静岡大学　　博士論文

モバイルコンピューティング環境における メモリ管理方式に関する研究

2000年　12月

大学院理工学研究科
設計科学専攻

横山 繁盛

論文の要旨

近年のコンピュータ技術、有線／無線通信技術の発展とその普及は目覚ましく、パーソナルコンピュータ(PC)や携帯電話、インターネットの急速な普及によりモバイルコンピューティング環境が整いいつつある。特に最近になりiモード、EZweb、J-skyなど、インターネットとの接続機能をもった携帯電話が爆発的に普及している。これはどこからでもいつでも情報の送受信が可能という、これまで未来の夢の技術と考えられていたことが、その一部ではあるが実現可能になったことによると思われる。

しかしながらモバイル端末とサーバとで構成されるモバイルコンピューティングシステムは、無線通信の帯域幅の問題や接続の継続性、通信コスト、モバイル端末のバッテリーの持続時間等の課題が多い。さらにモバイル端末とサーバとが連携したアプリケーションにおいては通信処理があるため、その構築が複雑であり、一般的に普及しているアプリケーションは限定されたものにとどまっている。

本研究では、モバイルコンピューティング環境下における、アプリケーション構築の容易化、サーバとモバイル端末間でのデータの同期の容易化、および通信の効率化等を目的とし、モバイルコンピューティング環境向きメモリ管理方式(MMM)を提案する。これはモバイル端末の一部のメモリ領域と、サーバのメモリの一部の領域を共通メモリとし、この内容の一貫性管理を行い、共有できるように構成することで、モバイル端末やサーバから自由に共通メモリにアクセス可能にする、モバイルコンピューティング環境向きのメモリ管理方式である。各メモリ領域をラインと呼ぶ一定の単位に分割し、ラインを基本の単位として共通メモリの管理を行う。ラインをさらにブロックに分割し、データ転送をブロックの単位で管理することにより、通信トラフィックの減少を図る。MMMは、モバイル端末とサーバとが1:1の関係で同一メモリ空間を共有する基本メモリ管理方式(基本MMM)と、複数モバイル端末とサーバがn:1の関係で同一メモリ空間を共有する拡張メモリ管理方式(拡張MMM)から成る。拡張MMMでは複数モバイル端末間での同一メモリ領域の競合管理が可能である。さらにメモリの一貫性管理方式により基本MMMでは、基本MMM管理方式1、基本MMM管理方式2、基本MMM管理方式3の

3種の方式、拡張MMMでは、拡張MMM管理方式1、拡張MMM管理方式2の2種の方式を提案する。各方式はメモリの一貫性管理方式と並列動作性に関し異なり、アプリケーションの種別により適切に使い分けることができる。本研究ではさらに、次に使用する可能性のあるデータを、通信の空き時間を利用して、プリフェッチすることを提案する。モバイルコンピューティング環境化でのアプリケーションは、インタラクティブなものが多いと考えられるため、モバイル端末の使用者からの応答待ち時間を利用してデータのプリフェッチを行い、実行時間の短縮と通信の効率化を図る。MMMはまた、既存の標準的なアーキテクチャを持つプロセッサに対して、比較的簡単なハードウェアを付加することにより、性能に影響を与えることなく実現可能である。

メモリ管理方式には、メモリアクセスの高速化、大容量化、プログラミングの容易化等のため、従来よりキャッシュメモリ、仮想メモリ、分散共有メモリ等の重要な技術があり、幅広く研究、実用化されている。MMMはこれらの技術と類似した側面を持つが、共通化したメモリを共有する方式という面で、従来の技術とは異なったものである。

アプリケーションとして、スケジュール管理、住所録管理等を想定し、モバイル端末とサーバとが連携した動作のシミュレーションを行い、従来の一括伝送方式に比べ通信時間が減少し、アプリケーションの実行時間が短縮されることを示す。さらに通信の空き時間を利用したデータのプリフェッチを行うことにより、アプリケーションの実行時間が短縮できることを示す。また通信の課金方式が、接続時間による時間課金かデータ量による従量課金かにより、より有利な方式を選択することができることを示す。

以上の方式を採用することにより、モバイル端末やサーバでのアプリケーションは、共通メモリの領域を単なる自身のメモリとしてアクセス可能となる。モバイル端末とサーバとが連携するアプリケーションでは、通信を意識する必要がなくなるためプログラムの構築が容易となり、またモバイル端末とサーバと間でのデータの同期が容易となる。さらに必要とするデータのみの通信により、通信の効率を上げることができ、通信時間の短縮はモバイル端末の電力の節減にも寄与する。

目次

1. 緒論

1.1 研究の背景とその目的

近年のコンピュータ技術、及び有線／無線通信技術の発展、機器の低価格化やパーソナルコンピュータ(PC)の普及、通信網の整備、携帯電話やPHSの急速な普及により、インターネットの利用が急速に拡大しつつある。未来の夢であったモバイルコンピューティング環境も急速に整いつつある[1,2]。PHSを含む携帯電話の普及台数は2000年度で一般電話の普及台数約6000万台を超えたと言われている。さらにまた携帯電話では、iモードやEZweb、J-skyなどの簡易インターネット接続機能を持った機種が1999年2月に投入されると爆発的に普及し、これら機種の累計普及台数は、2000年10月末には2,180万台(注1)に達している。

しかしながらモバイルコンピューティング環境は、モバイル端末においてはバッテリー駆動のため使用可能エネルギーに制約のあることや、携帯性の面からは、質量、大きさに対する厳しい要求があり、モバイルコンピュータには、従来の据え置き型のコンピュータにはない様々な制約がある。さらにまた無線通信環境は、有限の周波数帯域を共用することによる伝送帯域幅の制約や、無線通信であることによる接続の不安定性を本質として持つ[3]。このためモバイル端末とサーバとの無線通信は、処理が複雑となり、また無線通信の種類によりその処理が異なる。これらの制約事項によりモバイル端末とサーバとが協調する一般的なアプリケーションは、インターネットのWWWの閲覧やメールの送受信等の限定された種類にとどまっている。

本研究では、これら課題の一部を解決するため、モバイルコンピューティング環境下で、アプリケーション構築の容易化、サーバとモバイル端末間でのデータの同期の容易化、通信の効率化等を目的とし、モバイルコンピューティング環境向きメモリ管理方式(MMM)を提案する。MMMは、モバイル端末とサーバの一部のメモリ領域を共通メモリとして、メモリを共有することにより、サーバやモバイル端末のアプリケーションが通

注1 (社)電気通信事業者協会　http://www.tca.or.jp

信を意識することなく、共通メモリを自身のメモリとしてアクセス可能な、モバイルコンピューティング環境向きのメモリ管理方式である。これによりモバイル端末やサーバのアプリケーションは、共通メモリ領域を単なるメモリとして参照可能となり、モバイル端末とサーバとが連携したアプリケーション構築の容易化、モバイル端末とサーバ間のデータの同期、及び必要とするデータのみの通信による通信の効率化が可能になる。さらに通信が時間課金方式の場合には、モバイル端末のアプリケーションの利用方法の特徴を利用したデータのプリフェッチを導入することで、実行時間の短縮や、通信接続時間の短縮が可能となる。またモバイル端末がサーバの共通メモリ領域を自メモリとしてアクセス可能になることで、サーバの資源を最大限に活用し、モバイル端末の資源の利用を最小化したアプリケーションの構築が容易となる。

1.2 論文の構成

第2章では、まず急速に発展しているモバイルコンピューティング環境に関し、各種モバイル端末の概要と代表的な無線通信システムの概要、およびそれらの課題について述べる。第3章ではコンピュータ技術分野の中でモバイルコンピューティング技術に関連する従来技術を概観する。まずモバイルコンピュータ向けハードウェア構成方式について概説する。メモリ管理方式については、モバイルコンピュータ向け専用ではないが、すべてのコンピュータ技術の基本であり、モバイルコンピュータのアーキテクチャを考える上での必須技術であり、本論文で述べるモバイルコンピューティング環境向きメモリ管理方式の基礎を成すため、基本事項について概説する。最後にソフトウェアの方式であるモバイルコンピューティング向きのファイル管理方式と、アプリケーションの構築手法についての代表的な例について紹介する。

第4章では、モバイルコンピューティング環境向きメモリ管理方式(MMM)の中の基本メモリ管理方式(基本MMM)のアーキテクチャとその特徴について述べる。基本MMMにはメモリの一貫性管理方式の違いにより基本MMM管理方式1、基本MMM管理方式2、及び基本MMM管理方式3の3種のメモリ管理方式があり、その制御方式と特徴、使い分けについて述べる。

第5章では基本MMMについて、他メモリ管理方式との差異、その適用分野、ライン

サイズやブロックサイズの選択、シミュレーションによる評価について述べる。

第6章では、第4章で述べた基本MMMを拡張した拡張メモリ管理方式(拡張MMM)について述べる。拡張MMMは、複数端末が同一メモリ空間を同一時間に共有可能であり、アーキテクチャとその応用、及び評価について述べる。

1.3 発表論文との関連

2000年の9月号の情報処理学会論文誌に発表した論文、及び2000年の7月と9月の国際会議で発表した3件の論文は、いずれも基本MMMに関してそのアルゴリズムと評価に関して発表した論文である。研究会や全国大会への発表は順次基本MMMの研究の進展に従って発表したものである。第6章で述べる拡張MMMの研究は、他では未発表である。

1.4 先行研究との関係

主たる通信手段が無線通信であり、モバイル端末のメモリとサーバのメモリとで、一部のメモリ領域を共通メモリとして管理することを特徴とする、メモリ方式に関する本研究は、これまで調査した内外の代表的論文誌の範囲では先行研究事例の記載はない。メモリ方式については、キャッシュメモリ、仮想メモリ、分散共有メモリという、コンピュータ技術にとって極めて重要な技術があり、数多くの研究と、実用化が行われているが、MMMはこれらメモリ方式と、ある面では類似点を持つが、いずれの方式とも異なっている。その差異については5.2節で述べる。なお基本MMMに関しては、1999年に「メモリ管理方式及びコンピュータ」の名称で特許出願し、2000年に特許公開されている。

2. モバイルコンピューティング環境

2.1 緒言

モバイルコンピューティング環境は、携帯可能なモバイル端末、事務所等に設置された据え置き型のコンピュータであるサーバ、及びそれらを結ぶ無線及び有線の通信システムより構成される。なおモバイル端末の使用開始時にサーバよりデータをダウンロードし、その後はオフラインで単独使用し、使用後はサーバに接続してデータをアップロードするという使い方も、モバイルコンピューティングと考えてよいであろう。この章ではモバイル端末と無線通信環境について概観する。

2.2 モバイル端末

主要なモバイル端末には、携帯可能なコンピュータであるモバイルコンピュータ、PDA(Personal Digital Assistant)と呼ばれる個人用携帯情報端末、業務用途の各種専用情報端末、メールやインターネット接続機能を持つ携帯電話等が含まれる。

2.2.1 モバイルコンピュータ

モバイルコンピュータは、移動可能なコンピュータであり、一般PC用と同じ汎用のオペレーティングシステム(OS)が搭載可能なノートPCやペンPC、モバイルコンピュータ用のOSを搭載するハンドヘルドPC、さらに小型用の専用OSを搭載するパームサイズPCに分けられる。パームサイズPCはまたPDAの一種に分類することもできる。

ノートPCは、汎用OSを搭載し、ハードウェア機能が据え置き型PCと互換性が要求されるため、大きさと質量を減らすのに限界がある。ディスプレイの解像度は最低でも640×480以上のカラーが必要であり、キーボードについてもキーピッチやキーストロークが操作性の面より、一般のキーボードに近いものが要求されるため、大きさは最低でもA5サイズ程度以上が必要であり、現在のモバイルPCの主流はB5サイズである。CPUは、一般のPC用と互換性が要求されることや、ハードディスク(HDD)の搭載が必要であること、ディスプレイの消費電力等により、最軽量クラスのもので1～1.5Kg、バッテ

リーの持続時間は1～2時間にとどまっている。

ペンPCは、液晶ディスプレイにポインティング機能を持たせた汎用OSを搭載するモバイルPCであるが、一般用途には普及せず産業用の利用にとどまっている。

一方ハンドヘルドPCは、モバイルコンピュータ用の専用OSを使用し、HDDの必要はなく、機能も限定されているため、ハードウェアは小型化、軽量化することができ、質量も1Kg以下となっている。しかしながらアプリケーションが一般PCと互換性がないためか、ノートPCに比べると普及の度合いが低い。

パームサイズPCは、PalmOSとよばれる専用OSを搭載した手のひらサイズ型の超小型のコンピュータであり、最近急速に普及している。これは基本としてスケジュール帳、住所録、メモ帳等の従来の個人の手帳の機能持つものである。一般PCとの連携は現時点では、オフラインでシリアル通信や赤外線通信により行われる。CPUの能力や、メモリ容量、ディスプレイ表示能力を限定することで、大きさや質量の低減と、バッテリー動作の長時間化を計っている。ハードウェア機能は限定されているが、アプリケーションプログラムを入れ換えることによりメモリ容量の制約の範囲で各種の機能が実現できる。従来の電子手帳と機能的には似ているが、最大の違いは、アプリケーションプログラムの作成を容易にすることで第3者によるアプリケーションを増加させ、個人の好みでアプリケーションを入れ替えることでカスタマイズ可能にしている。今後携帯電話等の無線通信機能の利用により、インターネットやコンテンツサーバとの連携機能が強化されていくと思われる。

2.2.2 PDA

PDAは、個人用携帯情報端末と呼ばれ、各種の機種が出されており、ビデオカメラ機能やGPS等の機能を持つものがあるが、パームサイズPCを除くと各機種間の互換性はなく、各社それぞれ独自の機種となっている。

2.2.3 専用情報端末

専用情報端末は、主として各種の業務用途向けのモバイル端末である。ハンディターミナルと呼ばれる端末は、無線LAN等の無線通信機能を内蔵するものがあり、倉庫や工

場内管理用に使用されている。サーバとの連携をオフラインで行う端末は、レストラン内の販売管理、鉄道車両内での運賃の精算や商品販売管理、スーパーの在庫管理、宅配便管理等で幅広く使用されている。車載用の分野では、今後自動車カーナビゲーションを発展させた車載端末が、車載用のモバイル端末として普及、発展していくと思われる。

2.2.4 携帯電話

インターネット接続機能を持った、iモード、EZweb、J-skyなどの機能を持つ携帯電話もモバイル端末に分類することができる。これらの機種は、音声通話の他に、インターネットによるメールの送受信機能と、インターネットのホームページへの簡易アクセス機能という限定された機能を持つ。しかしながら、従来の音声通話用の携帯電話とほぼ同じ100 g以下の軽量化の実現と、携帯電話会社による情報料の代行徴収という、コンテンツプロバイダーによる情報提供の枠組みを提供したことにより、優良なコンテンツが増大し、爆発的な普及に至ったものと考えられる。これらが今後、機能と軽さ及びバッテリーの持続時間とのバランスをとりながら、どのように発展していくかの予測は難しいところである。

