XMCAPI: 組込み機器向け on-chip/off-chip コア間通信機構

三浦信一[†1] 塙敏博[†1,†2]
朴泰祐[†1,†2] 佐藤三久[†1,†2]

1. はじめに

近年の半導体集積技術の向上や求められるデータ処理量の増加に伴い，組込み機器に用いられるプロセッサチップには同一チップ上に複数のプロセッサコア（以後，コア）を配置するマルチコアプロセッサ技術が使われている．IA32 アーキテクチャに代表される汎用マルチコアプロセッサでは，チップ上にホモジニアスなコアが均一に配置され，キャッシュ一貫性制御を持った共有メモリシステムを持つのが一般的である．しかし組込み機器では，多様なサービスを提供するために，機能に応じた異種プロセッサを複数配置することも多く，また必ずしも共有キャッシュや共有メモリが提供されるとは限らない．したがって，これらのコア間で協調動作する場合，何らかのデータ通信機構が必要になる．そのため，これらのコア間の通信をアーキテクチャに依らず記述できる，移植性の高いプログラムインタフェースとして，Multicore Association[1] により，コア間の通信に特化して通信のためのオーバヘッドを削減し，メモリフットプリントを小さくした，MCAPI（Multicore Communications API）がコア間の通信 API として標準化されている[2]．

一方，組込み機器のプロセッサには，より一層の性能向上が要求される．しかし組込み向けのマルチコアプロセッサにおいては，パッケージサイズや消費電力などの観点から，同一チップ上に配置可能なコア数には限界がある．そのため，組込み機器においてもプロセッサチップ自体を複数配置し，これらをネットワークで結合し協調動作させることで性能を向上させる必要がある．このような環境において，プロセッサチップ間でもデータ交換や同期などにおける通信が必要になる．それらの通信 API は，マルチコアプロセッサ内の各コア間の通信と同様に，アーキテクチャ毎に異なり規格の統一性がない．このことから，今後マルチコアプロセッサ内部のコア間通信と同様に，組込み機器向けのマルチプロセッサ環境における，各プロセッサチップ間の通信 API の標準化が必要になる．

我々はマルチコアプロセッサ向けにチップ内のコア間通信として仕様策定された MCAPI を，異なるチップに存在するコア間の通信にも用いることを提案する．機種に依存しない統一された API が提供されることで，組込み機器のソフトウェア開発や移植が容易になる．我々は MCAPI の API の仕様に基づき，新たに on-chip および off-chip のコア間通信 API として XMCAPI を提案・開発している[3]．

†1 筑波大学 計算科学研究センター
†2 筑波大学大学院 システム情報工学研究科

2. XMCAPI

MCAPI は，マルチコアプロセッサチップ内のプロセス間通信を目的とした API である．コア間データ転送の物理的な手段として，共有メモリやオンチップネットワークを想定して規格化されている．MCAPI は socket や MPI[4] と非常に近い API の体系になっているが，MCAPI はマルチコアプロセッサ内ネットワークが持つネットワークの特徴を生かした軽量な API である．チップ内外によらず，同様の通信 API を複数のチップで構成されたコア間通信に用いることができれば，シームレスな通信を実現でき，チップ間に跨るプロセスの割り当てを自由に行える．我々は，この MCAPI を，何らかのネットワークで接続された，複数のマルチコアプロセッサチップで構成されたシステムに適用することを検討し，MCAPI で規格化されたチップ内のコア間通信 API を，チップ外のコアとの間にも利用する，XMCAPI (eXtended MCAPI) を提案している[3]．

現在公開されている MCAPI の実装は，Multicore Association が提供するリファレンス実装のみとなっている．このリファレンス実装は，共有メモリを想定し System-V shmem を用いた実装であり，共有メモリを持つ SMP 構成のマルチコアプロセッサのチップ内通信を除いて使用することができない．XMCAPI の実装は，下位の通信レイヤにおいて様々な物理通信インタフェース，通信プロトコルを想定する．まず，XMCAPI の実装の 1 つとして，下位の通信 API として TCP socket を用いた XMCAPI を実現する．本実装を xmcapi/ip ライブラリと呼ぶ．xmcapi/ip は，標準的な通信プロトコルである TCP を用い，主に Ethernet を物理インタフェースとして用いる．一般的な TCP を用いるため，Ethernet に限らず他の TCP を提供するネットワークにも適用可能で移植性が高い．

表 1 評価環境

Item	Specification
CPU	Intel(R) Xeon(R) E3110 @ 3.0GHz
Memory	DDR2-800 Dual Channel 8.0 GBytes
NIC	Intel Pro/1000PT Dual port
OS	Linux Kernel 2.6.31.12
CC	gcc 4.4.4

