開始した。設計に際し、時間粒度が細かく高精度・高速な分散制御システムを実現可能とし、ターンアラウンドの短いシステム設計が実現できることを目標とした。また、設計当初から国際標準を意識して設計を行い、当時の通商産業省の国際標準創成研究開発制度を用いて最初の研究開発を開始した。*Responsive Link* は国際標準創成研究開発制度から ISO/IEC 標準になった数少ない例の一つでもある。

これらの分野では国内メーカーが既に強い国際競争力を持っているが、さらなる付加価値及び優位性を確保するのに *Responsive Link* が役立つように設計を行った。

2.2 リアルタイム性

制御にはリアルタイム性が必要となる。リアルタイム性とは、処理や通信等の正しさが時間にも依存するという性質である。狭義には、与えられた時間制約（デッドライン）を守るということの意味する。リアルタイム性は、ハードリアルタイムとソフトリアルタイムに大別できる。

ハードリアルタイム性とは、必ず時間制約を守らなければならない性質であり、時間制約を少しでも破ると価値が 0 になる性質である。狭義には、時間制約を破った場合、システムに損害を与える可能性のあるリアルタイム性のことである。主に制御系の処理や通信が要求する

リアルタイム性である。

ソフトリアルタイム性とは、時間制約を多少破ることを許容する性質であり、時間制約を破っても価値はただちに0にはならない。多くの場合、時間制約を破ると、時間経過と共に価値が減少していく。また、時間制約を破っても、システム自身に損害を与えることはない。多くのマルチメディア系（動画・音声処理等）の処理や通信はソフトリアルタイム性を必要とする。

近年の多くのシステムにおいては、上記のハードリアルタイム性とソフトリアルタイム性の両方が必要とされている。

2.3 リアルタイムスケジューリング

通常、複雑なリアルタイム処理は、リアルタイムスケジューラによってスケジューリングされた順番にタスクを実行することによって実現される。リアルタイムスケジューラは、時間制約（デッドラインや周期等）を元に優先度を計算し、その優先度に従ってリアルタイムオペレーティングシステムがプリエンプション（横取り）を行いながらタスクの実行を行う。図1に EDF (Earliest Deadline First)[2]によるスケジューリング例を示す。EDFでは、デッドライン（ μ 切時間）に近いタスク順に高い優先度をタスクに与え、高い優先度のタスクから順に実行を行う。ここで、プリエンプションは、演算の場合はコンテキストスイッチによって実現することができるが、通信の場合には従来は実現が困難であった。

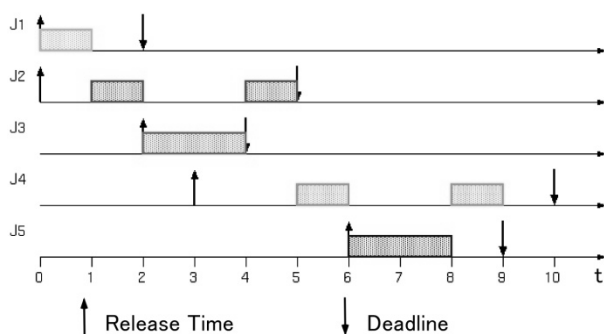


図1. EDF スケジューリング

Responsive Link はリアルタイムスケジューリング理論に基づいてリアルタイム通信を実現するために、通信のプリエンプションを実現する。具体的には、ノード毎に優先度によるパケットの追い越し機構を実現することによって、リアルタイム通信を実現している。基本的には、リアルタイムスケジューラでスケジューリングされた（優先度付けされた）通信をハードウェアでプリエンプティブに行うことによって、リアルタイム通信を実現する。

2.4 従来のリアルタイム通信

IEEE1394[3]や USB[4]等の従来のリアルタイム通信（アイソクロナス転送等）では、通信データのバンド幅の集中管理を行なっているルートノードにあらかじめ通信のバンド幅を予約しておき、ルートノードがバンド幅を一元管理することでソフトリアルタイム通信を実現している。このような集中管理方式は、PC にペリフェラルを接続して制御する用途には便利であるが、大規模なリアルタイム通信システムへの応用は困難である。元々、そのような用途での使用は想定されていないので、USBでは最大127ノード、IEEE1394では最大63ノードしかサポートしておらず、また、ルートノードが故障した場合は全ての通信が停止してしまう。また、リアルタイム通信モード（アイソクロナス転送）ではハードウェアによるエラー訂正を行わないので、分散制御用途でアイソクロナス転送を行うのは危険である。