2.3 無線通信システム

2.3.1 無線通信の特徴

無線通信には、無線局が固定されている固定無線と,無線局が移動可能な移動無線があるが、本論文では、特にことわらない限り無線通信は移動通信を示す。点と点を結ぶ有線通信と異なり、移動通信は面での通信を可能とし、「いつでも、どこでも、誰とでも、どんな情報でも」通信を可能とする理想的な情報通信システムと言われている[4]。しかしながらこれらの特徴の元となる無線通信は、データ伝送の面から見ると、伝送帯域幅の制約、電波の弱い地域や障害物のある地域での通信の切断、携帯電話等のセルラー方式通信では、基地局をまたがって移動するときに基地局の切替え(ハンドオフ)による瞬断が発生するという課題がある。

2.3.2 広域通信

コンピュータにおいて広域での無線通信の利用が一般的に使用可能になったのは、携帯電話がデジタル化されデータ通信のサービスが開始されてから、及びPHSでのデジタル通信のサービス開始以降であろう。それ以前でも音声通信を使用したデータ伝送が可能であったが、実用データ伝送速度が1.2Kbps程度以下と極めて遅く、また接続の信頼性の低いものであった。現在のデジタル携帯電話は、PDC(Personal Digital Cellular)方式では、データ通信用として時間課金方式の9.6Kbpsと、データ量課金のパケット通信方式の9.6Kbpsまたは28.8Kbpsが、PHSでは時間課金方式の32Kbpsと64Kbpsが実用化されている。

現行の携帯電話システムは、デジタル化され第2世代と言われているが、全世界で見ると、日本のPDC、米国のAMPS(Advanced Mobile Phone System)、米国と日本の一部のCDMA(Code Division Multiple Access)、欧州と東南アジアの一部のGSM(Global System for Mobile Communications)等の複数の規格が並立している。このため、次世代の携帯電話システムとして、IMT-2000が国際標準として実現化に向けて開発がおこなわれている。IMT-2000の目標とする主要な特徴は次の通りである(注2)。

(a) 世界的に高度の共通性を持つ設計

(b) IMT-2000内及び固定網とでサービスの互換性

(c) 高品質

(d) 世界中で使用できる小型の端末

(e) 世界的なローミング機能

(f) マルチメディアアプリケーション及び広範囲なサービスと端末の能力

通信速度については、固定端末で2Mbps、携帯情報端末で最大384Kbps、自動車等の高速の移動体との通信では最大144Kbps、携帯電話端末で16Kbpsのデータ伝送速度が目標とされている[5]。

通信帯域幅に関しては、広域の公衆用の無線通信は当初はアナログ方式の音声通信が

(注2) International Telecommunication Union(ITU) htt p://www.itu.int/

主であり、データ伝送は、データ端末側で音声帯域の周波数で変調し音声として伝送するため、実用的には1.2Kbps程度以下であった。デジタル化されたときに、音声通話機能と共にデータ伝送の機能が携帯電話のサービス項目として追加され、データ伝送の信頼性が向上した。

2.3.3 近距離通信

(1) 赤外線通信

近距離通信では、赤外線を用いたIrDA(Infrared Data Association)と呼ぶ通信方式が規格化されており、最大4Mbpsまでの速度が定義され、機器間接続のための1m以内程度の近距離通信をおこなうことができる。ハードウェアは赤外線の送受信のための赤外線モジュールがあればよく単純である。電波を使用しないため地域による規制がなく、世界の国々で使用可能である。2.2.1項で述べたモバイルコンピュータには、基本的にはすべて、IrDAが標準で搭載されている。

(2) 無線LAN

無線LANはIEEEで規格化[注3]され、ISM(Industrial, Scientific, and Medical)周波数帯を使用し、FHSS(Frequency Hopping Spread Spectrum, 周波数ホッピング拡散方式[6])、またはDSSS(Direct Sequence Spread Spectrum, 直接拡散方式[6])のスペクトラム拡散方式による通信を行う。IEEE802.11[7]では通信速度が2Mbpsまでであったが、IEEE802.11b[8]で11Mbpsに仕様が拡大された。事務室や家庭内でのLAN接続のコードレス化や、工場内や倉庫等の構内でのワイヤレス通信用として利用されており、百m程度以内の通信が可能である。

[注3] IEEE P802.11, The Working Groupe for Wireless Lans http://grouper.ieee.org/group/802/11/index.html
[注4] The Official Bluetooth SIG Website http://www.bluetooth.com/

(3) Bluetooth

Bluetoothは、機器間の近距離通信を目的とし、1999年にBluetooth SIG(Special Interest Group)により制定された[注4]。ISM(Industrial, Scientific, and Medical)周波数帯を使用し、FHSS方式のスペクトラム拡散方式を採用している。音声用では64Kbps、データ通信用では最大723.2Kbpsまでの速度が定義されている。通信は1:1または1:Nのマスタースレーブ方式を基本としている[9]。

2.3.4 課金方式

広域の公衆用の無線通信網は、第1世代のアナログ携帯電話の時代では、データ通信を行う場合には、端末側で音声帯域に変調し音声に変換する必要があり、またきわめて信頼性の低いものであった。通信の高品質化、周波数帯域利用の効率化のため、デジタル化され第2世代となり、PDC方式では、この時点で音声通話に加え9.6Kbpsのデータ通信のサービスが加わった。このデータ通信の課金方式は、音声通話と同じ課金体系であり距離と時間により課金される。この後9.6Kbpsまたは28.8KbpsのDopaと呼ばれるパケット通信方式のデータ通信のサービスが追加され、さらにこのパケット通信を利用しiモードのサービスが開始された。パケット通信は情報量による課金方式であり、接続時間や距離によらずパケットと呼ばれる128バイトを単位とした情報量により課金される。CDMA方式においてもPacketOneと呼ばれる14.4Kbpsまたは64Kbpsのパケット通信サービスが開始された。

2.4 結言

本章では、モバイルコンピューティング環境を構成するモバイル端末と無線通信システムについての概要、及びそれらの持つ課題と今後について概観した。モバイル端末や無線通信システムの技術の進展や普及は目覚ましい。しかしながら無線通信はその本質上通信の切断は避けられず、また無線の周波数は有限の共有資産であり、携帯電話等の公衆通信システムについても各種無線システムとの共存が必要である。このため使用できる周波数帯域も制約がある。したがってシステム的な工夫によりこれら課題解決が必要である。

3. 従来の研究の概観

3.1 緒言

この章では、コンピュータ技術分野に関し、モバイルコンピューティング技術の、主として小型化、軽量化、電力節減、通信の切断への対応について、ハードウエア及びソフトウェアの従来研究を概観する。一般のコンピュータの基本技術であり、本研究のモバイルコンピューティング環境向けメモリ管理方式の基本を成すため、従来のメモリ管理方式についても概観する。

3.2 ハードウェア構成方式

モバイル端末の課題は、軽量化とバッテリによる動作時間の増大である。しかしながらバッテリの容量増加と軽量化は相反しており、消費電力の低減が求められている。半導体プロセスの微細化、部品の小型化や軽量化により、各部品レベルでの低消費電力化が進展している。CPUについても製造プロセスの微細化や動作電圧の低下により低消費電力化[10]が図られているが、一方高速化のため回路数の増加による消費電力の増加傾向も見られる。一方、方式的な工夫での電力削減も各種の研究が行われており，CPUの使用状況によりCPUの速度を可変にして低消費電力化を図る方式[11,12]、使用していないIOの動作を停止する方式[13]、CPUのアイドル状態の時に低消費電力状態に移行させる自動スリープ機能の方式[14]、バッテリーのパルス放電方式により持続時間の増大を図る方式[15]、通信処理の効率化により通信時間を減らして低消費電力化を図る方式[16]、PDA等の超小型端末の近距離通信用にRF部の低消費電力化を図った無線通信方式を採用した研究例[17]等がある。さらに無線通信処理が比較的電力消費が多いことより、通信のエラー制御方式の工夫により低電力化を図る研究例[18]もある。

小型軽量化や消費電力の観点より、モバイル端末の機能を通信と表示の機能に特化し、他をサーバ側に持たせるアーキテクチャとしたInfoPadと呼ぶ端末の研究例がある[19]。これはCPU能力の要求されるプログラムをサーバ側で実行させ、端末には表示と通信に特化させることにより、電力消費量の多い高性能CPUの替わりに低消費電力CPUを使

用した端末方式である。

商用事例では、ハードウェアと基本ソフトウェアが協調することで省電力を実現するACPI(Advanced Configuration Power Control Interface)[20]と呼ぶ省電力方式を規格化しノートPCで採用されている。これはアプリケーションが動作していない時に、CPUのクロック周波数の低下、動作していないI/Oコントローラのクロックの停止、バッテリー動作の場合には液晶ディスプレイの輝度を下げる等の省電力制御の枠組みを提供する。さらに最近になると、モバイルコンピュータ用のCPUでは、バッテリー電源とAC電源とでCPUのクロック周波数と電圧を可変にし、バッテリー電源使用時にはクロック周波数とCPUのコア電圧を下げて電力を節減する機能を持つものが出ている。

マイクロプロセッサの構成方式としては、いわゆるCISC型とRISC型があるが、汎用のPCの世界ではこれまでCISC型が使用されてきた。これは過去のソフトウェア財産の継承のためソフトウェアインターフェースの互換性を保つためである。一方RISC型CPUは、高速化のため論理を単純化し、一部の機能はソフトウェアにより実現し、過去のソフトウェアとの互換性は切り捨てることで高性能化を実現した。RISC型CPUは、UNIX搭載のワークステーション用のマイクロプロセッサや、組み込み型のマイクロプロセッサで実用化されている。2000年の始めに、これまでCISC型のCPUが使用されてきた汎用OSを搭載するモバイルPC向けに、低消費電力をうたったRISC型のVLIW(Very Long Instruction Word)方式CPUを採用したCrusoeと呼ばれるCPUが発表された(注5)。これは、RISC型とすることでCPUを単純化し、高速化と低消費電力化を図り、ソフトウェアによるエミュレーションにより従来のCPUとのソフトウェアインターフェースの互換性を保ちつつ、高性能と低消費電力化を図ったとされている[21]。

3.3 メモリ管理方式

ストアドプログラム方式の計算機の発明以来、計算処理の高速化や大規模化のため、処理装置の高速化と共に、メモリの高速化や大容量化のため多くの研究がおこなわれ実用化されてきている。メモリの高速化には、メモリ素子自身の高速化とメモリの方式技

(注5) Crusoe - a new world of mobility from Transmeta http://www.transmeta.com/

術による高速化がある。

メモリ高速化の方式技術の中で特に重要なのはキャッシュメモリ[22-27]である。キャッシュメモリは、その有用性や、半導体技術の進展によるメモリの高速化、大容量化、低価格化及び高集積化により、現在では比較的廉価なワンチップマイクロコンピュータ[28,29]にも採用されるようになっている。またキャッシュメモリの方式についても各種実用化され[30]、また後に述べる他のメモリ方式の中でも重要な役割を担っている[31]。さらに、CPUの高速化やメモリの大容量化に向けてのキャッシュメモリを複数の階層で持つメモリシステムの研究[32-34]もある。

一方処理速度の高速化のため、各種の処理方式の工夫により、処理装置自身の高速化が図られたが、さらにそれを超えた高性能化のため、複数処理装置で構成される各種コンピュータシステムの方式があり、単一アドレス空間のメモリを複数の処理装置で共有するマルチプロセッサシステム、メモリを共有しないマルチコンピュータシステム[53]、処理装置毎にメモリを持ち、それらのメモリが単一アドレス空間のメモリ空間となるように制御する分散共有メモリシステム等について、各種方式が研究され実用化されている[30,35-50]。

単一メモリ空間を共有するマルチプロセッサは、メモリアクセスの通信路(バスや高速スイッチ等)のボトルネックのため接続できる処理装置数に制約があり、大規模な構成向きでない。一方マルチコンピュータは、大規模な構成は可能であるが、メモリを共有していないため、プログラムの構築が複雑であり、異なるシステムへのプログラムの流用(移植)性に難点がある。大規模なシステム向けとしては、プログラムの容易化や、汎用性のために、単一のメモリ空間を持つ分散共有メモリ方式が適している。これは物理的に離れていたり異なっていたりする構成を持つメモリを、一つのアドレス空間となるように構成する方式である。各種の制御方式が研究されており[35-50]、さらに並列して動作するプロセッサ間の同期方式に関する研究[51]、分散共有メモリのキャッシュの一貫性制御をアプリケーションに応じてカスタマイズ可能とする研究[52]、性能向上のため一貫性制御を緩めた各種の一貫性制御方式に関する研究[53-59]等がある。これらの単一メモリ空間のシステムにおいても、プログラムの効率的な実行のためには、データの配置等についてメモリの物理的構成を考慮したプログラミング[60,61]が必要である。した

がってアプリケーションの効率は共有メモリ方式に依存する。なお共有メモリや分散共有メモリでは、処理装置とメモリ、またはメモリ間は、高速の専用バスや高速の通信網で接続されるが、処理性能を制約する大きな要因の一つのである。

一方、メモリ空間の拡大やマルチプログラミング、プログラムの再配置等の容易化のため仮想メモリ方式が提案された[62]。これは計算処理の高速化用ではなく、物理構成を超えた大規模なプログラムの構築や、基本プログラムによるメモリ管理容易化が目的である。当初は汎用計算機で実用化されたが[63]、現在ではマイクロプロセッサでも広く一般的に採用されている技術である。

3.4 ソフトウエア構成方式

モバイルコンピューティング環境においては、必要な時に必ず通信できるわけではなく、また一度接続ができても通信の途中で切断される可能性があり、これらの環境下でサーバとクライアントが協調したシステムの構築法としては、オペレーティングシステムの側で対応する方式、アプリケーション側で対応する方式、および両者が協調して対応する方式がある[64]。オペレーティングシステムによる方式としては各種のファイル管理方式がある。

(1) ファイル管理方式

Coda[63-65]は、比較的信頼度の低い大多数のクライアントと、少数の高信頼度のファイルサーバが、高速のネットワークで接続されるシステムを想定したファイル管理方式である。科学技術計算分野で典型的なデータのアクセス向けであり、高度の並列性や細粒度のデータアクセス向けは意図していない。複製を持った複数のサーバと、クライアントでのキャッシュにより、通信の一時的な切断に対応可能にしている。

Bayou[68]のシステムは、クライアントがサーバの複製データに対してリードまたはライトを行う時に他の、複製と明示的な協調を行わない。サーバへの複数の書き込でコンフリクトが発生した時には、アプリケーションに依存したコンフリクトの解決とマージ処理を行う。複数のサーバ間での複製データの同期処理は、通信接続をしているコンピュータ間で一斉に行うのではなく、対向した2台のコンピュータ間で行うことを順次

繰り返すことで最終的に一貫性を保つ方式とすることにより、通信の一時的な切断に対応可能にしている。

(2) アプリケーション構築法

モバイルコンピューティング環境下でのアプリケーション構築のためのRoverToolkit [69,70]は、通信の切断の可能性のある環境化で、モバイル端末とサーバとが連携する信頼性のあるアプリケーションの構築法を提案している。ここではRelocatable dynamic objects (RDOs)と、Queued remote procedure call (QRPC)という2つのアイデアを提案し、アプリケーションが、切断の可能性のある通信環境で信頼性のある通信を可能とするツールを提供している。

