「情報社会を支える新しい高性能情報処理技術」
平成 15 年度採択研究代表者

松井　俊浩

（独立行政法人産業技術総合研究所デジタルヒューマン研究センター　副センター長）

「ヒューマノイドのための実時間分散情報処理」

1．研究実施の概要

本プロジェクトの目的は、ヒューマノイドロボットの性能および信頼性を向上させるために、従来の少数のプロセッサによる集中制御に代わる、30 以上のプロセッサによる分散制御を実現することである。それによって、処理能力のスケーラブルな拡張が可能になり、並列処理活用することで、低消費電力で従来以上の処理性能が得られる。プロセッサをセンサやアクチュエータの近くに配置することで、信号線の延長を大幅に削減し、高い信頼性が得られる。一方、従来モノリシックなプロセスで制御可能であったものを分散化させるため、通信およびマルチプロセス(スレッド)を実時間化し、優先度や資源割り当て等を行う基盤ソフトウェアが必要になる。

従って、本研究は次の三つから構成される。

①RMTP の設計・開発：実時間性にすぐれたマルチスレッド向きアーキテクチャ、低クロックでも性能を発揮するための並列性、実時間通信チャネルを備えたプロセッサの作成
②実時間基盤ソフトウェア：Linux-2.6 を拡張して優先度に基づく実時間スケジューリングを可能にし、プログラムの実行時間を静的に予測する技術の研究
③ロボット実証：認識、対話、歩行、経路計画などの機能をモジュラーに構成して分散化し、並列処理を活用することによる性能向上

2005 年度までに、μRMTP のチップ試作、パッケージ化、PCI 評価ボード実装、PC による RMTP ソフトウェアシミュレータが完了し、RMTP への Linux カーネル移植、X86 アーキテクチャでの Linux の実時間化に着手した。2006 年度(今年度)は、μRMTP の SiP による小型化とモータ制御モジュールの製作、PCI 評価ボードでの Linux の検証、小型モータモジュールへの RMTP の組込と実時間 Linux による制御、ベンチマーク性能評価、認識・計画処理向き HP-RMTP の設計などを行う計画であった。

PCI 評価ボード上の μRMTP は、動作はしたものの、投機実行、MMU 制御などにいくつかのバグがあり、Linux を完全動作させるまでには至らなかった。製作前にテスト、シミュレーションを行っていたものの、Linux カーネルのような大規模かつ複雑なソフトウェアで発生する条件をすべ

て網羅できていなかったと考えられる。そのため、次の三つの方針をとることとした。

①RMTP の再試作に向けて、投機実行、マルチスレッド、例外処理、MMU など問題の発生が予想され、まだテストできていない部分を中心に論理検証を入念にやりなおす。
②ソフトウェア開発をストップさせないため、ソフトウェアシミュレータを用いて Linux 開発、CORBA 開発を継続する。また、電力制御アルゴリズムなど、PC を用いた研究を実施する。
③コンパイラ、アセンブラに修正を加えてバグを回避し、実機 RMTP によるテスト、評価、移植が勧められるようにする（普及しているプロセッサでもバグを含むことはよくあり、ソフトウェアによる回避は常套手段）

論理検証の結果、新たなバグは発見されず、次の再試作を行うことができた。シミュレータを用いて、Linux の実時間性、電力制御の有効性などを実証することができた。実機による評価では、十分な低電力性、ベクトル化による高速化の効果が計画どおりであることが検証できた。

２．研究実施内容

2.1 RMT-LSI グループ

分散リアルタイム処理用プロセッサである μRMT Processor（μRMTP)の設計、検証、システムインパッケージ(SiP)化および SiP 評価ボードへの実装を行った。μRMTP には、以下に述べるリアルタイム計算、リアルタイム通信、コンピュータ制御、ロボット向き制御 I/O 各機能を System-on-a-chip として実装している。