また、自動車内などで用いられている CAN (Controller Area Network) は優先度をサポートしておりソフトリアルタイム通信が可能であるが、シリアルバスを採用しているため、通信の実効速度が接続モジュール数に依存する。さらに、ノード数が少ない場合でも最大通信速度がおおよそ1Mbps（実際には500kbps程度）と遅く、高速・高精度な分散制御用途に使用するのは困難である。

2.5 集中制御と分散制御

従来、ロボット等の制御方式として主に行われているのは集中制御という方式であり、一つのコンピュータ（パーソナルコンピュータや組込みマイコン等）で全てのI/O（センサやアクチュエータ等）を集中的に制御する方式である。典型的には、パーソナルコンピュータに必要な制御ボード（A/Dコンバータボード、D/Aコンバータボード、PWM (Pulse Width Modulation) 発生器ボード等）を挿し、それら制御ボードを用いてロボット全体の制御を行う。この方式では構成が簡単になるという利点があるが、配線量が膨大になるという問題や、冗長性がないため耐故障性が低い等の問題点がある。さらに、パーソナルコンピュータを使用する集中制御の場合、サイズが大きく消費電力もかなり大きくなってしまいう問題点がある。

これらの問題点を解決するには、I/O制御機能、ネットワーク機能（リアルタイム通信機能）、制御（演算）機能が付いた分散コントローラを用いてロボットの分散制御を行えばよい。分散制御は、集中制御に比較してアルゴリズムや制御プログラムが複雑になるというデメリ

ットもあるが、メリットとしては上記の問題を解決し、小型化、省配線化、低消費電力化、耐故障性等を実現することができる。

図2はヒューマノイドロボット Kojiro における分散制御システムの例を示している。一般にロボットの分散制御では各関節等に組込まれた分散コントローラが自身に接続されている I/O (センサやアクチュエータ等) の制御をリアルタイムに行い、かつ分散コントローラ間で必要なデータをリアルタイムに通信することによって、システム全体の制御を行っている。

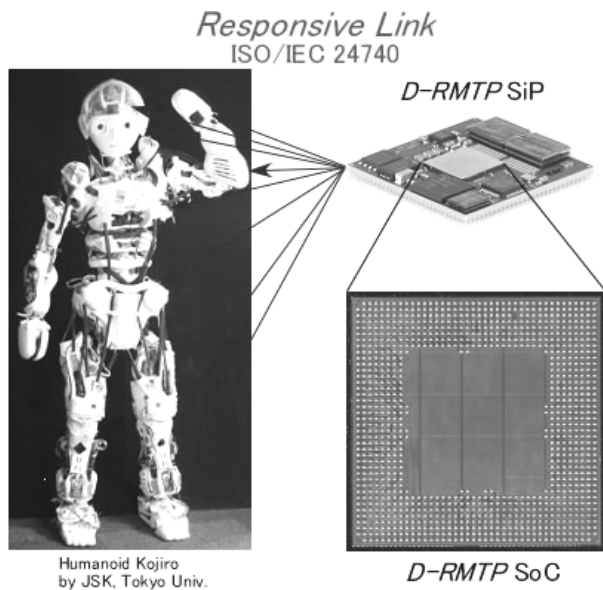


図2. 分散制御の例 (ヒューマノイドロボット Kojiro)

ここで、分散制御を行う場合には以下の式を満たす必要がある。

$$T_{\text{sens}} + T_{\text{com}} + T_{\text{calc}} + T_{\text{act}} < T_{\text{cycle}}$$

T_{sens} : センサデータの取得時間

T_{com} : 通信時間

T_{calc} : 次の指令値の計算および発行時間

T_{act} : アクチュエータ制御に必要な時間

T_{cycle} : 制御周期

実際に使用するセンサ-アクチュエータ間の全組み合わせで上式を満たせば分散制御可能であり、シミュレーション等によって判定できる。ここで T_{sens} と T_{act} は容易に求めることができ、 T_{cal} はリアルタイム処理を行うことで上限を制約できる。 T_{com} は通常の通信では上限を課することが困難であり、リアルタイム通信が要求される。

2.6 リアルタイム通信への要求仕様

以上述べてきた通り、従来の通信規格では以下の要求を同時に満たしていないため、そのまま分散制御用途に使用するには困難が伴う。従って、Responsive Link では