Odyssey API[71]は、アプリケーションとシステムが協調することにより、モバイルコンピューティング環境に合わせた、アプリケーションを構築可能にするための手法を提供している。

WebExpress[72]は、モバイルコンピューティング環境下でWWWをアクセスする時に、クライアントとサーバから透過な、トラフィック軽減のためインターセプト技術を提案している。

ALICE[73]は、クライアントとサーバで構成される分散コンピューティング環境下で、オブジェクト指向アプリケーションを構築するための枠組みを提供するCORBA (Common Object Request Broker Architecture)を、モバイルコンピューティング環境下で適用可能にするための、ALICEと呼ぶアーキテクチャを提案している。

3.5 結言

モバイルコンピューティング関連の従来の研究と、モバイルコンピュータを含めたコンピュータの基礎技術としてのメモリ管理方式について概観した。分散共有メモリ方式の一つの目的は、複数のプロセッサを効率的に動作させるためのアプリケーションを容易に構築できるようにすることであった。第4章以降で、モバイルコンピューティング環境下でのアプリケーションの構築の容易化と、通信の効率化を図ることを目的とし、モバイルコンピューティング環境向けメモリ管理方式について提案する。

4. モバイルコンピューティング環境向きメモリ管理方式

4.1 緒言

2章で述べたように、モバイルコンピューティング環境が急速に実現されつつある。しかし現状は、無線通信の伝送帯域幅が狭く、接続の継続性に難点があり、またモバイル端末においてはバッテリーの動作時間が短く、CPU性能、メモリ容量、入出力装置等に制約がある等の課題が多い。またモバイル端末とサーバとでデータを共有する場合には内容の同期が課題である。さらにモバイル端末とサーバとが連携したアプリケーションでは、モバイル端末とサーバとの間で通信が必要であり、通信も各種の媒体や通信方式があるため、その構築が複雑となる。

本章では、よりハードウェアに近い方式である、モバイル端末とサーバのメモリ領域の一部を共通メモリとして制御することを特徴とする、モバイルコンピューティング環境向きメモリ管理方式、MMM(A Memory Management Architecture for Mobile Computing Environment)を提案する。

MMMにおいて、モバイル端末は、必要とするデータの存在するサーバ上のメモリ空間を、自メモリ空間の一部として制御することにより、アプリケーションは、通信を意識することなく、サーバ上のデータを単なるメモリとしてアクセスすることができる。サーバについても同様の制御により、モバイル端末上のデータを単なるメモリとしてアクセスできる。これによりアプリケーションの構築の容易化、データの同期化を図ることができ、さらに必要時、必要な量のデータ通信により、通信による遅延の最小化と、通信コストの低減が可能となる。

MMMは、基本MMMと拡張MMMから成り、本章では基本MMMのアーキテクチャについて述べ、その評価については第5章で論じる。拡張MMMについては第6章で述べる。

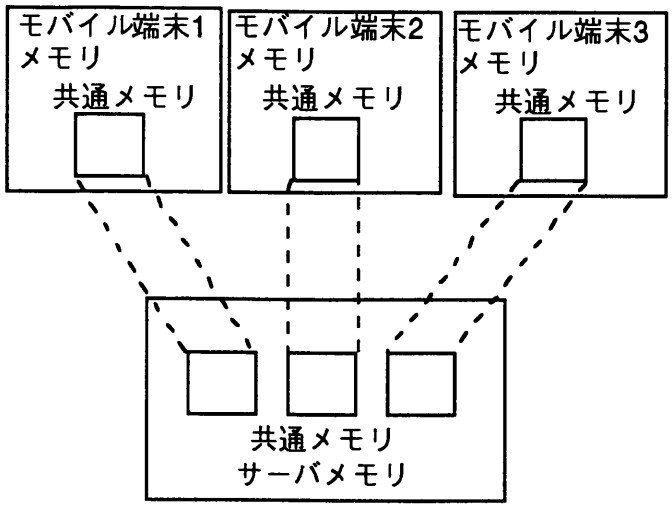


図 4-1 基本 MMM のメモリ管理方式

4.2 基本 MMM のアーキテクチャ

4.2.1 概要

MMM は、モバイル端末とサーバのメモリ空間の一部を共有する方式であり、一種の分散共有メモリと考えることができる。

一般の分散共有メモリは、ネットワーク上に分散したコンピュータのメモリを共有する方式であり、通信の帯域幅は比較的広く、任意のコンピュータ間で必ず通信ができることを前提としている。単一のCPUでは得られない大容量高速のデータ処理性能を得るために、多数のCPUを連携して動作させることを主目的とし、メモリを単一メモリ空間とすることにより、プログラミングの容易化を図った方式と考えることができる [35-50]。

これに対して、MMMでは携帯電話等の無線通信を基本とし、帯域幅が狭く切断の可能性がある通信路を前提として、モバイル端末とサーバ間とでのメモリの共有を行う方式である。モバイル端末のアプリケーションが通信を意識することなく、共通メモリをアクセスすることにより、サーバ側のCPUやメモリ、入出力装置等のリソースを容易に利用可能とし、低速で切断の可能性のある通信路、及び通信非接続での利用が可能な、モバイルコンピューティング環境向きのメモリ方式である。さらに標準的なアーキテク

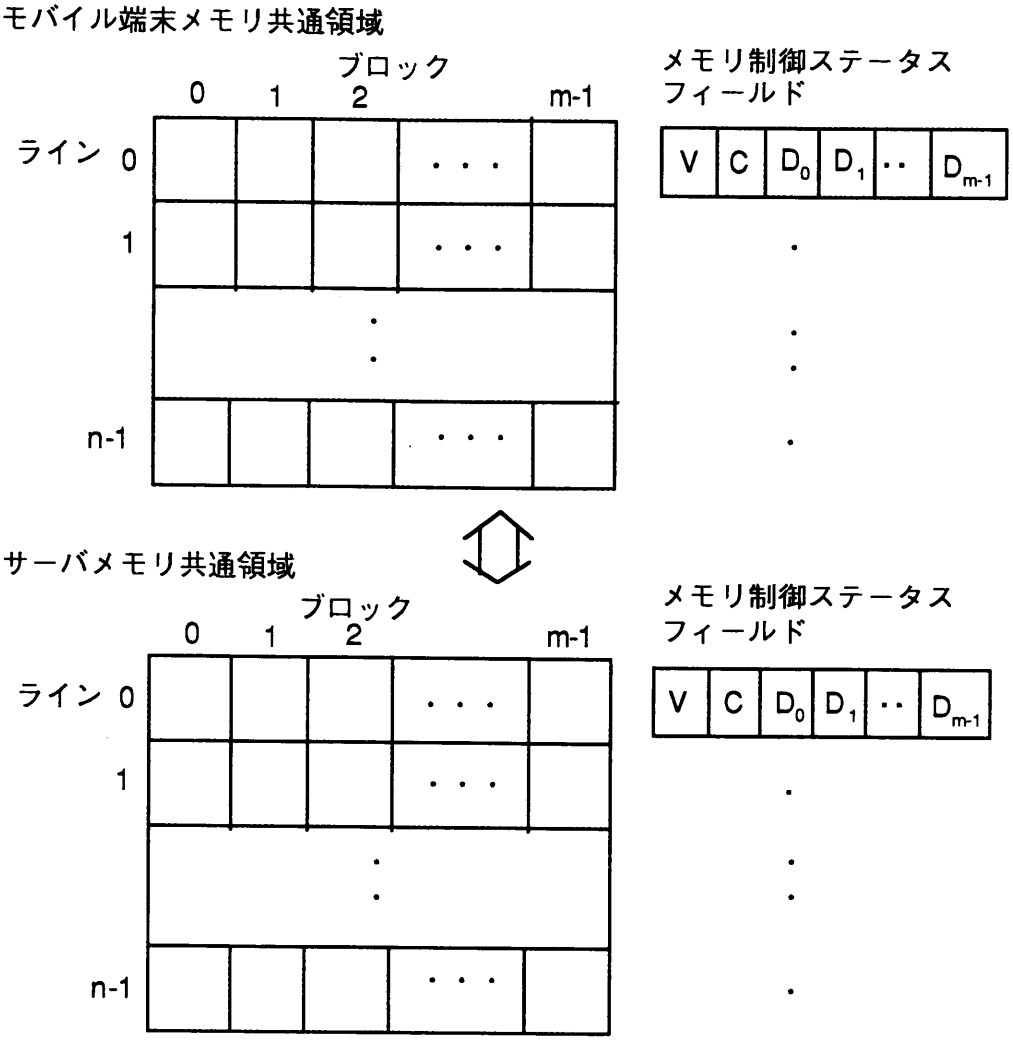


図 4-2 メモリのマッピングとメモリ制御ステータスフィールド

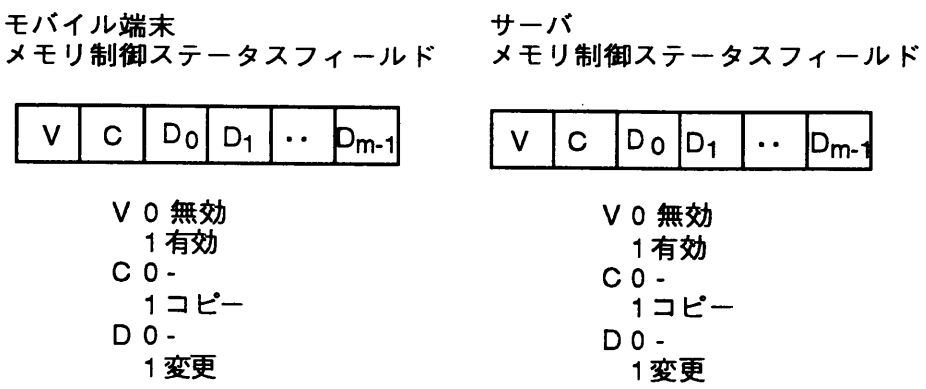


図 4-3 メモリ制御ステータスフィールド

チャを持つコンピュータに対し、比較的簡単なハードウェアの追加により、性能にほとんど影響を与えることなく実現可能である。

MMMは、モバイル端末のメモリ空間の一部とサーバのメモリ空間の一部とを共通メモリとしてメモリを共有する方式であり、図4-1に基本MMMの概念図を示す。図においてモバイル端末1メモリは、モバイル端末のメモリの全体を示し、その中の共通メモリで示す領域が、サーバメモリの中の共通メモリの領域と同一内容となるように制御される。モバイル端末2、モバイル端末3も同様であるが、サーバメモリ上ではそれぞれの共通メモリ領域は、別のメモリ領域に割り当てられる。基本MMMでは、モバイル端末とサーバの共通メモリ領域を1:1の対向で制御を行う。これによりモバイル端末から、サーバのメモリ空間の一部が自メモリ空間としてアクセス可能となり、またサーバからも、端末のメモリ空間の一部が自メモリ空間としてアクセス可能となる.

ソフトウェアによる分散共有メモリ方式では、共有メモリの制御のため、仮想メモリ方式の機構を利用している例がある [39]。MMMでは、共有メモリの制御に専用の付加ハードウェアとソフトウェアにより実現する。仮想メモリ機構は、サーバ側の共通メモリ領域を二次記憶と見なし、モバイル端末側のメモリ空間の拡大に利用する。これについては4.2.5で述べる。

基本MMMは、メモリ内容の一貫性管理方式の相違により、以降で述べる基本MMM管理方式1、基本MMM管理方式2、基本MMM管理方式3、の3種のメモリ管理方式から成る。

4.2.2 基本MMM管理方式1

(1) メモリラインとメモリ制御ステータスフィールド

サーバとモバイル端末のメモリの共通領域を、ラインと呼ぶ固定長の領域に分割し、各ラインに対応したメモリ制御ステータスフィールドをエントリに持つ、メモリ制御ステータスメモリを設ける。モバイル端末とサーバの同一番号のラインを1対1に対応させ、そのステータスをメモリ制御ステータスフィールドにより管理する。図4-2にその対応を示す。ラインをさらにブロックに分割し、メモリの変更をブロックの単位で管理

表4-1 基本MMM管理方式1の
メモリラインステータス

VCD	ラインステータス
100	最新，内容は変更していない(相手側と一致)、アクセス権
101	最新，内容を変更した(相手側と不一致)、アクセス権
11-	最新でない可能性がある(相手側が最新)、アクセス権なし
0--	無効，初期状態

注 Dは複数ビットある場合各ビットの論理和をとったものである

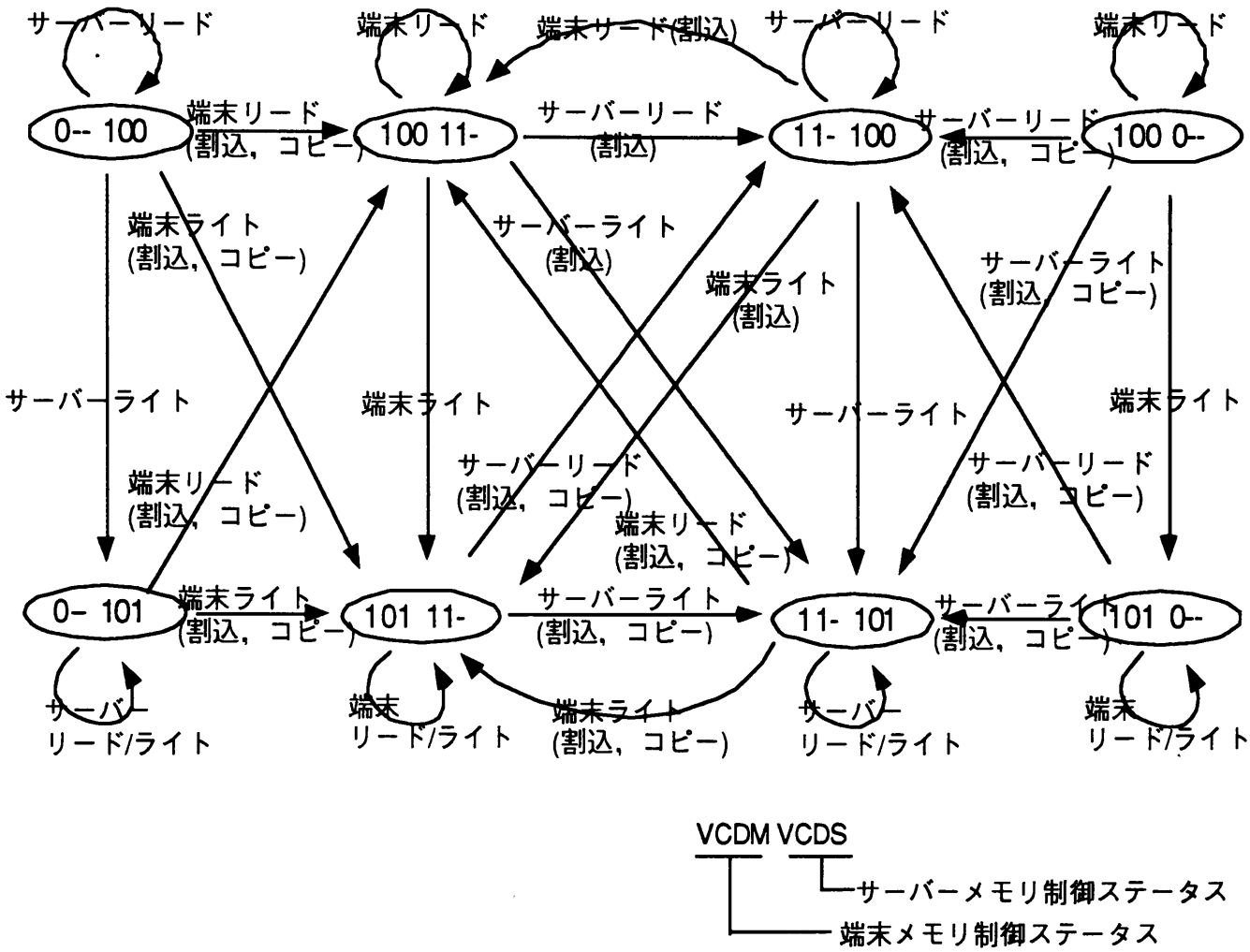


図4-4 基本MMM管理方式1のメモリ制御ステータス遷移図

可能にする。ラインの大きさは16-4096バイト程度、ブロックの大きさはラインを2のべき乗を単位に分割し、8-1024バイト程度を想定する。ブロックは一貫性制御のためのラインの書き戻し時のデータ量を減らすことが目的であり、メモリステータスの管理の基本はライン単位である。ラインのサイズは、通信路の幅、代表的アプリケーションに対応して最適に選択可能である。なおラインサイズ、ブロックサイズは、モバイル端末とサーバとでそれぞれ同一でなければならず、その切り替えをおこなう場合には、共通メモリを使用するアプリケーションの実行開始前に、管理プログラムにより行われねばならない。アプリケーション自身はそれらを意識する必要はない。なお共通領域はモバイル端末とサーバとで同一の容量でなければならない。