・リアルタイム処理機能：RMT PU
・リアルタイム通信機能：Responsive Link
・コンピュータ用周辺機能：PLL(1GHz，クロック制御機構)、PCI-X，IEEE1394(AV-Link)，RS-232C(2ch)，32bit DMAC (12ch)，256bit DMAC (1ch)，DDR SDRAM I/Fs (128/32bit)，ROM I/F，汎用バス I/F
・制御用周辺機能：PWM generators (32bit，100MHz，9ch)，PWM input (32bit，2ch)，Pulse counters (32bit，A，B，Z 相，9ch)

μRMTP は、バックエンド設計までを慶應義塾大学で行い、ファブリケーションは、TSMC 0.13μm LV-FSG(CMOS 8 層銅配線)プロセスで行った。ゲート数は約 14M ゲート、ダイサイズは 9,980 um x 9,980 um = 99.6mm^2、Core 電圧 1.0V、I/O 電圧 3.3V (5V tolerant)、ピン数 1004 ピンである。消費電力は、ソフトウェアとハードウェアの電力制御機構により 0.1～4W と可変である。以下に、レイアウト図とチップ写真を示す。

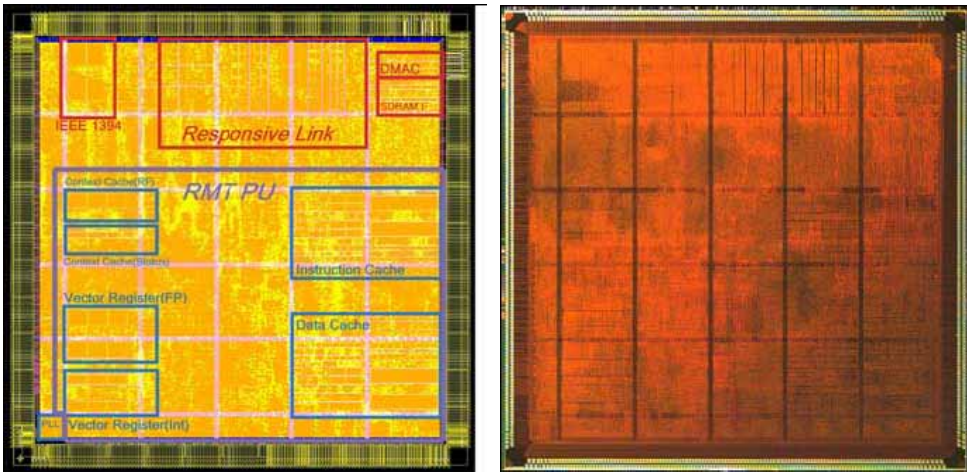

図 1　μRMTP のレイアウト及びダイ写真

μRMT Processor ベアダイおよび DDR SDRAM ベアダイを搭載したシステムインパッケージ(SiP)である μRMT SiP 並びに μRMT SiP の PCI 評価ボードを設計・実装した。μRMTP SiP は上下 4 層の最先端のビルドアップ基板(最細線幅 20μm)で実現している。PCI 評価ボードは、PCI フルサイズの 12 層基板で作成し、DC-DC 電源(2.5V, 1.0V)、ブート ROM、Responsive Link コネクタ(RJ45 x 2)、64bit 3.3V PCI エッジ、RS-232C、信号を計測するための Mictor コネクタ等を配置している。この PCI 評価基板を用いて、μRMTP 及び SiP の評価及び検証を行った。