以下の要求仕様を同時に実現することを目指して設計を行った。

- リアルタイムスケジューリング理論に基づいたリアルタイム通信の実現: 通信におけるプリエンプションの実現
- ハードリアルタイム通信とソフトリアルタイム通信の両立
- 分散制御のための時間粒度の細かいリアルタイム通信: 1msec より小 (10~100 μ sec オーダ)
- マルチメディアデータ等を通信しても実用的な速度: 100Mbps 以上
- リアルタイム通信時のエラー訂正能力

3. Responsive Link の機能

分散制御用途のリアルタイム通信への要求仕様を実現するために、Responsive Link は下記の方法を組み合わせることによって、大規模かつ時間粒度の細かいリアルタイム通信を実現する。

- 通信パケットに優先度を付け、高い優先度の通信パケットが低い優先度の通信パケットを通信ノード毎に追い越し (通信におけるプリエンプションの実現)
- ハードリアルタイム通信とソフトリアルタイム通信の経路の分離
- 通信パケットの優先度を通信ノード毎に付け替えることによってパケットの通信速度の加減速を分散管理で制御
- 全く同じネットワークアドレス (送信元アドレス及び送信先アドレス) を持つ通信パケットの経路を優先度によって別の経路に設定することによって専用回線や迂回路を設けリアルタイム通信を制御
- エラーによる再送を可能な限りしないようにするために、ホップ毎にフレーム単位でハードウェアによる前方エラー訂正

3.1 通信のプリエンプション (優先度による追い越し機構)

通信においてプリエンプションを実現するために Responsive Link では優先度を用いたパケットの追い越し機構を実現している。具体的には、パケット追い越し用バッファと退避用外部記憶を有したルータスイッチを設計・実装している。

図3は5入力5出力で、一つの入力部当たり追い越し用バッファが4パケット分あるネットワークスイッチの構成を示している。0番はプロセッサコア(CPU)に接続さ

れ、1~4 番は外部に入出力されている。規格的には外部ポートは1ポート以上あれば通信できるが、ネットワークポロジの汎用性と、VLSIに集積するゲート数とのトレードオフから、これまで設計された *Responsive Link* のルータスイッチはすべて5入力5出力(外部へは4入力4出力)の構成を採っている。

入力ポート(In0~4)から入力された通信パケットは、通信ノードで衝突しない場合、そのまま出力ポート(Out0~4)へ出力を行う。異なる入力ポートから入力された通信パケットが同じ出力ポートに出力を行なう場合、通信パケットに付加された優先度に従い、優先度の低い通信パケットを追い越し用バッファ(意味的には追い越され用バッファ)に貯めて出力を待たせ、高い優先度の通信パケットを先に出力させる。高い優先度の通信パケットの出力の後に低い優先度の通信パケットを追い越し用バッファから出力ポートに出力し、優先度に従った通信パケットの追い越しを行う。

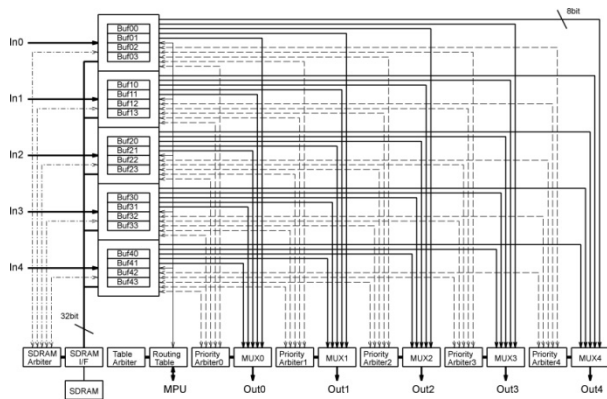


図3. 追越し機能付きルータスイッチ

上記の通信パケットの追い越しを実現するために通信パケットの大きさと等しい追い越し用バッファを複数本入力ポート側に設けている。図3の例では4パケット分の追い越し用バッファを設けているが、これまで開発されたVLSIには全て8パケット分の追い越し用バッファを実装している。ここで、追い越し用バッファのサイズは大きいほど追い越し用バッファが溢れる可能性が低くなり(つまり通信のストールや再送が起きにくくなり)さらには性能も高くなるが、サイズを大きくするとVLSIのゲート数(面積)が増えてしまうというトレードオフが存在する。リアルタイム性を維持するためには再送をさせないようにすることが最優先であり、そのためにシミュレーションを繰り返し、8パケット分の追い越し用バッファがあればほとんどの場合において追い越し用バッファが溢れないということが分かった。そのため、VLSIのゲート数(面積)は増加してしまうが、リアルタ