(2) メモリ制御ステータスフィールドとラインのステータス

メモリ制御ステータスフィールドはVビット、Cビット、および複数のDビットにより構成される(図4-3参照)。メモリラインのステータスは、V、C、Dの組み合わせによって示され、それを表4-1に示す。Vは有効ビットでラインの有効無効を示し、Cはコピービットで有効ビット(V)が1の場合に有効であり、0は最新のデータとアクセス権が自身にあることを示し、1は最新のデータとアクセス権が相手側にあり、自身は“最新でない可能性がある”ことを示す。メモリアクセスが許されるのは“最新(VC=10)”の場合だけであり、他のステータスでのメモリアクセスでは、メモリ例外の割込が発生する。Dは変更ビットでラインのステータスが“最新”の場合にのみ有効であり、ラインの中のブロック数分のビットがあり、ブロック単位にデータの変更がおこなわれたかどうかを示す。メモリ制御ステータスフィールドは、モバイル端末とサーバとで同一の意味を持ち、全く対称である。

メモリ制御ステータスの遷移を図4-4に示す。図において各楕円内はモバイル端末とサーバのラインステータスを組み合わせたものを示す。左側の2つのステートは、初期状態においてサーバ側のラインが有効で、モバイル端末側のラインが無効であることを示し、右側の2つのステートは、初期状態においてモバイル端末側のラインが有効でサーバ側のラインが無効であることを示す。中央の4つのステートは、無効のラインから有効のラインに転送がおこなわれた後の定常状態を示す。初期状態及び終了時の状態が、

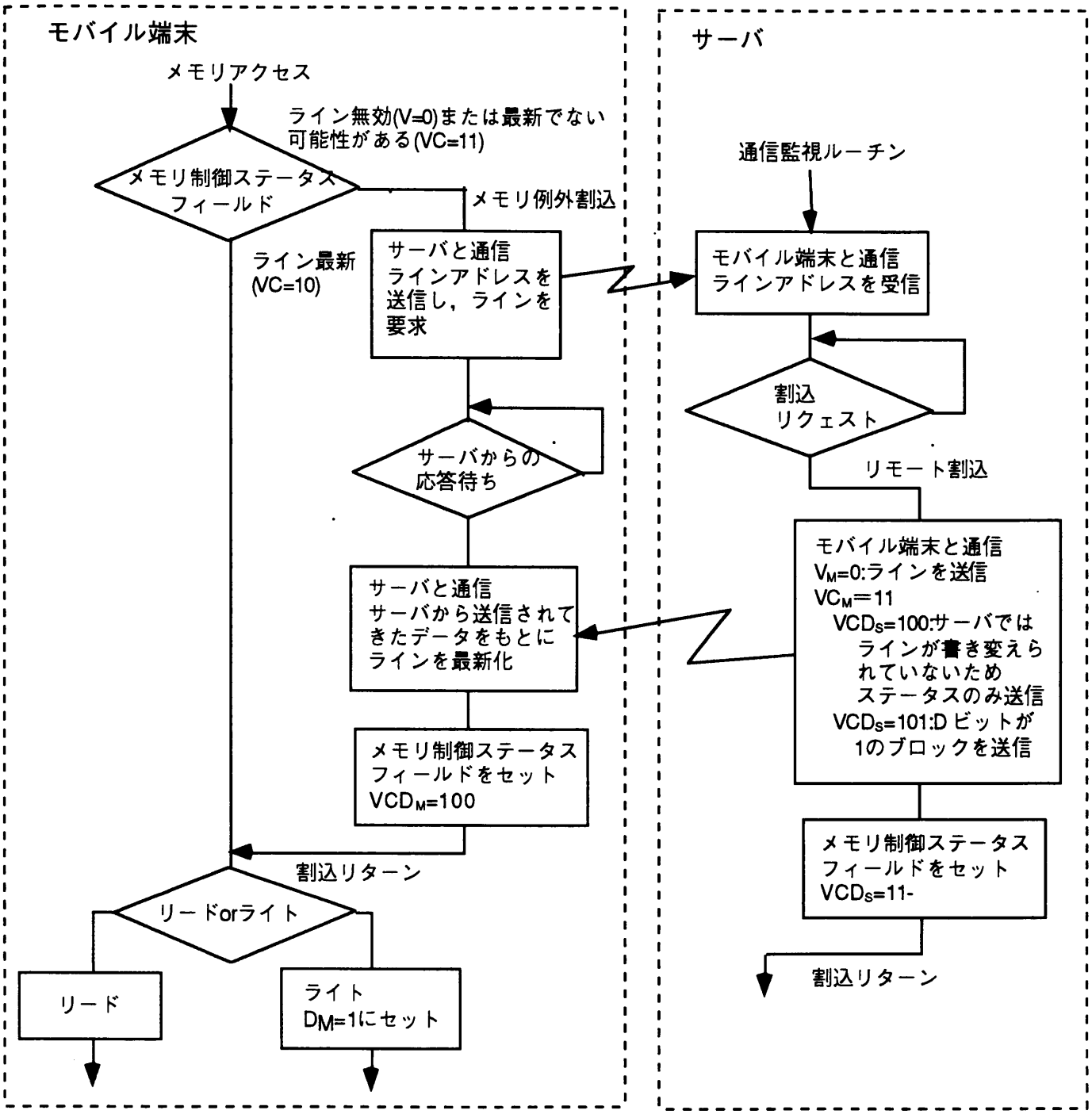


図 4-5 メモリライン転送制御

最新のデータがサーバ側にあるという使い方の場合には、初期状態においては、図4-4の左上、すなわちサーバ側は“最新”、モバイル端末側は“無効”のステータスに設定される。もしこれが逆の使用方法、すなわち最新のデータがモバイル端末側にあるという使用方法であれば、初期状態のステータスは上記とは逆の状態に設定される。初期状態の設定はモバイル端末とサーバとが連携し、アプリケーションの開始直前に共通メモリ管理プログラムにより行われる。メモリ制御ステータスの変更は、ストア命令実行時にハードウェアによりDビットを1にセットする以外は、モバイル端末とサーバの共通メモリ管理プログラムが連携することにより、ソフトウェア制御で行われる。

(3) メモリアクセスとラインの転送制御

モバイル端末でのメモリアクセス時のメモリラインの転送制御を図4-5に示す。モバイル端末でメモリアクセスが発生すると、まずラインのステータスが調べられる。ラインステータスが“最新”の場合には、メモリの内容は有効であり正常にリードまたはライトがおこなわれ、さらにライトの場合には対応するブロックのDビットが1にセットされる。ラインステータスが“無効”、または“最新でない可能性がある”の場合にはメモリ例外の割り込みが発生する。以上まではすべてハードウェア制御によって行われる。メモリ例外割込処理の中では, サーバと通信を行ってサーバ側にリモート割込を発生させる。サーバ側のリモート割込の中の処理では、モバイル端末のラインステータスが“無効”の場合には、まだライン単位の転送が行われていないためサーバのラインを転送し、“最新でない可能性がある”の場合にはすでにライン転送が行われているため、サーバのステータスのDビットを調べ、1がセットされているブロックのみを転送する、同時にサーバのステータスを“最新でない可能性がある”にセットし、リモート割込を終了させる。モバイル端末側ではステータスを“最新”にセットし、メモリ例外割込処理を終了させ、割り込まれた命令を再実行する。以上はモバイル端末でのメモリアクセス動作であるが、サーバでのメモリアクセス時のメモリライン転送制御も同様である。

(4) ラインの競合制御

モバイル端末とサーバとが同一ラインを同時に使用(参照、または更新)した場合、タ

イミングによってはラインの取り合いとなり、デッドロックが発生する可能性がある。これはラインのアクセス権がない側でメモリアクセスによる割込が発生し、ラインのアクセス権を得て割込処理から戻った直後に、他方の側で同一ラインに対するメモリアクセスが発生し、ラインのアクセス権がなくなっているため割込が発生するという、両方の側で同一ラインに対するアクセス権取得のための割込が交互に発生し、命令が進まなくなるタイミングが発生する場合である。この防止のため、メモリ例外割込が発生してから、割込終了直後の最初の命令の実行完了までの間、リモート割込を禁止する。これによりモバイル端末とサーバとが同一ラインを使用した場合でも、1命令ずつ交互に実行可能となる。

(5) 通信が接続不可の場合の制御

モバイルコンピューティング環境下での通信は、一般的には無線通信のため、通信が開始できなかったり、途中で切断されたりする場合がある。モバイル端末とサーバとが連携したアプリケーションを実行中に、片方の側でメモリ例外の割込が発生し、割込処理の中で相手側との通信中で何らかの通信障害が発生し、通信リトライを行っても正しく通信ができない場合がある。割込処理の中で相手との通信に失敗した場合には、それまで割込処理の中で実行した中間結果をすべて無効化し、割込発生の直前の状態に戻し、通信接続不可をアプリケーションに対して通知するとともに、メモリ制御ステータスフィールドを調べ、現在のメモリステータスの内容を通知する。アプリケーションはこの通知を受け、通信ができない状態で実行を継続するのか、通信が可能となるまで実行を中断して待機するのかを判断する。具体的には、アプリケーションによりメモリの内容が最新でなくても実行可能の場合には、その旨を端末ユーザに通知した上で実行を継続し、実行不可の場合には、実行を中断し端末ユーザに次の動作の指示を待つ。メモリステータスがVC=11(“最新でない可能性がある”)の場合には、ラインの内容が一度最新になり、その後アクセス権が相手側に移り、その一部が変更されている可能性があることを示し、アプリケーションによっては、条件付きでデータを利用することが可能である。例えばスケジュール管理のアプリケーションの場合、メモリステータスがVC=11のラインは他方の側がそのラインをリードしており、場合によってはその一部を変更して

いるかもしれないということを示すが、そのことを認識した上でデータを読むことができる方が利用性を増す。

なお通信に失敗した場合でも、割込の無効化処理によりメモリ制御ステータスを正しく管理することができるため、メモリの一貫性に矛盾が発生することはない。

4.2.3 基本MMM管理方式2及び基本MMM管理方式3

4.2.2で述べた基本MMM管理方式1では、ラインはモバイル端末とサーバとで排他的に使用され、アクセス権は一方のみに与えられる。リードオンリーの利用であってもラインのアクセス権を得てから使用しなければならない。これに対して、リードオンリーの場合には、モバイル端末とサーバとで同時にメモリを参照することを許せば、並列度を上げることができる。さらに書込が発生したときのメモリ制御方式において、一貫性制御を緩和することにより、さらに並列度が増す。ただし方式によっては、内容の一貫性に関し、アプリケーションでも意識する必要が出てくるため、アプリケーションの構築に注意が必要になる。基本MMMでは、基本MMM管理方式1に加えて、次の2種類のメモリ管理方式が選択可能である。なお以降の説明で、ライト権は読み出しと書き込みが可能、リード権は読み出しのみが可能、アクセス権は書き込みまたは読み出しが可能、すなわちライト権またはリード権があることを意味するものとする。MMMでは、ライト権またはリード権の取得はそれが必要になった時点で、すなわちライト命令またはリード命令の実行時に自動的に取得されるため、アプリケーションが意識する必要はない。これは命令の実行時に必要なアクセス権限がない場合には割込を発生させ、他方の側と通信を行うことで権限を取得してから必要なアクセスが行われる。

(1) 基本MMM管理方式2

一方の側のラインがライト権のとき、他方にはリード権が与えられる。無効のラインにリードまたはライトが発生した場合には、割込を発生させ他方の対応するラインを転送して自ラインを最新化し、リードの場合には他方のラインのDビットをすべて0にリセットし、自ラインをリード権にセットすると共にDビットをすべてリセットしてリードを行い、ライトの場合には他方のラインをリード権にセットすると共にDビットをす

表 4-2 基本 MMM 管理方式 2 のメモリラインステータス

VCD	ラインステータス
100	最新、内容は変更していない(相手側と一致)、ライト権
101	最新、内容を変更した(相手側と一部不一致)、ライト権
110	最新、相手側と一致、リード権
111	最新ではない、相手側と一部不一致、リード権
0--	無効，初期状態