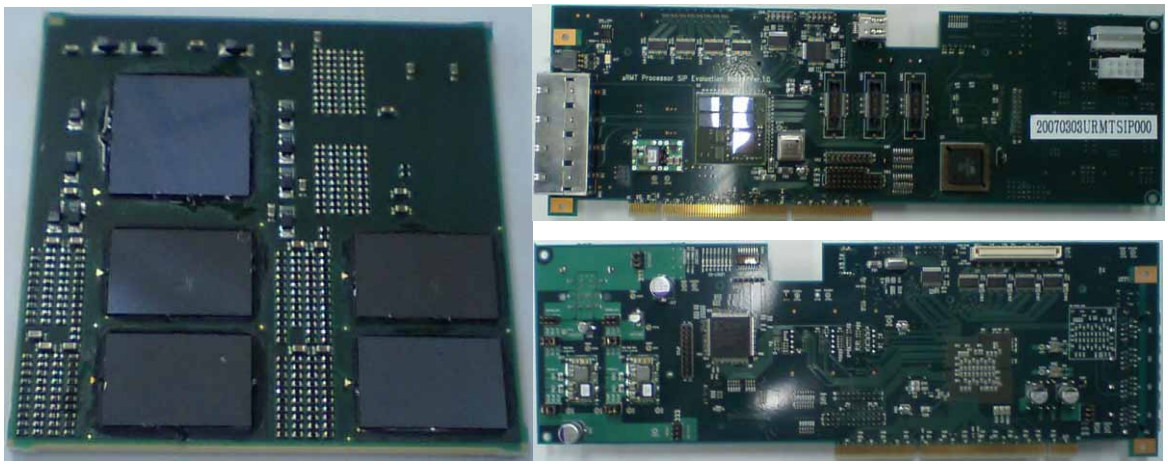

図 2 μRMTP SiP 及び μRMTP SiP PCI 評価ボード

μRMTP 用のクロス開発環境の設計及び実装を行った。μRMTP の ISA は MIPS の上位互換であるが、GNU の GCC と binutils をベースにし、μRMTP のネイティブコード(マルチスレッド命令、ベクトル命令等)を出力できるように設計を行っている。μRMTP Ver.1 には、投機実行、MMU、ベ

クタユニットなどにいくつかのハードウェアバグが見つかっているので、ソフトウェア(コンパイラ)でそれらのバグが顕在化しないようにする回避策を行っている。それらは、コンパイラオプションでON/OFF できるようにしている。

より高い実時間予測性と性能を得るために、プロセッシングユニットである Responsive Multithreaded Processing Unit (RMT PU)を二つ搭載したリアルタイム処理用 Chip Multiprocessor (CMP)である High Performance Responsive Multithreaded Processor (HP-RMTP)を設計した。通常の L2 キャッシュシステムでは L2 アクセス要求の優先度を考慮していないため、多数の L2 アクセスが競合する場合には L2 キャッシュが時間予測性を損なう要因となる。そのため、L2 キャッシュの競合が起きる箇所に優先度による調停機構を導入した。8 スレッドで並列演算を行うプログラムを CMP の 2 つのプロセッシングユニットに分散した結果、一つの RMT PU で実行する場合より実行時間を約 64%短縮し、スループットを約 72%向上させることができた。また、高優先度スレッドの実行時間が低優先度スレッドの影響で 8.9%延びていた状況で、L2 キャッシュの優先度制御を有効にすることにより、実行時間の増加を 1.85%に抑えることができた。以上により、時間予測性の向上と性能向上を達成した。HP-RMTP は、TSMC 90nm プロセスで設計を行い、300MHz で動作させた場合、μRMTP の 5 倍以上の性能を達成できることを確認した。

2.2 実時間分散系基盤ソフトウェア

2.2.1 WCET 予測ツール RETAS

安全で信頼性の高い実時間システムを構築するには、実時間タスクの実行がデッドラインを満たすことを保証する必要がある。一般には統計的に実行時間の最悪値を求めるため、デッドラインを満たすことは保証されない。プログラムの実行時間を予測するために、GCC(GNU Compiler Collection)を基にしたツールである RETAS を開発している。RETAS はソースコードレベルでタスクの全ての実行フローを解析し、最悪の実行時間パスを求めて、実行時間の最悪値を保証する。さらに、RETAS は、様々なアーキテクチャへ容易に移植できることを目指している。アーキテクチャの違いは GCC の中間表現を利用して吸収する。