イム性を重視して8パケット分の追い越し用バッファを実装している。

さらに、出力が待たされ続けている時に入力が入り続けバッファが溢れそうになった場合に、追い越し用バッファの内容を一時的に退避するための退避用外部記憶(外付けメモリ)をオプションで設けることができるようにしている[5]。上述の通り8パケット分の追い越し用バッファがあればほとんどの場合において追い越し用バッファが溢れることはないが、通信の状況においては発生する可能性が0ではない。退避用外部記憶(外付けDRAM)を実装することにより、事実上、無限サイズの追い越し用バッファを想定でき、リアルタイム通信を阻害する再送を回避することができる。

3.2 ハードリアルタイム通信とソフトリアルタイム通信の分離

一般に、ソフトリアルタイム通信のデータ(画像データ、音声データ等、以下、単に「データ」と呼ぶ)のサイズは大きく、それに対してハードリアルタイム通信のデータ(制御コマンド、同期信号等、以下、単に「イベント」と呼ぶ)のサイズは非常に小さい。従って、従来型の1系統の通信路で全ての通信を行う方法では、同時に通信すべき通信データとして、大量のデータパケットと、量的にはわずかではあるが分散制御用途には非常に重要なイベントパケットが同一種類のパケットとして存在する。データとイベントを、共有された同一の通信線を通して時分割に通信を行う従来方式ではイベントの伝達時間の上限を正確に制約できないので、時間粒度の細かいハードリアルタイムシステムは実現困難であると考えられる。

また、複数のモジュールでひとつの通信チャネルを共有するシリアルバスでは、同時に何台のモジュールが通信するかによってバンド幅が動的に変化し通信時間の上限を制約することが困難であり、実効速度も出にくい。

さらに、ソフトリアルタイム通信は主にバルク的なマルチメディアデータの通信等に用いられ、ハードリアルタイム通信は主に制御等に用いられるので、

- ソフトリアルタイム: バンド幅保証でよいのでスループットをできるだけ上げたい
- ハードリアルタイム: レイテンシを保証する必要があるレイテンシをできるだけ小さくしたい

というアプリケーション側からの要求がある。明らかにこれらの要求はトレードオフの関係にあり、パケットサイズを大きくするとスループットは高くなるが、同時にレイテンシも長くなる。逆にパケットサイズを小さくす

用途のリアルタイム通信には4レベルの優先度では足りないことが判明し、最終的には8bit (256レベル)の優先度に変更を行った。

3.4 通信速度とディペンダビリティ

Responsive Link は分散制御用途であるので、エラー訂正を行なう必要がある。その際、できるだけエラー訂正によってリアルタイム性が損なわれないようにする必要がある。ここで、パケット単位でCRCを付加しエラー訂正を行う方法では、パケット全体を受信しないとエラー訂正できない。その場合、ホップ毎にレイテンシが積算されていくので、リアルタイム通信のエラー訂正としては好ましくない。そこで、*Responsive Link* では1ホップごとにフレーム(図5の下部参照)単位でエラー訂正を行い、1フレーム(8bitデータ+4bit冗長符号)につき1bitのエラーであれば、再送することなしにハードウェアで誤り訂正を行うようにする。

Responsive Link の通信速度は、様々な環境(コンフィギュレーション、アプリケーション)を想定し、800, 400, 200, 100, 50, 12.5, 6.25 [Mbaud]の範囲で段階的に可変となっている(表1)。

表1. 通信速度とケーブルの関係

速度(Mbaud)	100	200	400	800
最大長(m)	100	80	60	40
推奨ケーブル	Cat5	Cat5e	Cat6	Cat6

4. *Responsive Link* を集積した VLSI システム

4.1 *Responsive Processor*

Responsive Processor は *Responsive Link* を最初に集積・実装した VLSI である[6]。 *Responsive Processor* に実装された *Responsive Link* は規格化を行っている最中に VLSI 化されたテストベッドである。 *Responsive Processor* を用いて様々な実機評価を行い、規格制定にフィードバックを行うことができた。