注 Dは複数ビットある場合各ビットの論理和をとったものである

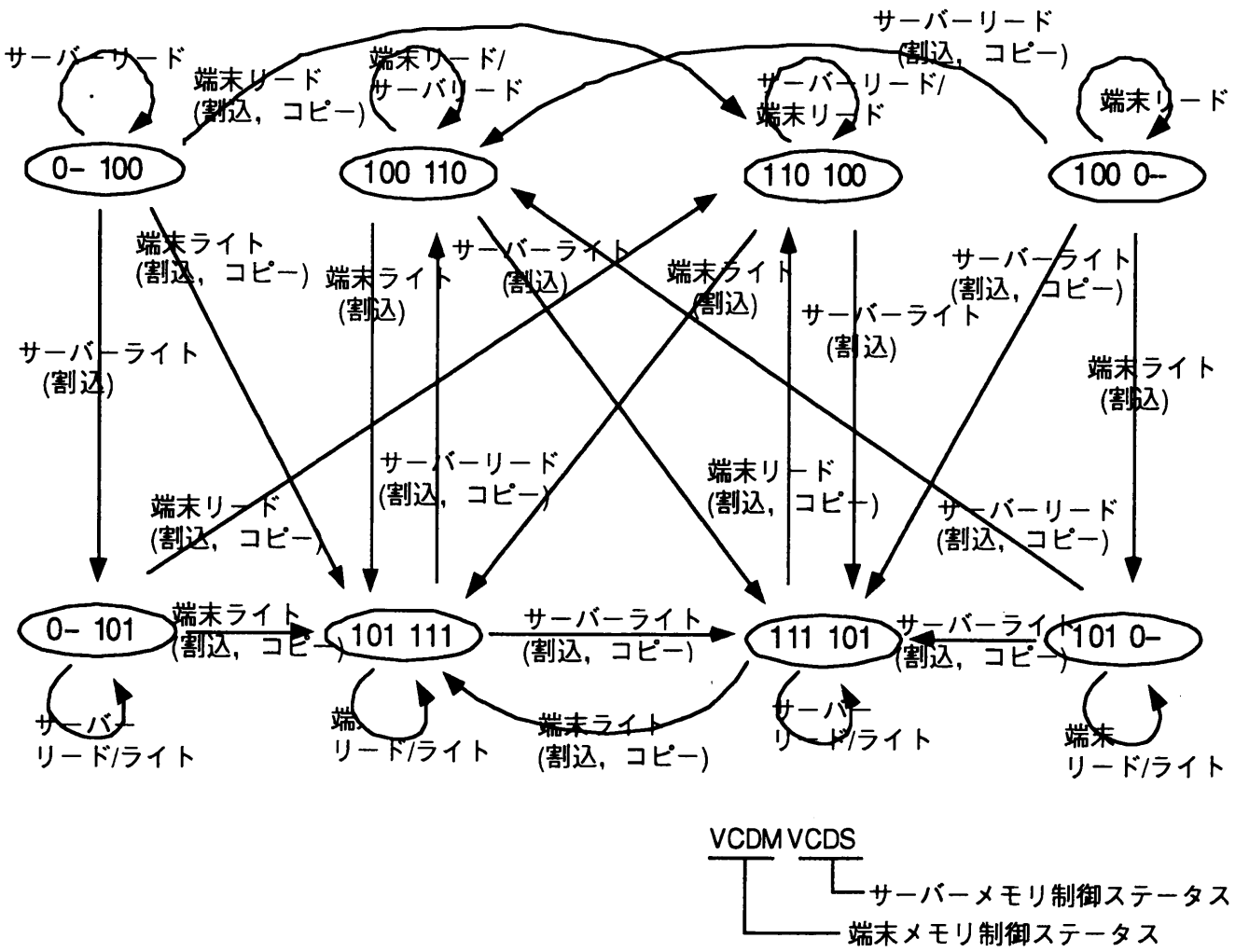


図 4-6 基本 MMM 管理方式 2 のメモリ制御ステータス遷移図

べて0にリセットし、自ラインをライト権にセットすると共にDビットをすべて0にリセットし、他方のラインの対応するDビットを1にセットし、自ラインの対応するDビットを1にセットしてライトを行う。ライト権のある側でライトが発生し、Dビットが0の場合には、割込を発生させ他方と通信をおこない、他方のDビットを1にセットした後、自身のDビットを1にセットしてライトを行い、Dビットがすでに1にセットされている場合にはそのままライトをおこなう。これはライトを行う場合、他方に対してデータの更新はおこなわず、制御情報のみを送り、一度Dビットを1にセットすると、以降の通信は行わないように制御することで通信のトラフィックを減らす。ライト権のある側でのリードは、内容が最新のためそのままリードを行う。リード権がある側でリードが発生し、かつDビットが1にセットされている場合には、相手側でライトが行われ最新でなくなっているため、割込を発生させて通信をおこない，Dビットが1にセットされているブロックを他方から転送してラインを最新化し、他方と自身のDビットをすべて0にリセットしてからリードをおこない、変更ビットが0の場合には内容が最新のためそのままリードする。リード権のある側でライトが発生した場合には，割込を発生させて通信をおこない，自身のDビットがどれか1にセットされている場合には対応するブロックを他方から転送してラインの最新化をおこない、他方と自身のDビットをいったんすべて0にリセットしてライト権を取得し、ライトするブロックに対応する他方と自身のDビットを1にセットしてライトをおこない、自身のDビットがすべて0の場合には内容が最新のためデータ転送をおこなわずにライト権を入手し、ライトするブロックに対応する他方と自身のDビットを1にセットしてライトをおこなう。基本MMM管理方式2では、DビットはCビットが1でも意味を持ち、リード権のある側でDビットが0は、対応するブロックの内容が最新(ライト権のある相手側と一致)であり、Dビットが1は、内容が最新でない(相手側で書き替えている)ことを示す。基本MMM管理方式2についてのメモリラインステータスを表4-2に、メモリ制御ステータスの遷移図を図4-6に示す。

(2) 基本MMM管理方式3

一方の側のラインがライト権のとき、他方はリード権が与えられる。無効のラインに

表 4-3 基本 MMM 管理方式 3 のメモリラインステータス

VCD	ラインステータス
100	最新、内容は変更していない(相手側と一致)、ライト権
101	最新、内容を変更した(相手側と不一致)、ライト権
11-	最新でない可能性がある、リード権
0--	無効，初期状態

注 Dは複数ビットある場合論理和をとったものである

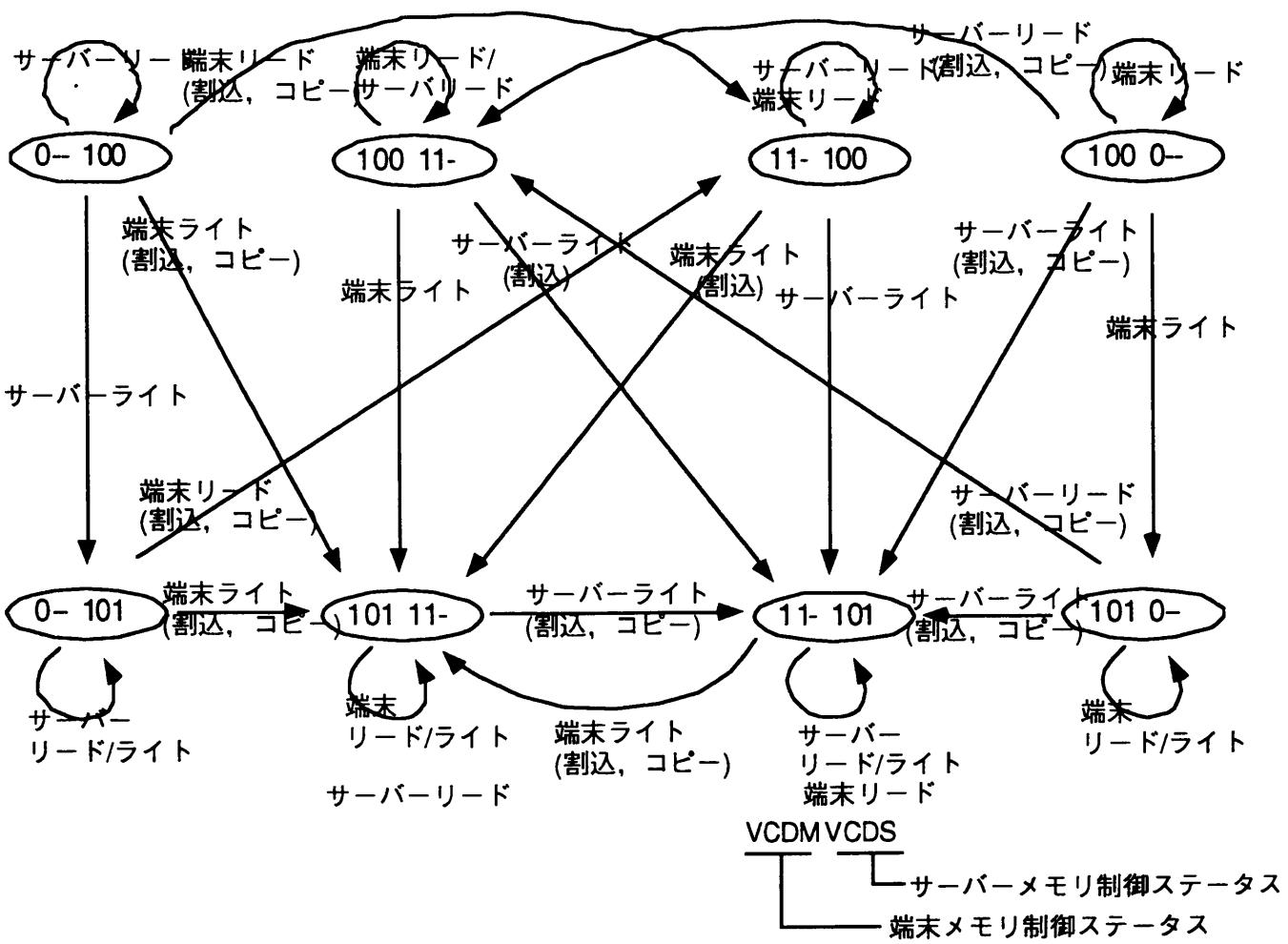


図 4-7 基本 MMM 管理方式 3 のメモリ制御ステータス遷移図

リードまたはライトが発生した場合の制御は、ライトの時に他方のラインのDビットを変更しないことを除き、基本MMM管理方式2と同様である。ライト権のある側でライトが発生したときは、自身のラインのDビットのみ1にセットし、他方のラインのDビットは変更しない。リード権のある側でリードが発生したとき、特にプログラムからの指定がない場合には、最新化せずにリードする。リード権のある側でライトが発生したときには、割込を発生させ他方と通信をおこない，他方の側のDビットが1にセットされているブロックを転送してラインの最新化を行い、自身のDビットをいったんすべて0にリセットしてライト権を入手し、その後ライトをおこない自身の対応するブロックのDビットを1にセットし、他方のDビットは変更しない。リード時に最新の内容をリードするためには、同期開始命令を実行し同期モードにしてからリードをおこなう。同期開始命令が実行されると、割込を発生させて他方の側と通信をおこない、ライト権のあるラインのDビットと対応するリード権のあるラインのDビットを一致させ、これをすべてのラインについておこない、その後同期モードとなる。同期モードになると、基本MMM管理方式2と同じ制御をおこなう。同期モードの解除は同期終了命令によりおこなう。基本MMM管理方式3では、リード権のある側では内容は必ずしも最新ではなく、内容が変更されているかどうかはライト権のある側のDビットで示され、この情報は同期開始命令の実行によりリード権のある側に通知される。基本MMM管理方式3についてのメモリラインステータスを表4-3に、メモリ制御ステータスの遷移図を図4-7に示す。

4.2.4 メモリの一貫性制御

分散共有メモリ方式では、メモリ内容の一貫性制御が最大の課題である。共有メモリ方式のマルチプロセッサでは、ほぼシーケンシャル計算機としてのメモリ内容の一貫性制御が可能であるが、分散共有メモリ方式では同様の一貫性制御を行うと、オーバーヘッドのため性能に大きな影響が出る。このため分散共有メモリ方式では、プログラミングに制約を与え、メモリの一貫性を緩和することで性能の向上を図っており、各種一貫性制御方式が提案されている [53-59]。基本MMMには、4.2.2及び4.2.3で述べたように基本MMM管理方式1、基本MMM管理方式2及び基本MMM管理方式3の3種のメ