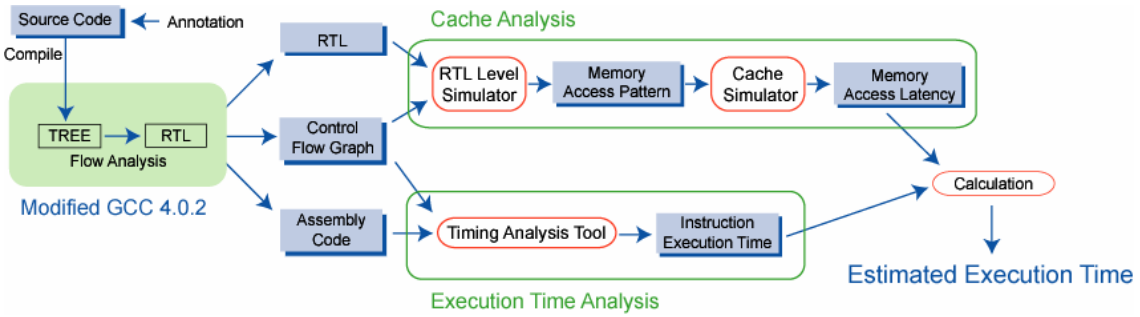

図 3 実行時間予測システム RETAS

現在の予測システムの構成を図 3 に示す。予測対象のソースコードから、RTL(Register Transfer Language)、Control Flow Graph(CFG)、アセンブリコードを生成する。RTL は GCC 内部で使用されるアーキテクチャに非依存な中間表現である。GCC 内部にフロー解析器を実装し、ソースコード中の全ての実行フローを表現した CFG を生成する。

プログラムの実行時間は、メモリアクセス命令の実行時間とメモリアクセス以外の命令実行時間に分けて計算する。メモリアクセス命令の実行時間は、キャッシュの振る舞いに依存する。RTL と CFG からメモリアクセスパターンを求める RTL レベルシミュレータ、メモリアクセスパターンからメモリアクセス命令の実行時間を求めるキャッシュシミュレータを開発した。メモリアクセス以外の命令実行時間は、CFG とアセンブリコードから基本ブロック毎の実行時間を計測するコードを生成し、ターゲットアーキテクチャで実行、計測する。メモリアクセスの時間および基本ブロックの実行時間からプログラム全体の実行時間を予測する。

今まで、評価環境として、Pentium-M や XScale といった実際のアーキテクチャを用いてきた。これらアーキテクチャは内部動作に関する詳細なデータシートが非公開なために、ツールの予測精度を厳密には検証できなかった。そこで、アーキテクチャの内部動作がソースコードレベルで提供されている SimpleScalar シミュレータを用いることにした。SimpleScalar シミュレータはクロックレベルでスーパースカラー、out-of-order 実行、キャッシュの動作をシミュレーション可能である。

表 1 に、評価に使用した SimpleScalar のパラメータおよびベンチマークプログラムの予測値と SimpleScalar シミュレータによる測定値の結果を示す。

表 1 SimpleScalar シミュレータでの実行時間予測結果

Processor Model	SimpleScalar 3.0/PISA
Data Cache Size	L1 4KB / L2 64KB
Cache Access Latency	1cycle / 6 cycles
Way	4Way, Set Associative
Replacement Policy	LRU
Line Size	32 bytes / 64 bytes
Memory Access Latency	18 cycles

ベンチマーク	測定値	予測値	割合(%)
行列乗算	30044	32459	108
挿入ソート	10139	10514	103
フィボナッチ	2135	4012	192

どのベンチマークプログラムも測定値よりも予測値が大きく、安全に最悪実行時間を予測している。フィボナッチで測定値と予測値に大きな乖離が生じているのは、タイミング解析の精度が小さな基本ブロックに対しては悪くなるためである。タイミング解析では基本ブロック毎に実行時間を測定するが、基本ブロック間のパイプラインオーバーラップを考慮していないため小さな基本ブロックで測定誤差が広がる。

今後、パイプラインのオーバーラップを考慮したタイミング解析を行い精度の向上を目指す。具体的には、複数の基本ブロックを連結し実行時間を測定することを検討している。また、命令キャッシュのレイテンシや分岐予測ミスのペナルティの予測をシステムに組み込む予定である。また、本システムをフリーソフトウェアとして配布すべく準備を進めている。