例えば、 *Responsive Processor* 上の *Responsive Link* の片方向の最大変調速度は100[Mbaud]であり、優先度は4レベル(2bit)、I/O インタフェースは PECL となっていた。各種の実機評価の結果、速度も優先度も不足していることが判明し、最終的には片方向の最大変調速度は800[Mbaud]、優先度は256レベル(8bit)、I/O インタフェースは LVDS に変更を行った。

4.2 *Responsive Multithreaded Processor (RMTP)*

Responsive Multithreaded Processor (RMTP) は分散リアルタイム制御を容易に実現することを目指した SoC (System-on-Chip) であり、フル規格の *Responsive Link* を集積している[7]。現在までに *RMTP*、 μ *RMTP*、*M-RMTP*、*D-RMTP* の4種類が開発されている。

4.3 *D-RMTP* SoC

D-RMTP は *RMTP* シリーズの最新バージョンであり、分散制御に必要なほとんどの機能を1チップに集積した SoC である。 *Responsive Link* 以外にも、リアルタイム処理用プロセッシングコア(*RMT PU*)、メモリ(内蔵 SRAM、DDR SDRAM I/F)、コンピュータ用周辺機能(DMAC、PCI-X、IEEE1394、Ethernet、RS-232C等)、制御用周辺機能(PWM発生器、PWM入力、エンコーダ、デジタルI/O、SPI等)を1チップに集積している(図5)。

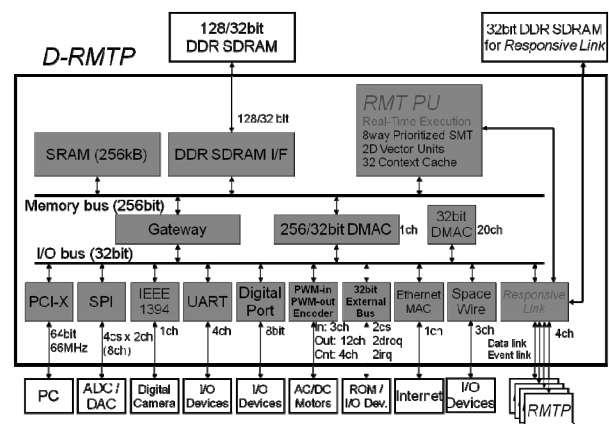


図5. *D-RMTP* のブロック図

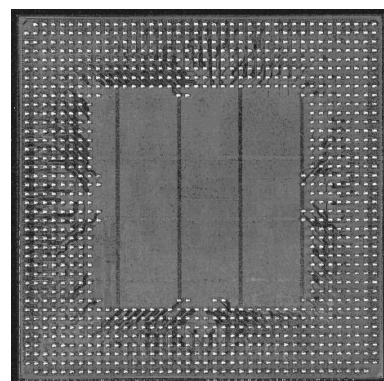


図6. *D-RMTP* チップ写真 (10mm 角)

図6に *D-RMTP* のチップ写真を示す。 *D-RMTP* のI/Oピンはバンパによるフリップチップ実装されており、1000ピン以上の多くのピンを効率よく入出力すると共に、SiP (System-in-Package) への設計・実装を行いやす

ようにしている。

4.4 D-RMTP SiP

D-RMTP はシリコン上にシステムを構築（集積）した SoC であり必要な I/O は全てチップ上に集積しているが、DRAM と Flash memory はプロセス及びゲート数的に集積することができない。そこで、Responsive Link を用いた分散制御を容易に実現するために、D-RMTP, DRAM, Flash memory 等を用いて超小型の SiP (System-in-Package)を開発した。