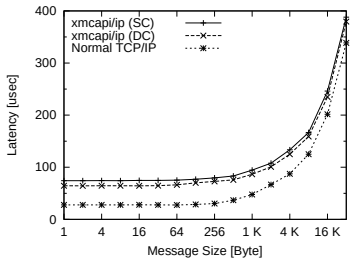

図 1 通信遅延時間 （MCAPI Message）

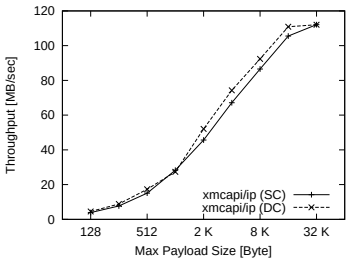

図 2 スループット （Packet Channel）

3. 性能評価

2 ノード間での通信性能を評価する．xmcapi/ip の実装は，ユーザプログラム部分と通信部分の計 2 つのスレッドが存在するマルチスレッド動作になる．本評価では，これらの影響を比較するために，各ノードのプロセッサがシングルコア（SC），デュアルコア（DC）の 2 種類で評価する．評価内容は MCAPI の代表的な通信タイプである Message 及び Packet Channel である．**表 1** に評価で用いたノード構成を示す．

図 1 に，Message を用いた 2 ノード間での 10,000 回の Ping-Pong 通信における，片方向平均通信時間を示す．シングルコア環境，デュアルコア環境の Message 1 Byte のデータ転送時間は，それぞれ約 74 μsec，64 μsec となり，メッセージサイズが 512 Byte 以下の場合でほぼ一定であった．シングルコア環境ではマルチスレッド動作によりオーバヘッドが現れ，約 10 μsec の差となった．処理内容としては基本的に通信のみであり，ユーザスレッドの処理時間の多くは通信完了の待機となる．その結果，シングルコア環境においても，プロセッサコアの資源競合は発生しない．しかし，実際の MCAPI アプリケーションでは計算と通信のオーバラップが発生する．そのためシングルコア環境下における実アプリケーションでは，本評価結果よりも大きな性能低下が予想される．また，xmcapi/ip 自体のオーバヘッドを見るために，同時に標準的な Socket と TCP/IP を用いた場合の遅延時間についても示した．この場合，Message 1 Byte のデータ転送時間は，約 28 μsec であった．この結果，xmcapi/ip を用いたデュアルコアの結果は，Socket を直接利用するよりも約 36 μsec 遅延時間が上乗せされる．

図 2 に Packet Channel を用いた場合のスループットの評価結果を示す．評価では 2 ノード間で計 1.0 GByte のデータの送信に要した時間を計測し，スループットを算出した．最大のスループットはシングルコア環境，デュアルコア環境で共に最大転送サイズ 32 KByte 時に約 112 MByte/sec となった．これは Gigabit Ethernet の最大性能 125 MByte/sec に対して約 90%の性能であり，実用上ほぼ十分な性能が得られている．このことから，1 度の転送サイズが大きいようなアプリケーションにおいては，シングルコアのような環境においても，xmcapi/ip の実装は有効に機能すると期待できる．

4. おわりに

本稿は，プロセッサコア間通信 API として標準化されている MCAPI を拡張する XMCAPI の提案を行った．また，XMCAPI の具体的な実装として xmcapi/ip を実装した．xmcapi/ip は通信に TCP socket を用いることで，既存の Ethernet を用いた通信を可能にし，アプリケーションの開発・デバック時における，実システムを必要としてない開発環境をユーザに提供する．実装した xmcapi/ip を用いた基礎的な性能評価の結果，通常の socket API を直接使う場合と比較して，バンド幅においてはオーバヘッドなしで XMCAPI を使用できる．一方で，遅延時間は比較的大きいという問題がある．xmcapi/ip の目的は，複数のチップ・ノードから構成されるシステム向けの MCAPI 環境のリファレンス実装であり，これらの性能は大きな問題にはならない．しかし，今後より現実的なシステムに適用するために，より高性能なネットワーク環境に適用できる XMCAPI を実装する必要がある．

謝辞 本研究の一部は，科学技術振興機構戦略的創造研究推進事業（CREST）研究領域「実用化を目指した組込みシステム用ディペンダブル・オペレーティングシステム」，研究課題「省電力高信頼組込み並列プラットフォーム」による．

参考文献

1) Multicore Association: http://www.multicore-association.org/.
2) Multicore Communications API Working Group: Multicore Communications API, http://www.multicore-association.org/workgroup/mcapi.php.
3) 三浦信一ほか：組込み機器向け on-chip/off-chip コア間通信機構, 情報処理学会研究報告, Vol.2009-ARC-184, No.2, pp.1–7 (2009).
4) Message Passing Interface Forum: MPI: A Message-Passing Interface Standard (1994).