2.2.2　実時間オペレーティングシステム Shi-Linux

ロボットなどのハードリアルタイムシステムでは、タスクの実行はデッドライン時間内に終わらなければならない。通常のカーネルでは、デバイスからの割り込みが発生すると、デッドライン時間に近づいているタスクがあるにもかかわらず、タスクの実行が中断され、割り込み処理が行われる。このため、実時間タスクのデッドラインをミスする可能性がある。本研究で実装している Shi-Linux では、これまでの一般的な実時間 Linux カーネルと違い、デバイスからの割り込みを禁止し、デバイスドライバを一つの実時間周期タスクとして実現される。これにより、デバイス割り込みによるランダムな実行がなくなり、タスクの実行時間の予測性が増す。実時間タスクスケジューリングのために、タイマー割り込みのみを許す。Shi-Linux は、x86 系 CPU で実装を進め、現在、第2版となっている。現在、以下の機能が実現されている。

(1)デバイスファイルを通じた実時間タスクの制御。

(2)EDF アルゴリズムに基づいたスケジューラの実装。

(3)x86 系の PIC(Programmable Interrupt Controller)では、スプリアス割り込みが発生するため、Local APIC(Advanced PIC)を利用したタイマー割り込みを使用。

(4)x86 系の BIOS では、SMI(System Management Interrupt)を用いて USB レガシーが実現されている。これにより、最大 140us の間、CPU が横取りされる。SMI を生じさせないように変更することにより、コンテキストスイッチ時間を、最大 140us から最大 2.5us までに減少させ、かつ、コンテキストスイッチ時間が安定。

2.2.3 プロセス単位スケジューラ

Intel 社 Pentium M プロセッサ等で実現されている DVFS 機能 (Dynamic Voltage and Frequency Scaling)は、ソフトウェアから CPU の周波数と電圧を動的に変更する機能である。CPU 負荷に応じて、周波数と電圧を制御することにより省電力化が可能となる。従来実現されているソフトウェア技術では、デーモンプロセスにより大域的な CPU 負荷統計情報を用いて DVFS 機能により周波数と電圧を制御していた。

我々は、実行時に動的にプロセス単位で電力を制御するスケジューラのプロトタイプを Linux 上に実装した。Linux において、プロセスの優先度は以下の式で動的に決定される。

$$\text{動的優先度} = \text{静的優先度(Nice 値)} - (\text{BONUS} + 5)$$

静的優先度は、プロセス生成時に決められる値である。BONUS は、Linux のスケジューラによって決められる値で、0～10 の値を有する。応答性能を有するプロセスに対して優先度を上げるような工夫をしている。本プロトタイプでは、プロセスのコンテキストスイッチを行う時に、BONUS の値を用いて電力制御を行う。すなわち、0 から 10 の値に対して、周波数 600MHz から 1600 MHz の値にマッピングする。プロセスコンテキストスイッチ時に、この値で CPU 周波数を設定する。以下のタスクセットおよび計測環境(図 4)を用いて本プロトタイプシステムを評価した。

計算機の種類	タスクの種類	タスク
評価対象機	応答性能を必要とするタスク	WEB サーバ
	バッチタスク	prelink (Linux で定期的に稼動しているデーモン)
クライアント機	WEB クライアント	Apache Bench

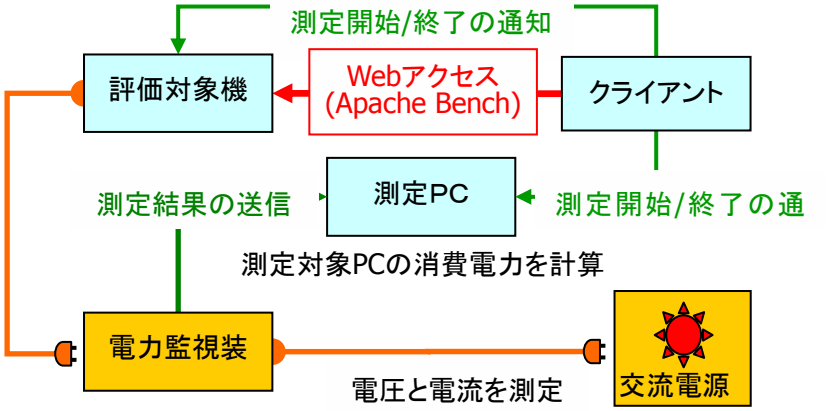

図 4　計測環境

本ベンチマークでは、WEB サーバの平均応答性能を維持し、かつ、消費電力量がどのくらい減るかが評価軸となる。実現した方式に対する比較として、以下に示すとおり、既存 Linux で実現されている電力制御アルゴリズムおよび手動で WEB サーバには最大電力、prelink には最小電力を割り当てた時の性能を計測した。なお、既存 Linux の手法は、プロセス単位での制御ではない。