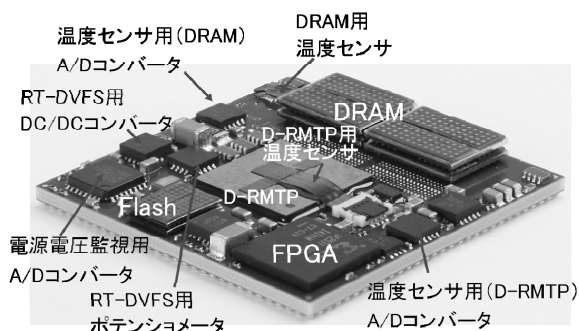


図 7. 30mm 角 D-RMTP SiP

図 7 に 30mm 角 D-RMTP SiP を示すが、4 個の DRAM 及び 2 個の Flash memory を FFCSP (Flexible carrier Folded real Chip Size Package) という技術を用いて上下にスタックするように 3 次元実装することにより、超小型の SiP を実現している[8]。30mm 角 D-RMTP SiP には、上記のコアパーツ以外にも各種センサ（DRAM 用温度センサ、D-RMTP 用温度センサ、電源電圧監視用センサ等）を SiP 上に集積し、リアルタイム動的電圧周波数制御 (RT-DVFS: Real-Time Dynamic Voltage and Frequency Scaling) を SiP 上で実現可能にする。この機能により、分散制御（リアルタイム通信・処理）をできるだけ低消費電力で行うことを可能としている。

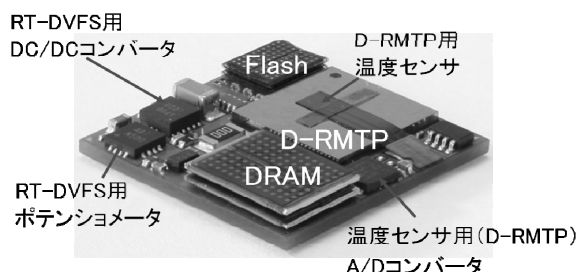


図 8. 20mm 角 D-RMTP SiP

図 8 に最新の 20mm 角 D-RMTP SiP を示す。30mm 角 D-RMTP SiP と同様に 2 個の DRAM 及び 2 個の Flash memory を FFCSP を用いて 3 次元実装し、さらに RT-DVFS に必要な機能も 20mm 角に集積している。

20mm 角 D-RMTP SiP は図 2 に示すヒューマノイドロボット Kojiro を分散制御するために研究開発を行ったものであり、Kojiro の体内に 60 個の 20mm 角 D-RMTP SiP を組み込み Responsive Link を用いて相互接続して分散制御を行う。

5. 検証及び試験

Responsive Link を互換性を持って安全に相互接続するためには、相互接続性や耐ノイズ性が重要となってくる。

5.1 相互接続検証

Responsive Link の相互接続検証には、現在、図 9 に示す D-RMTP 評価ボードをリファレンスボードとして使用している。この評価ボードには D-RMTP SiP 専用ソケットを実装しており、D-RMTP SiP を基板に半田付けすることなくソケットのまま動作させることが可能であり、検証に使用することができる。

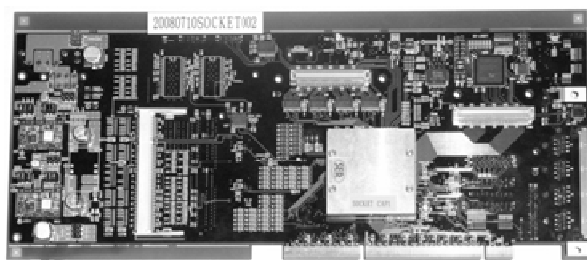


図 9. 30mm 角 D-RMTP SiP 評価ボード

この評価基板のソケット（図 11 中央やや右）に D-RMTP SiP を入れ、Responsive Link をループバック接続し、長時間の送受信耐久テストを行ってエラーが 0 のものを選択し、さらにその中からオシロスコープで波形を確認して電氣的に規格に準拠し最も安定しているものをリファレンスとして用いている。

現在、Responsive Link を搭載したチップのソケット付評価ボード（約 10 種類）は全て用意されているので、検証を行いたい場合には、その評価ボードのソケットにチップがマウントされたパッケージを入れ、そのボードとリファレンスボードを Responsive Link で接続することによって行う。検証ソフトウェアで検証する通信速度や時間等を設定すれば自動的に検証を行うことができる。図 10 に実際の検証の様子を示す。リファレンスボードと、検証を行いたいチップ用の検証用ボードを Responsive Link で接続し、通信の検証を行う。図 10 の検証のコンフィギュレーションでは、後述するノイズ発生機に Responsive Link ケーブルを通し、耐ノイズ試験も同時に行えるようになっている。新規チップの Responsive Link