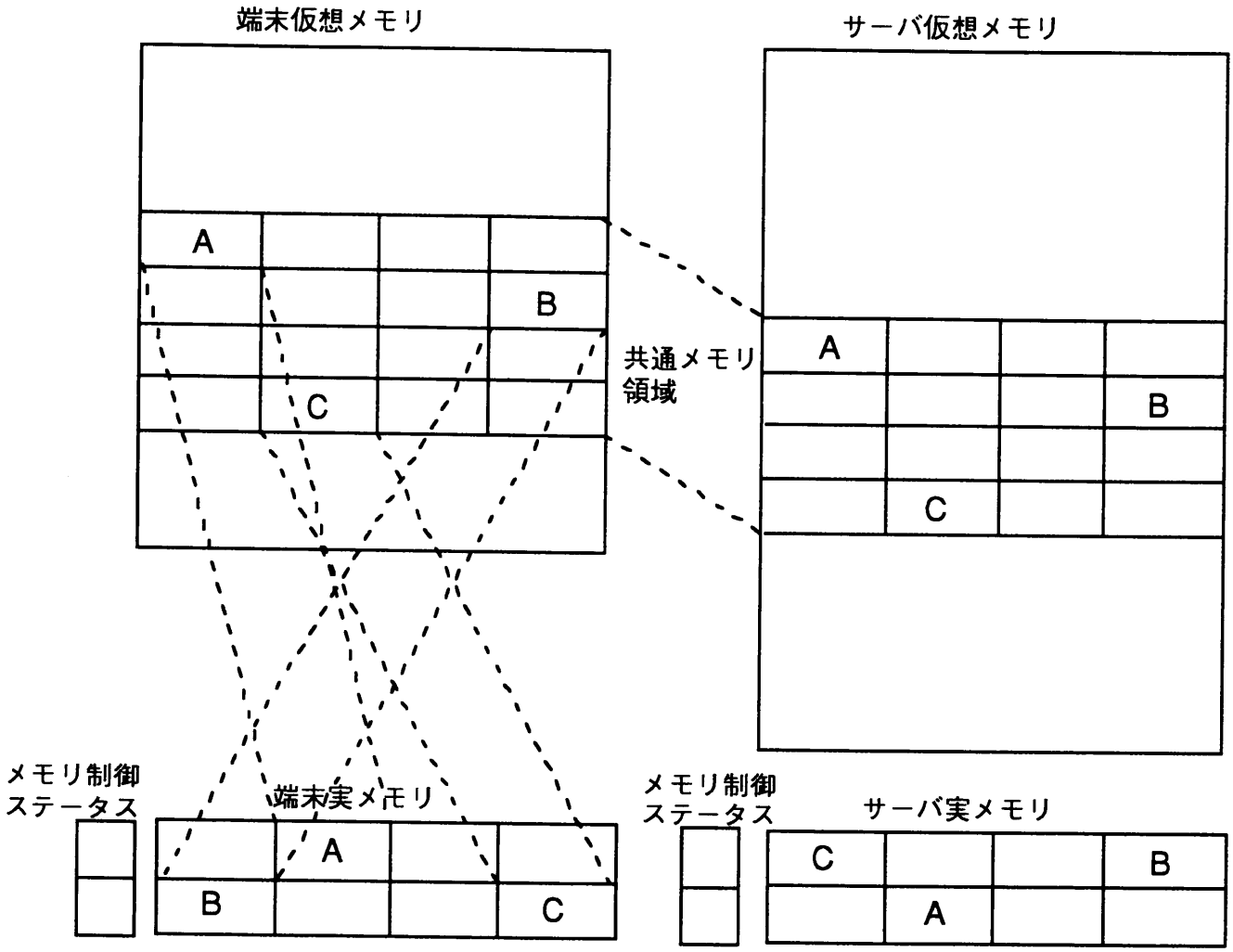


図 4-8 仮想メモリ方式による端末のアドレス空間の拡大

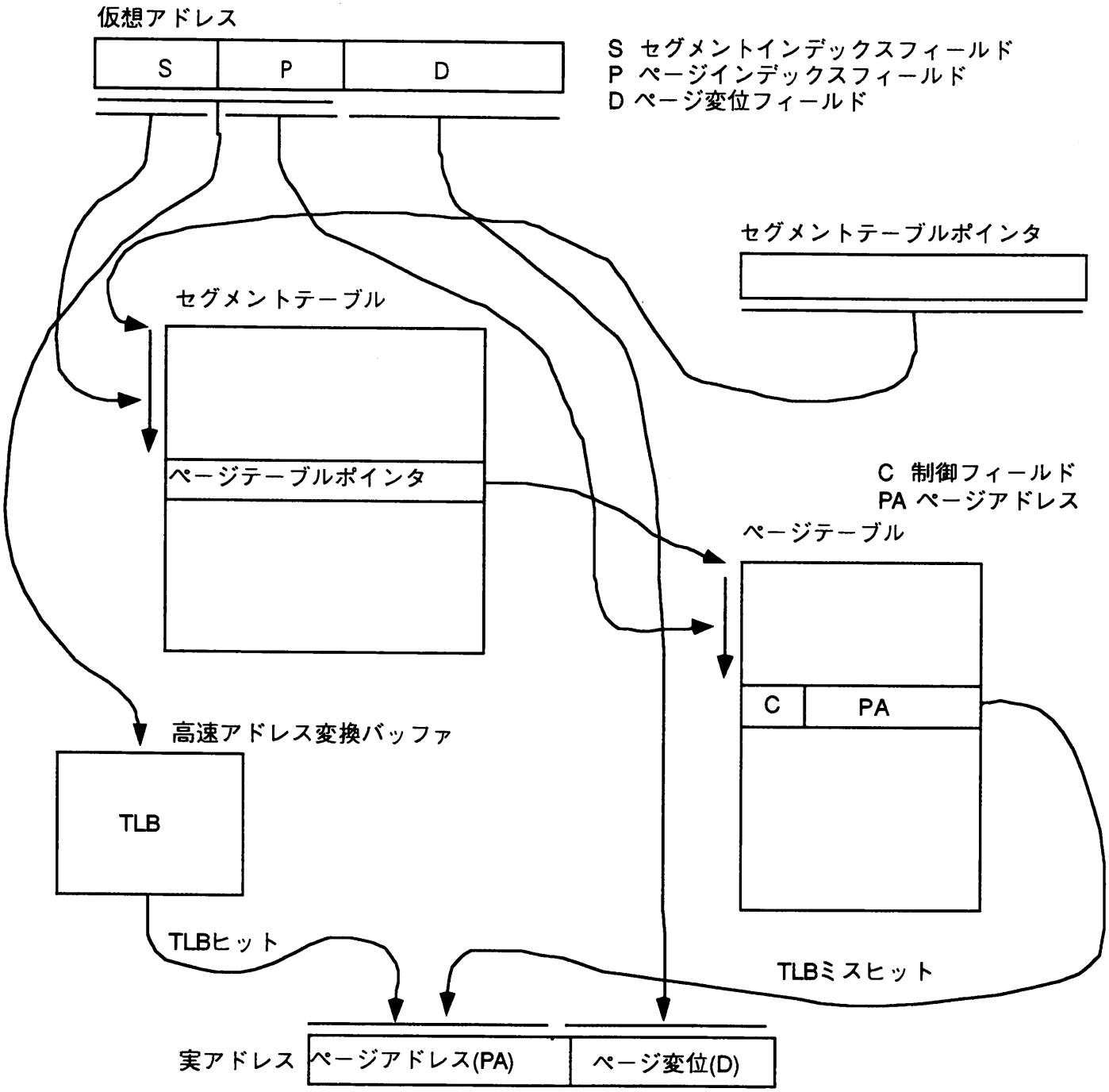


図4-9 仮想アドレスのアドレス変換

モリ管理方式があり、異なる一貫性管理方式を持つ。基本MMM管理方式1と基本MMM管理方式2がシーケンシャル型、基本MMM管理方式3が一種の遅延リリース型[57]の一貫性制御方式に分類することができる。シーケンシャル型は命令の実行の順序がプログラムで記述された順番どおりで、自分自身だけでなく他のプロセッサからの観測でも順番が守られる方式である。しかしながら命令の実行時間が有限であり、さらにメモリの速度がプロセッサより一般的に遅いこと、各プロセッサが性能向上のための各種の並列処理をおこなうため、シーケンシャル制御を厳密に守ることは必ずしも簡単ではない。一般的にはシーケンシャル型であっても若干のプログラムへの制約があることが多い。積極的にシーケンシャル制御を緩めることで性能向上を図った方式が緩和型メモリ一貫性制御方式と呼ばれる[30,53]。遅延リリース型一貫性制御方式は、緩和型メモリ一貫性制御方式の一つであり、一つのプロセッサの共有メモリの排他使用開始時に、まず共有メモリの内容の他との一致をとり、その後メモリをローカルに変更，排他使用の終了時には変更を他に通知せず、最新の内容が必要になった時点でメモリ内容の一致を取ることで、一貫性制御のための通信の軽減を図った方式である。

4.2.5 メモリ空間の拡大

端末の共通メモリ領域を仮想メモリ方式で管理することにより、端末側のメモリ空間の拡大が可能となる。図4-8に仮想メモリ方式によるアドレス空間の拡大の図を示す。図において端末の共通メモリ領域は、仮想メモリであり、実際のメモリの内容は、サーバ側の共通メモリ領域に置かれる。サーバの共通メモリ領域は仮想メモリであっても実メモリであってもよいが図の例では仮想メモリで示す。サーバの共通領域が仮想メモリであれば、実際のメモリの内容はさらにサーバの外部記憶装置に置かれる。端末の共通メモリ領域の仮想メモリに対しては端末の実メモリを対応させ、当面のプログラムの実行に必要な分を実メモリに置く。仮想メモリと実メモリはページと呼ばれる一定の単位に分割され、仮想メモリの管理はページ単位におこなわれる。MMMのメモリ制御ステータスメモリは、共通メモリ領域に対応した実メモリに対して設けられる。仮想メモリのページサイズとMMMのラインサイズが同じでない場合にはそれぞれが整数倍の関係となるようにサイズを選択しなければならない。

図4-9に仮想アドレスのアドレス変換過程の例を示す。図ではメモリ上に置かれるアドレス変換テーブルである、セグメントテーブルとページテーブルを使用する2レベルのアドレス変換の例を示す。アドレス変換は、まず仮想アドレスが与えられると、セグメントテーブルポインタで示されるセグメントテーブルから、仮想アドレスのセグメントインデックスフィールド（S）をインデックスとしてページテーブルポインタを求める。次に、このページテーブルポインタで示されるページテーブルから、仮想アドレスのページインデックスフィールド(P)をインデックスとしてページアドレスを求める。このページアドレスに仮想アドレスのページ変位部（D）を結合したものが実アドレスとなる。以上の2レベルのアドレス変換では、仮想アドレスから実アドレスに変換するため2回のメモリを参照する必要があり、オーバーヘッドが大きい。これを高速化するため高速アドレス変換バッファであるTLBを設け、1度アドレス変換がおこなわれると、変換結果をこのTLBに格納し、以降同じアドレスが参照された場合には、TLBから参照することにより、アドレス変換を高速化におこなう。

つぎに仮想アドレスを用いたMMMのメモリ制御を、端末でのメモリアクセスを例にして説明する。仮想メモリに対応する実メモリは共通メモリ領域に割り付ける。なおページを仮想アドレスのメモリ管理単位、ラインをMMMのメモリ管理単位とする。なお基本MMM管理方式1の例で説明するが他の管理方式でも基本的には同様である。

まず仮想アドレスが与えられるとアドレス変換テーブルを用いたアドレス変換が行われる。ページテーブルを参照したときに、ページが無効であるとページ例外の割込を発生させ、実メモリのページの割り付けを行い、対応するラインのメモリ制御ステータスフィールドの初期化(VCD=0--)を行うと同時にサーバと通信を行い、仮想ページアドレスと実ページアドレスを送る。サーバ側では外部記憶から仮想ページを読み出し、共通メモリ領域の対応するページに書き込みメモリ制御ステータスフィールドの初期化(VCD=100)を行う。端末側では割込から戻り、変換された実メモリアドレスによりメモリのアクセスを行うが、ラインのステータスが無効のため割込を発生させ、サーバと通信を行い、対応するラインを端末側に書き移し、サーバと端末のメモリ制御ステータスをそれぞれ、VCD_S=11-、VCD_M=100にセットし、端末側では割込から戻りメモリアクセスを行う。以上の説明では、わかりやすくするためページ例外の割込から一旦戻り、