Performance	周波数最大に固定
powersave	周波数最小に固定
ondemand	定期的に CPU 負荷を調べて設定
conservative	ondemand よりも長い周期で定期的に設定
Ideal	WEB サーバに最大周波数、prelink に最小周波数を設定

結果を図 5、図 6、図 7 に示す。図 5 および図 6 は、それぞれ、各電力制御機構の WEB 応答性能、各電力制御機構の消費電力量を示している。応答性能では、ideal についで良い結果となっていて、消費電力量では、powersave ならびに ideal についで良い結果となっていることが分かる。

powersave は、計測中、最低電力で常に動いているので、最も低い消費電力量となる。図 7 に、消費電力量あたりの応答性能の値をグラフに示す。理想値に近い性能が出ていることが分かる。

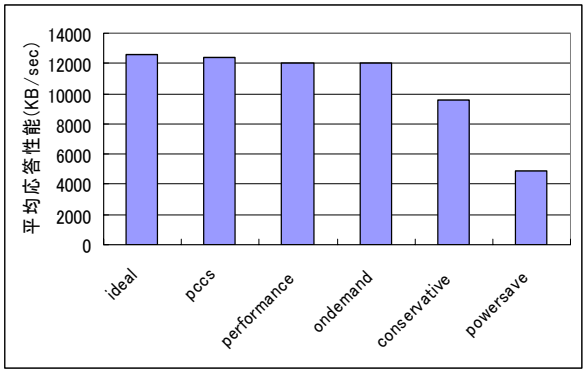

図 5 各電力制御機構の WEB 応答性能

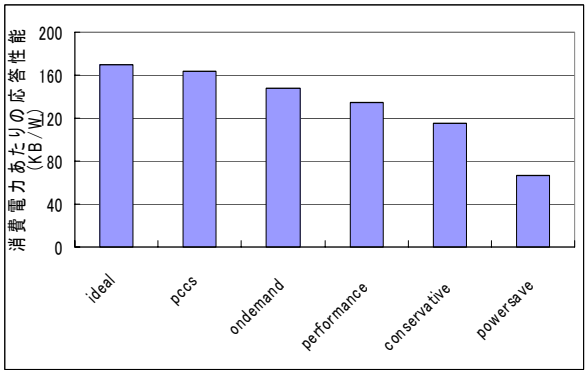

図 6 各電力制御機構の消費電力量

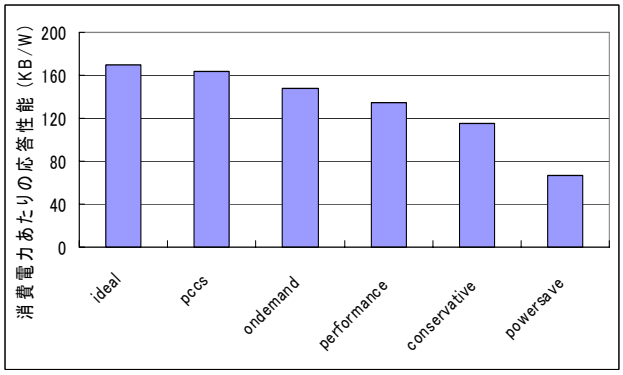

図 7 消費電力量あたりの応答性能値

2.3 ロボット実証グループ

2.3.1 モータ制御モジュール

μRMTP をキーコンポーネントとし、各種の DC/AC モータを制御できるモジュールの開発を行った。従来品に比べて体積比で 1/17 を達成した。2006 年現在、μRMTP の SiP パッケージは実現しておらず、小型モジュールへの搭載はできていない。別途の DSP を載せたボードと組み合わせて開発を行い、モジュールの評価試験を行った。その CPU ボードを μRMTP-SiP に交換することで、最終的なモータモジュールとする予定である。現在、評価ボードに小型モータ制御モジュールのアンプを接続し、DC および AC モータを回転させることに成功した。