の相互接続検証試験を行いたい場合には、まず評価検証用ボードを作成し *Responsive Link* を用いて同様に接続して相互接続検証試験を行う。



図 10. 相互接続検証試験の様子

5.2 耐ノイズ試験

Responsive Link は分散制御を主目的にしているため、周囲にノイズ発生源（各種アクチュエータやセンサ）がある可能性が高い。例えば、東京大学 JSK が *D-RMTP SiP* を用いて研究開発している分散制御用モータドライバ [9] は非常に大出力（80V, 50A 連続, 200A ピーク）であり、通信が不能になるようなノイズを周囲に発生させる可能性がある。そのため、実際の動作環境に近い状況で耐ノイズ性に関する試験を行う必要がある。

ロボット制御に使用する実際のモータドライバから発生するノイズを増幅してケーブルに対して与え、そのノイズ強度を変化させて耐ノイズ試験を行う。具体的には図 11 に示すようなモータドライバ（東京大学 JSK 作成）とノイズ発生機（コイル）を用いて、ロボット制御時とまったく同じノイズを発生させ試験を行う。

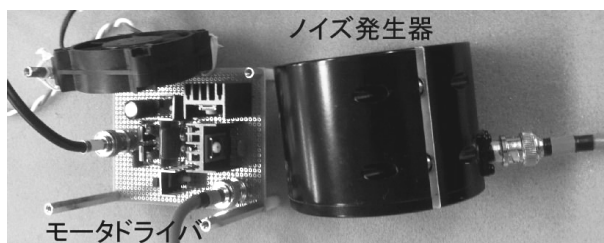


図 11. ノイズ発生機

図 12 にノイズ発生器を通した信号波形を示す。元の送信波形（スパイクの乗った矩形波）と、ノイズ発生器を通りノイズが載った受信波形（振幅の小さい矩形波）とを同時にオシロスコープを使用してキャプチャしている。このノイズ発生機のコンフィギュレーションでは、通信の

クロック周波数とほぼ同じ周波数のノイズを発生させているが、受信波形を見るとそのように緑色の受信波形にノイズが載っている様子が分かる。



図 12. ノイズ波形

ノイズの強度を変化させ、測定用ソフトウェアで BER (Bit Error Rate) を測定することにより耐ノイズ試験を行うことができる。

6. 標準化

通信規格は標準化を伴わないと容易に相互接続することができず、普及も困難となる。*Responsive Link* は、ISO/IEC での標準化を目指し、1999 年に国内の JTC1 SC25 WG4 内にレスポンスリンク標準化 SG を立ち上げ標準化を開始した。国際標準化提案はすぐには進展せず、その時期にちょうど情報処理学会試行標準制度が立ち上がり、国際標準化と並行して国内標準化を行うことになった。まずは学会試行標準として国内標準化を行い、それをベースに ISO/IEC で国際標準化を行うのが良いとの判断があった。

6.1 学会試行標準 IPSJ-TS 0006:2003

Responsive Link の国内における標準化を学会試行標準で行った。2002 年に学会試行標準 WG6 を発足し、NP (New Work Item Proposal) の提案を行った [10]。WG6 で議論を重ね、2003 年に IPSJ-TS 0006:2003 として標準化を行った [1]。

6.2 ISO/IEC 24740:2008

IPSJ-TS 0006:2003 をベースにして、2003 年に ISO/IEC JTC1 SC25 WG4 に *Responsive Link* の NP を提出し、その年に承認され国際標準化を開始した。その後、CD (Committee Draft) を提出するために、まずは N ナンバ (非公式ドキュメント) の CD のドラフトを各国に配布して

意見を聞いた。その際、特に物理インタフェースに関して意見があった。SC25 にはケーブルやコネクタに関するメーカーから多くの参加者があり、物理インタフェースには特に興味があるという背景がある。ケーブルとコネクタに関しては、当初は細く小型のものを新規提案することを考えていたが、各国の意見を考慮し規格の通りやすさを考え、最終的には既存の IEC 61156-X (通称イーサネットケーブル) と IEC 60603-X (通称 RJ-45) とを用いることにした。これらの変更を行いつつ、2005 年の CD 承認、2006 年の FCD (Final Committee Draft) 承認を経て、2008 年に ISO/IEC 24740:2008 として国際標準化された。

7. おわりに

本論文では分散制御用リアルタイム通信規格である *Responsive Link* の設計思想、通信アーキテクチャ、応用例、標準化等について述べた。*Responsive Link* はリアルタイムスケジューリング理論を用いたリアルタイム通信を実現するために、通信のプリエンブション (パケットの追越し) やハードリアルタイム通信とソフトリアルタイム通信の分離等、様々な特徴を有している。