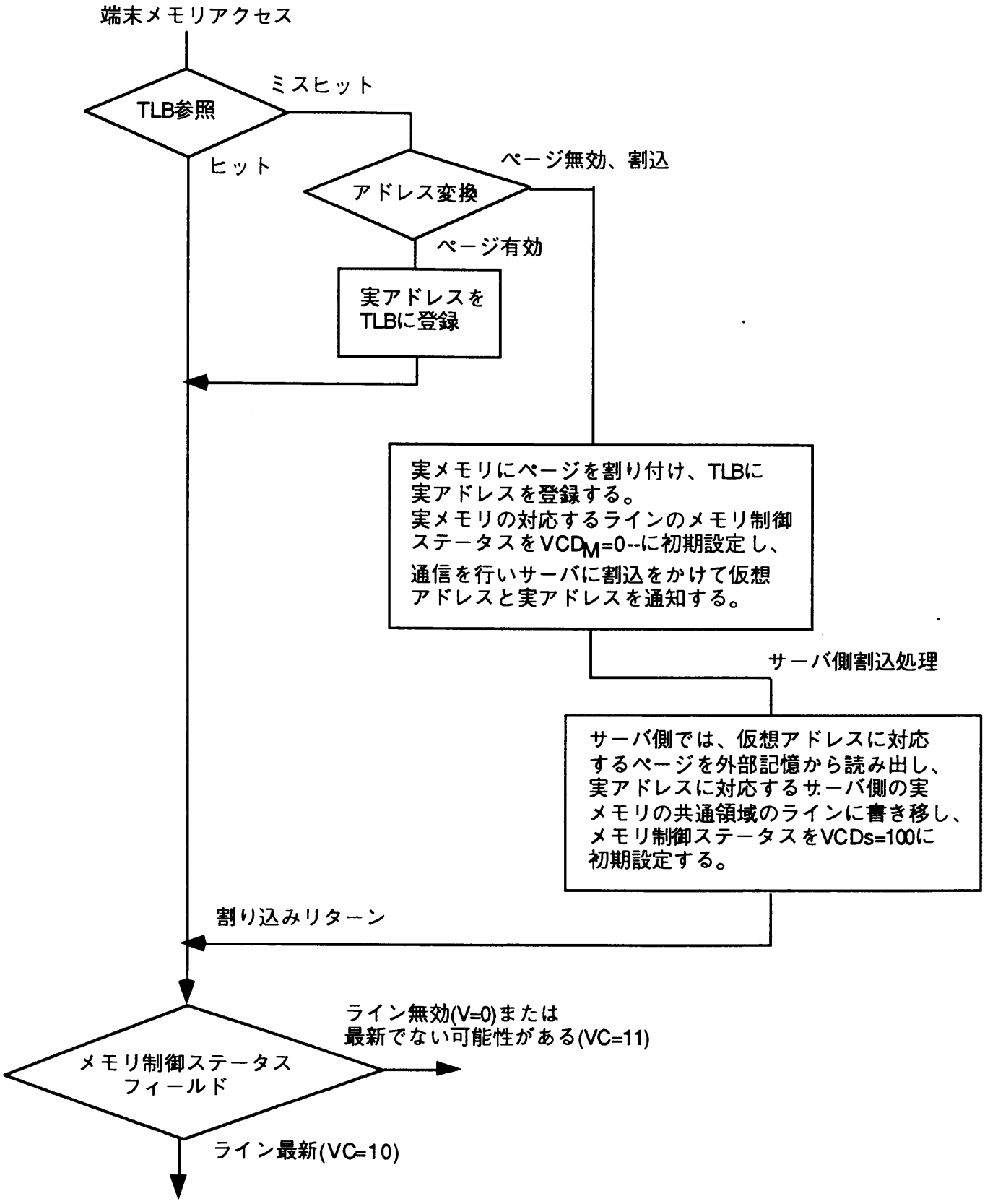


図4-10 仮想アドレスによる端末メモリアクセス

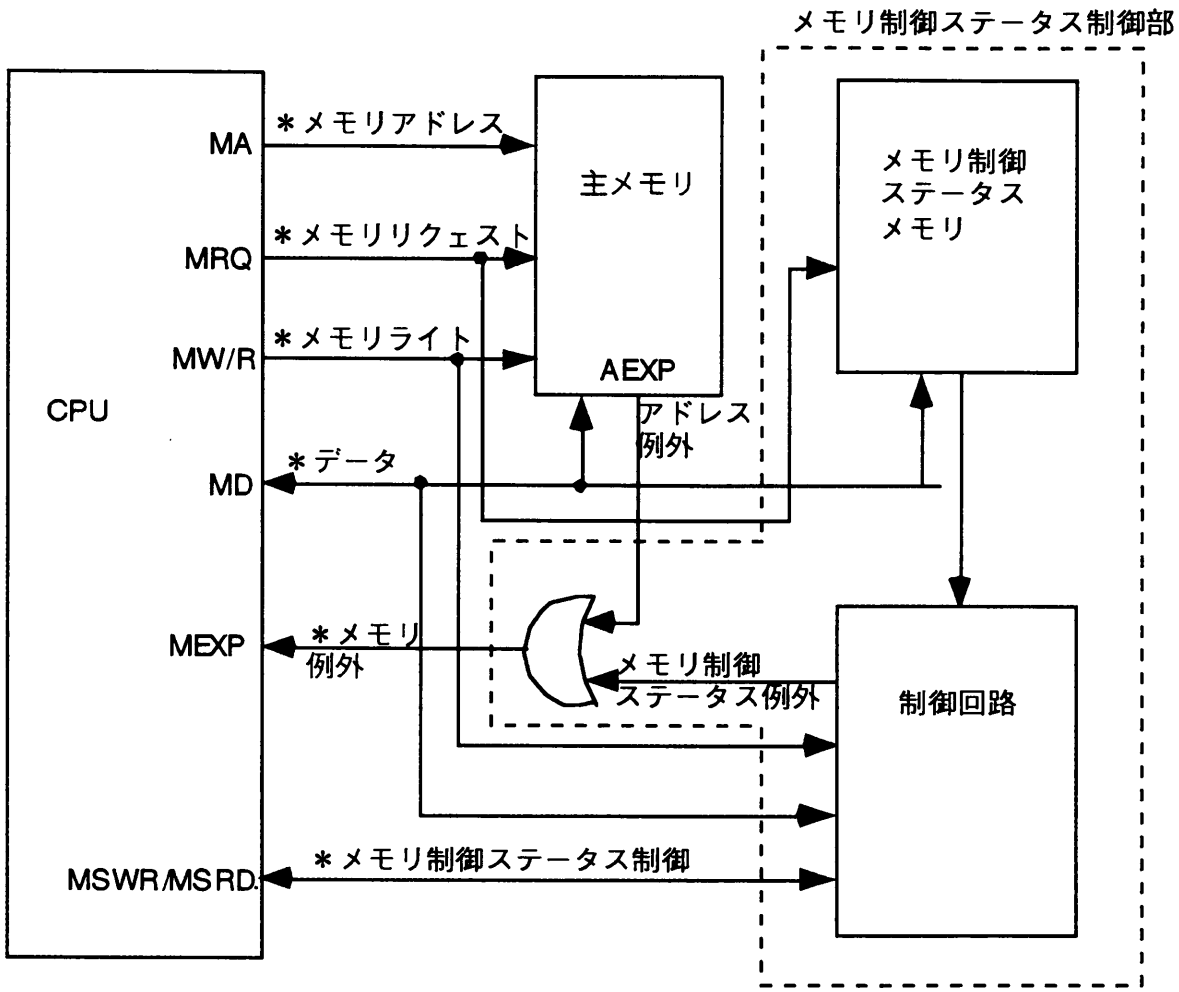


図 4-11 MMM のハードウェア構成

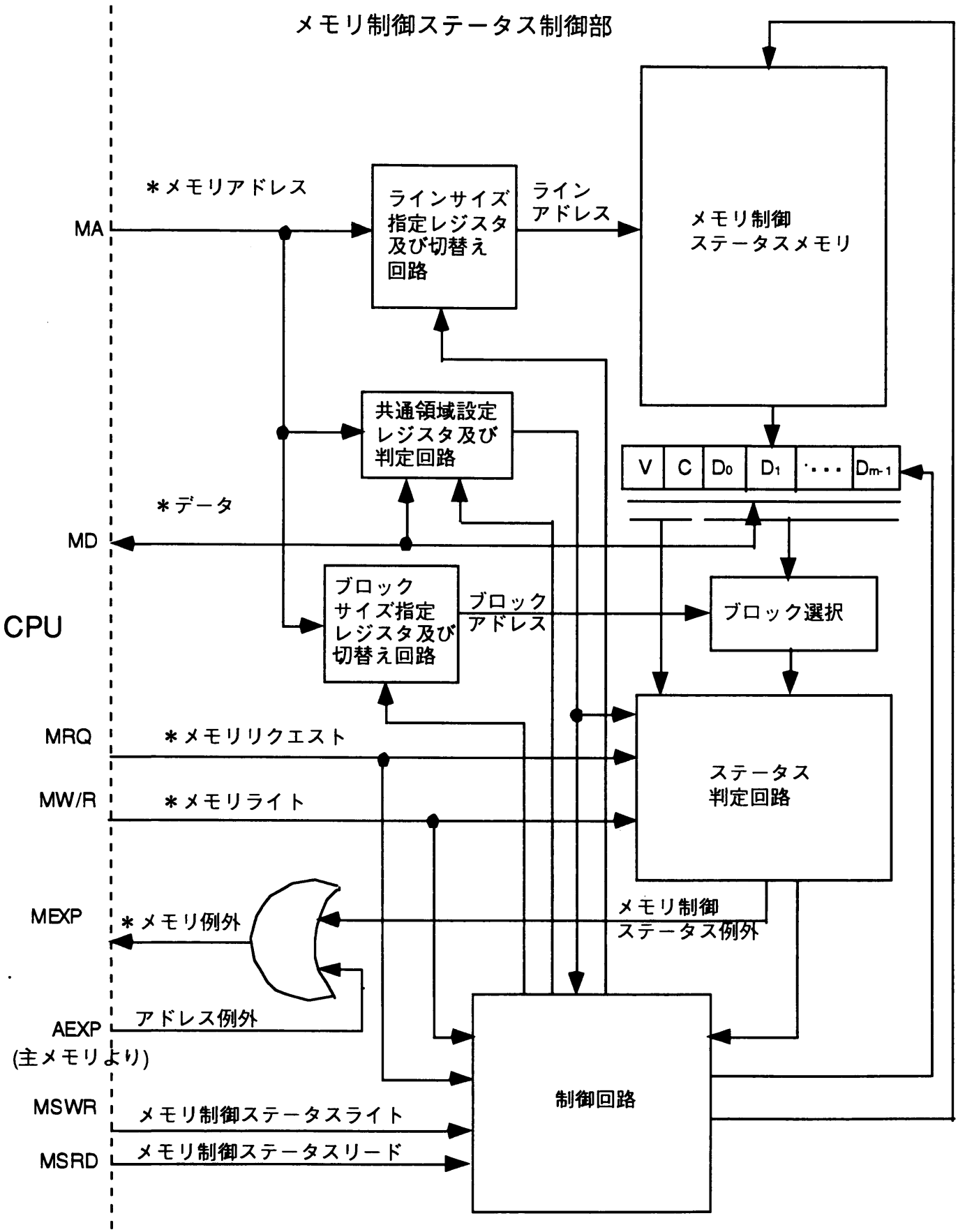


図 4-12 メモリ制御ステータス制御部

MMMの例外が発生するとしたが、実際は割込のオーバーヘッドを減らすためページ例外の割込処理の後続けてMMMの処理を行う。なおMMMのメモリ管理単位はラインのため、ページが複数のラインで構成される場合には、共通メモリ領域にページに対応する複数のラインの割り付けを行い、転送は参照が行われた仮想アドレスを含むラインのみ行い、ページ内の他のラインはメモリ制御ステータスの初期化のみを行い、実際のデータ転送は命令の実行で要求があった時に行われる。ラインが複数のページで構成され、ページ無効が発生して実ページを割り付ける場合には、ラインに含まれるすべてのページの割り付けを同時に行い、データ転送はライン単位で行う。

アドレス変換の結果、ページが有効の場合には、変換された実アドレスによりメモリ制御ステータスが参照され、その内容により必要な制御がおこなわれる。

以上はTLBがない場合であり、TLBがある場合には仮想アドレスが与えられると、まずTLBが参照される。1度アドレス変換が行われていれば、実ページアドレスは、TLBに登録されており、実アドレスはただちに求められる。TLBに登録されていなければ、アドレス変換テーブルを用いたアドレス変換がおこなわれる。図4-10に仮想アドレスによる端末メモリアクセスを示す。図において割込から戻った後のメモリステータスフィールドの参照以降の制御は図4-5にしたがう。サーバ側の仮想アドレスによるメモリアクセスも同様の動作となる。

ページ無効が発生し、実ページの割り付けが必要になった時に、実ページがすべて割り付けられており空きページがない場合には、最も使用されなかった(LRU方式)実ページを無効にすることにより、新しい仮想ページを割り付ける。実ページを無効にするのは、対応するラインのメモリ制御ステータスを調べ、内容が最新であり(VC=10)、かつ内容が変更されていれば(D=1)、変更されたブロックの内容を、モバイル端末からサーバに書き戻し，端末側のラインのステータスをV=0(無効)，サーバ側のステータスをVCD=100(最新)にセットし、メモリ制御ステータスの内容が他のステータスの場合にはデータの書き戻しはおこなわずメモリ制御ステータスのみの変更を行い、サーバ側では対応するページが変更を受けている場合には外部記憶に書き戻し、変更されていなければ書き戻さず、その後新しい仮想ページの割り付けをおこなう。

以上に示したように、MMMに仮想メモリ方式を採用することにより、共通メモリ領

域のアドレス空間の拡大が可能となる。また従来の仮想アドレス方式のアーキテクチャを変更することなく利用可能である。

4.3 MMMの構成

4.3.1 ハードウェア構成

図4-11にMMMのハードウェア構成のブロック図を示す。基本的には従来の標準的なアーキテクチャを持つCPUのハードウェアを変更することなく、メモリ制御ステータスメモリ用のRAMとその制御回路を付加することで実現可能である。図4-11において点線で囲まれたメモリ制御ステータス制御部が、MMMを構成するための付加回路であり、図4-12にさらにその詳細なブロック図を示す。図おいて、メモリ制御ステータスメモリは高速SRAMで構成され、共通メモリ領域の最大ライン数分のフィールド数の容量を持ち、1フィールドはV、Cの各1ビットと1ライン中の最大ブロック数分のビットを加えたもので構成される。メモリ制御ステータスメモリをアクセスするためのアドレスは、主メモリアドレスの上位ビットが使用される。共通領域設定レジスタ及び判定回路は、共通領域上限レジスタと下限レジスタを持ちメモリアドレスが共通領域かどうかを判定する。ラインサイズ指定レジスタ及び切替え回路は、ラインサイズの指定により、メモリアドレスの中でメモリ制御ステータスメモリのアドレスとして使用する範囲を切り替えるため制御を行なう。ブロックサイズ指定レジスタ及び切替え回路は、ブロックサイズの指定により、メモリアドレスからブロックを選択するための範囲を選択し、ブロック選択はメモリ制御ステータスメモリから同時に読み出された同一ライン内の複数のDビットよりブロックアドレスで指定されたDビットを選択する。ステータス判定回路は、メモリリクエスト、共通領域判定信号、メモリライト、Vビット、Cビットの組み合わせにより、メモリ制御ステータス例外を発生させる。この信号は主メモリ部で生成されるアドレス例外と論理和が取られCPUに入力される。制御回路はメモリライト時のDビットのセット、CPUからのメモリ制御ステータスライト命令によるV、C、Dビットの書き込み、メモリ制御ステータスリード命令のよるV、C、Dビットの読み出しを制御する。図4-12の左側の点線部がCPUとのインターフェース部であり、そこを出入りする

信号がCPUまたは主メモリとのインターフェース信号である。これらの中で＊記号のついた信号が標準的なCPUでは、標準に用意されている信号である。

前記で述べたメモリ制御ステータスメモリのアクセスには、メモリアクセス時にメモリアドレスがCPUの外部に出ていることが必要である。従ってサーバ、モバイル端末とも内部にキャッシュメモリを持つことはできるが、その場合にはストア(ライト)スルー方式の制御方式とすることが必要である。

4.3.2 MMMのハードウェアとソフトウェアの分担

MMM制御は、命令の実行において性能にかかわる最小限度の部分についてハードウェアを用意し、その他についてはすべてソフトウェアにより処理する。次にMMMの主要な制御についてハードウェアとソフトウェアの分担を示す。

(a) ハードウェア制御

・メモリアクセス時にメモリ制御ステータスメモリを読み出し、メモリアクセスの可否を判定し、メモリアクセスが正常に可能でさらにライトの場合はDビットを1にセットし、メモリアクセスが不可の場合にはメモリ例外の割込を発生させる制御回路。

・メモリ制御ステータスメモリのエントリを、ラインアドレスを指定してV、C、Dの各ビットをセットまたはリセットする回路。

・共通領域設定レジスタ、ラインサイズ指定レジスタ、ブロックサイズ指定レジスタ、MMM各種モード指定レジスタの各レジスタと設定回路。

(b) ソフトウェア制御

・メモリ制御ステータスメモリの、ライト時のDビットのセット以外のステータスの設定。

・ラインのデータ転送。

・MMMの各管理方式の制御、MMMの各種制御用のレジスタ設定。

4.3.3 MMMの性能への影響

MMMを導入することによる性能への影響は次のとおりである。メモリ制御ステータ

表 4-4 基本 MMM のメモリ管理方式の特徴

	メモリ参照	内容の一貫性	通信接続	データ転送量
管理方式1	アクセス権は一方のラインのみ，アクセス権のないラインにアクセスを行う場合には，通信をおこない他方よりアクセス権を取得	常に最新	使用する領域のアクセス権をすべて取得すれば，その後は非接続での利用可	3方式の中では最も多い
管理方式2	一方にライト権があるとき，他方はリード権，リード権のあるラインにライトを行う場合には通信を行い他方よりライト権を取得		常時接続して利用，ただしリードオンリーでの利用であれば，使用する領域のライト権またはリード権をすべて取得することで，その後は非接続で利用可	3方式の中で中間
管理方式3		リード権のあるラインは必ずしも最新でない．最新化のためには同期処理が必要	使用する領域のライト権またはリード権をすべて取得すれば，その後は非接続での利用可，ただしライト権の場合にはリード及びライト，リード権の場合にはリードのみでの利用	3方式の中では最も少ない

注　アクセス権はライトまたはリードが可能、ライト権はライトとリードが可能、リード権はリードのみ可能

スメモリを高速SRAMで構成し、表4-1のメモリステータスの判定をCPUの1クロック以内で終了させることにより、必要とする共通メモリのデータが自身のメモリにある場合には従来の性能でのメモリアクセスが可能である。必要とするデータが自身のメモリにない場合には、メモリ例外の割込が発生し割込処理のプログラムが実行され、他方のCPUと通信を行い必要なデータを他CPUから転送を行う。この割込処理時間の大部分は通信時間が占めることになり、その他のMMM制御のためのソフトウェア処理時間の割合は小さい。他のCPUとの通信はMMM以外の方式であっても、各種方式により効率は異なるが必須である。以上のようにMMMを導入することによる命令実行の性能低下は基本的には発生しない。

4.3.4 MMMの実装法

MMM上で性能に影響を与えることなく命令を実行するためには、4.3.1項で述べた付加ハードウェアが必要であるが、MMM上で命令実行の必要がなければ、すべてソフトウエアでの処理が可能である。MMM上での命令実行がモバイル端末上だけで、サーバ側でMMM上での命令実行の必要のない使用方法であれば、付加ハードウェアはモバイル端末側のみにあればよく、サーバ側ではMMMの制御を命令実行性能に影響を与えることなく、すべてソフトウェアの処理とすることが可能である。

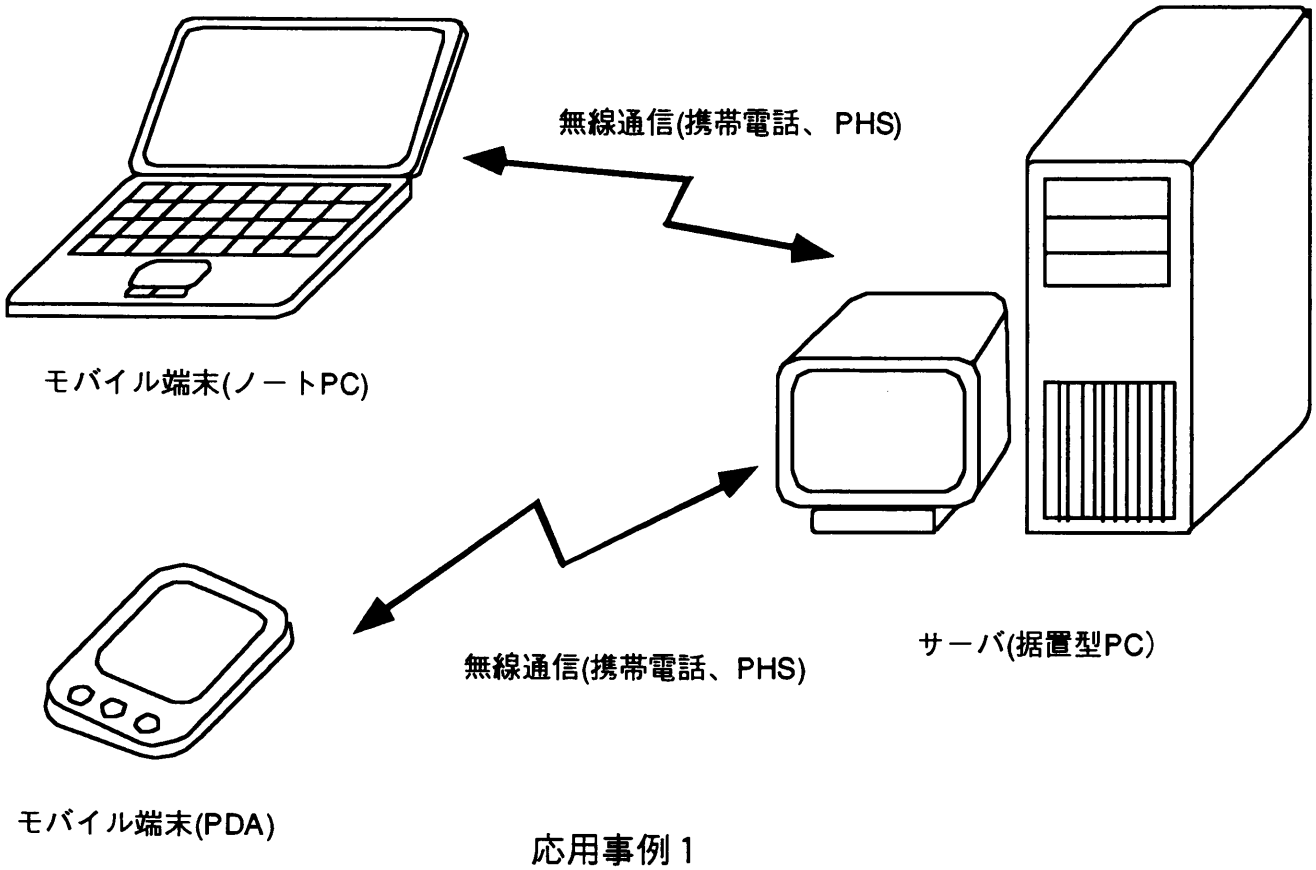
4.4 基本MMMの各管理方式の特徴と応用

基本MMM管理方式1、基本MMM管理方式2及び基本MMM管理方式3にはそれぞれ長所と短所があり、特徴を表4-4にまとめる。モバイル端末の利用形態による使い分けについてはつぎに述べる。