項　目	仕　様	備　考
基板寸法	40×45 [mm]	外出しｺﾈｸﾀ&ｹｰﾌﾞﾙ含まず
制御電源	DC　5 [V]	
モータ駆動電源	DC 48 [V]	
対象とするモータ	DCモータ & ACモータ	ﾌｧｰﾑｳｪｱによる切替
モータ駆動方式	PWM	ﾊﾟﾜｰMOSFET:6個搭載
連続定格電流	10 [A]	
対象とするｴﾝｺｰﾀﾞ	ｲﾝｸﾘﾒﾝﾀﾙｴﾝｺｰﾀﾞ	

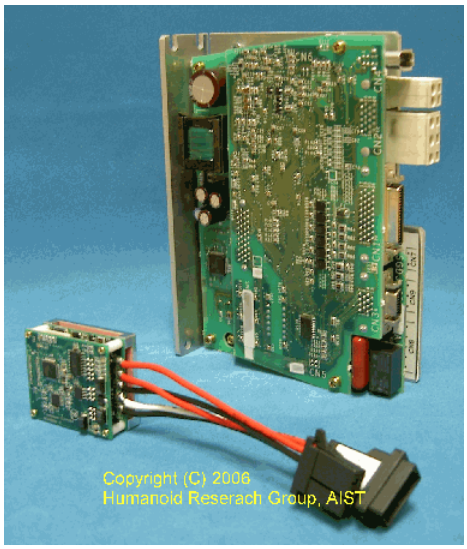

図 8 小型モータ制御モジュール
（後方は従来の市販品）

3. 研究実施体制

(1)「RMT-LSI」グループ

①研究者名

山﨑　信行（慶應義塾大学大学院　助教授）

②研究項目

・実時間通信・処理・制御用システムオンチップの研究開発

(2)「基盤ソフトウェア」グループ

①研究者名

石川　裕（東京大学大学院　教授）

②研究項目

・実時間オペレーティングシステムと開発環境の開発

(3)ロボット実証グループ

①研究者名

比留川　博久（産業技術総合研究所　副部門長）

②研究項目

・ヒューマノイドロボットの分散制御系の研究

4. 研究成果の発表等

(1)論文発表(原著論文)

○加藤真平, 小林秀典, 山崎信行, “SMT プロセッサにおける実行効率を向上するリアルタイムスケジューリング”, 情報処理学会論文誌コンピューティングシステム, Vol. 47, No. SIG 12, pp. 133-146, September, 2006.

○宮川, 石川, “低消費電力のためのスケジューリングアルゴリズム”, 情報処理学会 研究報告 2006-OS-102, pp. 17 - 22, 2006.

○宮川, 石川, “電力制御スケジューラのプロトタイプ実装”, 情報処理学会 研究報告 2006-OS-103, pp. 109-115, 2006.

○戴, 石川, “厳密な周期タスク実行を支援する実時間 Linux カーネルの実装”, 情報処理学会 研究報告 2006-OS-103, pp. 41 - 46, 2006.

○山本, 石川, 松井, “移植性の高い実行時間予測手法の設計と実装”, 情報処理学会 研究報告 2006-ARC-169, pp. 127-132, 2006.

○船岡健司, 加藤真平, 山崎信行, “Pfair スケジューリングにおけるコンテキストキャッシュの有効利用”, 情報処理学会論文誌コンピューティングシステム, Vol. 48, No. SIG 3, pp. 1-12, Feburary, 2007.

○金広文男, 石綿陽一, 齋藤元, 赤地一彦, 宮森剛, 五十棲隆勝, 金子健二, 比留川博久, “実時間 Ethernet を用いたヒューマノイドの分散 I/O システム”、日本ロボット学会誌, Vol. 25, No. 3, (掲載予定), April, 2007.

○K. Yamamoto, Y. Ishikawa, T. Matsui, ″Portable Execution Time Analysis Method,″ the 12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, pp. 267 - 270, 2006.

(2)特許出願

平成 18年度特許出願:1 件(CREST 研究期間累積件数:3 件)