Responsive Link は本論文で紹介したロボットの分散制御用途以外にも、多くの応用分野が考えられる。例えば自動車内 LAN やハイエンドなユビキタスシステムのバックボーンとして使用可能である。今後、日本発のリアルタイム通信規格として、幅広く使用されることを期待している。

現在、さらなるディペンダビリティの向上のために、*Responsive Link* の改版を行うための評価や再設計等の準備を進めている。*Responsive Link* を用いて分散制御システムを構築する際に、モータの出力が非常に大きい場合にはそれに伴う大きなノイズが発生し、現行のエラー訂正では追いつかない場合もあることが分かり、特にノイズ耐性に関する改版を行おうとしている。そして近い将来に *Responsive Link II* (仮称) の標準化も行いたいと考えている。

謝辞 本研究は科学技術振興機構 CREST の支援により行われている。本研究の一部は GCOE の支援による。

参考文献

- 1) 情報処理学会試行標準: IPSJ-TS 0006:2003, <http://www.itsecj.ipsj.or.jp/ipsj-ts/02-06/toc.htm>.
- 2) C.L. Liu and James W. Layland: Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment, Journal of Association for Computing Machinery, Vol.20, No.1, pp.46-61 (1973).
- 3) IEEE1394, <http://www.1394ta.org/>.

- 4) USB, <http://www.usb.org/>.
- 5) 山崎信行: 分散管理型通信方法及び装置, 特許第 3460080, 特願平 11-343139 (2003).
- 6) 山崎信行, 松井俊浩: 並列分散リアルタイム制御用レスポンス・プロセッサ, 日本ロボット学会誌, Vol.19, No.3, pp.68-77 (2001).
- 7) Nobuyuki Yamasaki: Responsive Multithreaded Processor for Distributed Real-Time Systems, Journal of Robotics and Mechatronics, Vol.17, No.2, pp.130-141 (2005).
- 8) 和田喜久男, 大須良二, 樋野滋一, 山崎信行: 三次元実装パッケージ実現のためのハイブリッド FPC の提案, 電子情報通信学会論文誌, Vol.93, No.11, pp.455-463 (2010).
- 9) 浦田順一, 伊東信之, 中西雄飛, 岡田慧, 稲葉雅幸, 山崎信行: ヒューマノイドロボットのための実時間プロセッサ: D-RMTP 搭載大出力モータドライバモジュールの設計, 第 28 回日本ロボット学会学術講演会, pp.1A3-3 (2010).
- 10) 情報処理学会試行標準 IPSJ-TS 0006:2003 NP, <http://www.itsecj.ipsj.or.jp/ipsj-ts/ts0006np.html>.

山崎 信行 (正会員)

E-mail: yamasaki@ny.ics.keio.ac.jp

1991 年慶應義塾大学理工学部物理学科卒業, 1996 年同大学院理工学研究科計算機科学専攻博士課程修了。博士 (工学)。同年電子技術総合研究所入所。1998 年 10 月慶應義塾大学理工学部情報工学科助手。専任講師を経て 2004 年 4 月同助教授。1997-2000 年科学技術振興事業団さきがけ研究 21 研究員を兼務。2007-2008 年カーネギーメロン大学 Research Scientist を兼務。リアルタイムシステム, 計算機アーキテクチャ, VLSI 設計等の研究に従事。電子情報通信学会, 日本ロボット学会, IEEE 各会員。

松井 俊浩 (正会員)

E-mail: t.matsui@aist.go.jp

1980 年東京大学計数工学科卒業。1982 年東京大学大学院情報工学専門課程修士修了, 工技院電子技術総合研究所入所。1991 年工学博士。マルチメディアディスプレイ, オブジェクト指向 Lisp, 幾何モデラー, 移動ロボットを研究。1991 年から 1999 年に米国スタンフォード大学, MIT, オーストラリア国立大学の客員研究員。2003 年より, 産総研デジタルヒューマン研究センター副センター長。産総研情報通信エレクトロニクス分野研究コーディネータ。日本ロボット学会, 計測自動制御学会, ISRR 国際会議から論文賞等, 工業技術院長賞などを受賞。日本ロボット学会, 情報処理学会, ACM, IEEE の会員。

投稿受付: 2011 年 6 月 26 日

採録決定: 2011 年 7 月 26 日

編集担当: 谷口倫一郎 (九州大学)