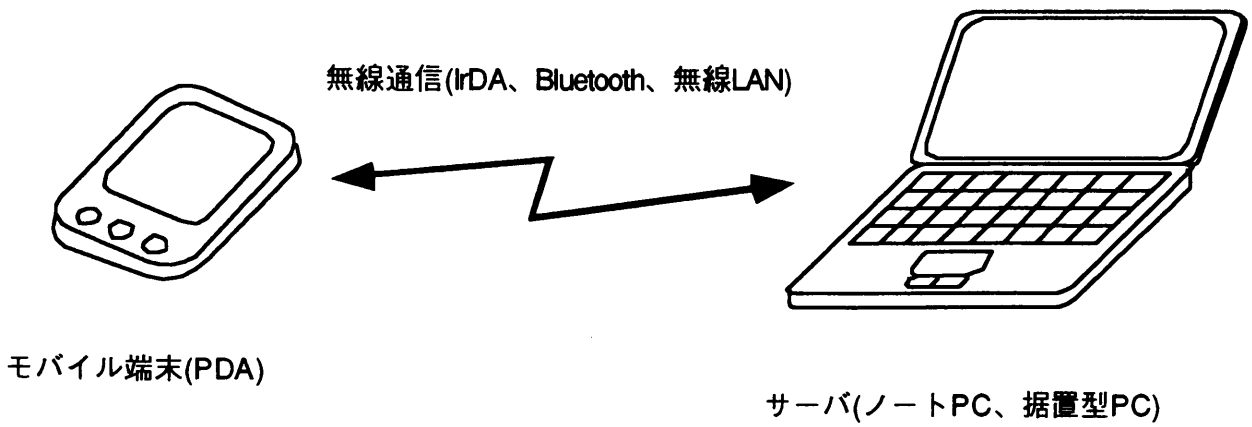
モバイル端末での共通データの利用形態を次のように分類する。

(a) 必ず通信を行い、同期を取って利用する方式で、共通データの内容は常に一致し、モバイル端末とサーバとの双方で内容の変更をおこなう使用方法。

(b) 移動前にモバイル端末に共用データをサーバよりダウンロードし、移動中のモバイル端末はリードオンリであれば、単独で使用し，内容を変更する場合にはサーバと通信し同期を取って利用し、サーバ側ではリードオンリで使用。



応用事例1



応用事例2

図 4-13 MMM の応用事例

(c) 移動前にモバイル端末に共用データをサーバよりダウンロードし，モバイル端末の移動中は単独で読み出しまたは書込を行い、移動から戻った時点でサーバと同期を取る方式で、サーバ側では同期をとるまで最新のデータを使用しない。

(d) モバイル端末が共通データを使用する場合には、サーバと通信を行い必要なデータをダウンロード、モバイル端末がデータを変更する場合にはサーバと同期をとって変更、サーバがデータ変更をする場合にはモバイル端末には通知せずに単独で変更、モバイル端末が最新のデータを必要とする場合にはサーバと同期をとって使用。

(a)は、通信と表示機能に特化し、モバイル端末を使用する場合には必ずサーバと連携して利用するような端末向きであり、基本MMM管理方式1または基本MMM管理方式2が適している。データの内容は常に一致するが、基本MMM管理方式1ではデータは交互に使用される。(b)は通信非接続での使用を基本とした端末向きであり、基本MMM管理方式2が適している。これは片方がリードオンリの使用方法のため基本MMM管理方式2により通信トラフィックが減少することと、内容は常に一致することによる。(c)も通信非接続での使用が基本であるが、モバイル端末の使用中にサーバが共用データを使用しない場合には基本MMM管理方式1、並行してリードオンリで使用する場合には基本MMM管理方式3が適している。いずれの方式もモバイル端末の使用中はサーバとの通信は発生しない。(d)はモバイル端末からイントラ/インターネットのホームページを見るような使い方であり基本MMM管理方式3が適している。これはサーバ側のデータ更新の頻度が少なく、モバイル端末側はほとんどの場合リードオンリでの使用を基本とする使用方法である。

モバイルコンピューティングシステムの応用事例を図4-13に示す。図中の応用事例1はノートPCやPDAをモバイル端末とし、事務所等に設置された据置型PCをサーバとして、その間を携帯電話やPHS等の広域の無線通信手段を用いて通信を行う例である。応用事例2はPDAをモバイル端末としノートPCや据置型PCをサーバとしてその間を赤外線通信のIrDAやBluetooth等の近距離通信手段により通信を行う例である。

4.5 結言

モバイルコンピューティング環境向きメモリ管理方式(MMM)の中の、基本MMMの3つのメモリ管理方式についてそのアーキテクチャと特徴を述べた。基本MMM管理方式3は、メモリの内容の一貫性管理について若干の緩和を行うことにより動作の並列性を高めたものであり、そのモバイル端末の利用形態に応じた使い分けについても議論した。なおメモリ内容の一貫性管理に関しては、従来の分散共有メモリで論じられてきたものと同様に論じることができ、これについては第5章で述べる。

5. モバイルコンピューティング環境向きメモリ管理方式の評価

5.1 緒言

モバイルコンピューティング環境向きメモリ管理方式は一種の分散共有メモリと考えることができるが、いくつかの点で従来の一般の分散共有メモリとは異なっており、5.2節でその差異について議論する。つぎにモバイルコンピューティング環境向きメモリ管理方式をモバイルコンピューティングシステムに適用するにあたり、考慮すべき項目について述べ、最後にアプリケーションのモデルを構築しシミュレーションによる評価をおこなう。

5.2 MMMと他のメモリ方式の差異

5.2.1 他のメモリ方式との比較

MMMと分散共有メモリ、キャッシュメモリ及び仮想メモリとの差異を表5-1に示す。

5.2.2 一般の分散共有メモリとの相違点

MMMと一般の分散共有メモリ及び分散キャッシュ型共有メモリとの主要な相違点を次に示す。

(a)一般の分散キャッシュ型共有メモリでは、図5-1に示すように主メモリは全体として一つのアドレス空間を構成、各主メモリはアドレス空間の一部を構成し、各プロセッサは自身のキャッシュ内に必要とする主メモリの写しをコピーすることによりメモリを共有する。これに対してMMMでは、図5-2に示すように主メモリの一部に共通メモリ領域を設けて、主メモリ間で同一内容となるように管理し、これをプロセッサ間で共有する方式であり、分散キャッシュ型共有メモリのキャッシュに対応するものは存在しない。

(b) 通信路が低速なモバイル環境に適したラインサイズの選択が可能であり、さらにラインを分割したブロック単位の書き戻し制御により通信量の最小化、通信コストの低減化が可能。

表 5-1 MMM と分散共有メモリ、キャッシュメモリ及び仮想メモリとの差異(1/2)

	MMM	分散共有メモリ
目的	1. 無線通信で接続されるサーバとクライアント間の無線通信を隠蔽しプログラミングの容易化 2. 通信の効率化 3. データの同期	1. 物理的に離れた多数のプロセッサを協調して動作させ、単一プロセッサでは得られない大容量及び高速のデータ処理用途
一般的構成	1.各主メモリの一部を共通メモリの領域とし、それらの内容を一致するように管理し、各プロセッサで共有 2. 共通メモリの領域をラインと呼ぶ一定の単位に分割し、そのステータスを専用のメモリ制御ステータスメモリにより管理 3. ラインを分割したブロックの概念を導入し、データ転送量を軽減 4. プロセッサ間は無線通信による接続であり、帯域幅が狭く、意図しない切断への対応と、通信非接続での利用が可能	1.各主メモリを単一メモリ空間の一部に割り付け、全体として単一メモリ空間化し、各プロセッサで共有 2.キャッシュ型分散共有メモリでは、各プロセッサは共有のためのキャッシュメモリを設け、必要とするブロックを他の主メモリから自分のキャッシュにコピーして使用 3. プロセッサまたはメモリ間は必ず通信できることが前提であり、通信帯域幅は比較的広い
プログラムからの見え方	1. メモリ領域の一部が共通領域として設定され、共通領域は自及び他のすべてのプロセッサで内容が同一であり、共有して使用	1. 論理的には単一のメモリ空間であるが、プログラムの効率的な実行には、物理的な配置を意識したプログラミングが必要
特徴	1. 各共通メモリ領域は基本的には対等であり、必用時にラインを自分のメモリに転送して使用 2. 有効と無効の中間の“最新でない可能性がある”という概念を導入し、可用性の向上、通信の効率化 3. ラインサイズは32-256バイト程度、アプリケーションの動作方法にもよるが、大きくすると未使用の割合が増大し効率が低下	1. プロセッサ間は一般的には有線通信回線であり、帯域幅は比較的広く、常時接続が前提、切断は考えていない

表5-1 MMMと分散共有メモリ、キャッシュメモリ及び仮想メモリとの差異(2/2)

	キャッシュメモリ	仮想メモリ
目的	1. 比較的遅いメモリに対して メモリアクセスの高速化 2.通信トラフィックの削減 (ソフトウェアキャッシュ)	1. 実メモリを超えたメモリの 大容量化 2. プログラムの動的再配置
一般的構成	1.当面のプログラムの実行に 必要な主メモリの領域をライン を単位として、キャッシュメモ リと呼ばれる高速メモリにコ ピーして使用 2. キャッシュメモリに写しがある かどうか示すタグメモリと呼ば れるアドレスサーチ用 の高速メモリと、データ格納用の 高速メモリより構成 3. ラインサイズは16-128バイト 程度、メモリの読み出し 幅によるが、大きくすると次の アクセスでミスヒットしたとき に悪影響	1. 仮想メモリと呼ぶ論理上の メモリを定義し、当面のプログ ラムの実行に必要な分を、実メ モリに置き、他を外部記憶に置き、 この対応を示すアドレス変換 テーブルを主メモリ上に用意 して管理 2. アドレス変換の高速化のため TLBと呼ばれるアドレスサーチ用 の高速メモリを用意
プログラムからの 見え方	1. 原則的にはプログラムからは透過 であるが、構成によっては無効化 処理を、プログラム側の責任 行う必要がある場合もある	1. 一般のプログラムからは、仮想 メモリが通常のメモリとして見える
特徴	1. キャッシュメモリは基本的には主 メモリの写しである	1.主メモリをキャッシュメモリ、 外部記憶を主メモリと見なすと 一種のキャッシュメモリと考える ことができる

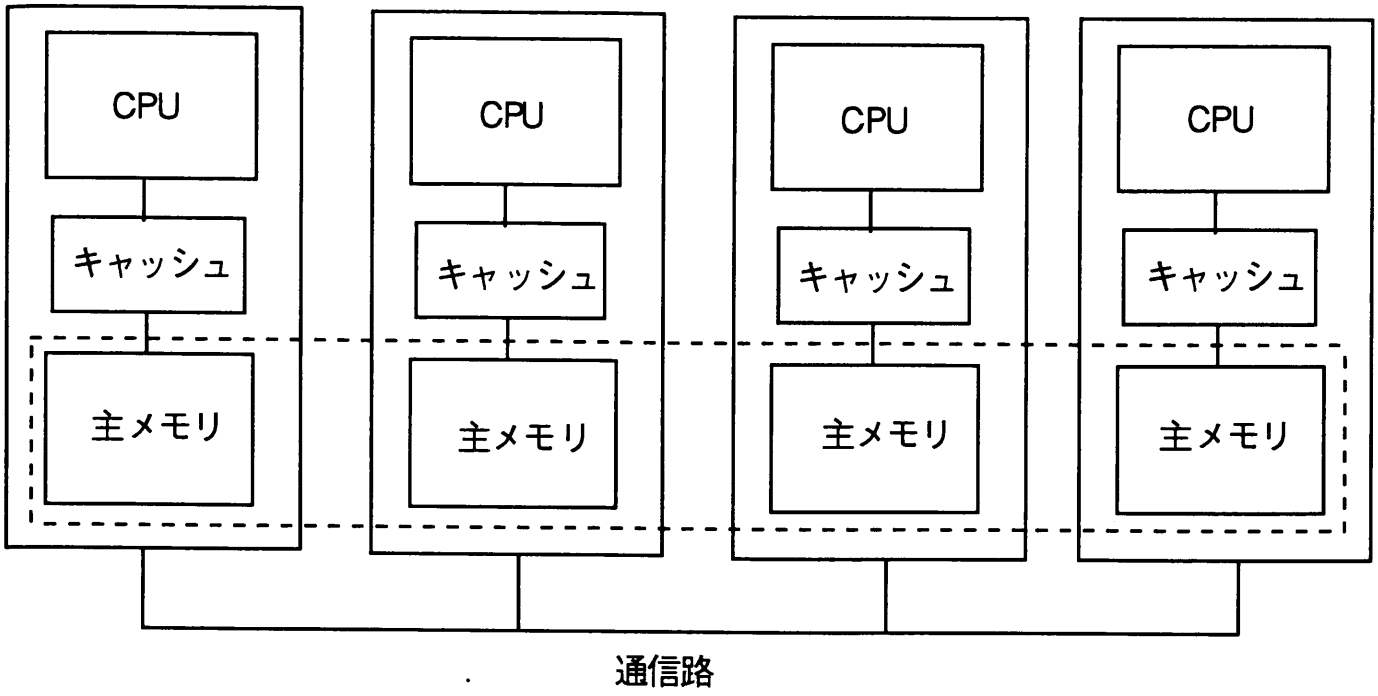


図 5-1 一般の分散キャッシュ型共有メモリ

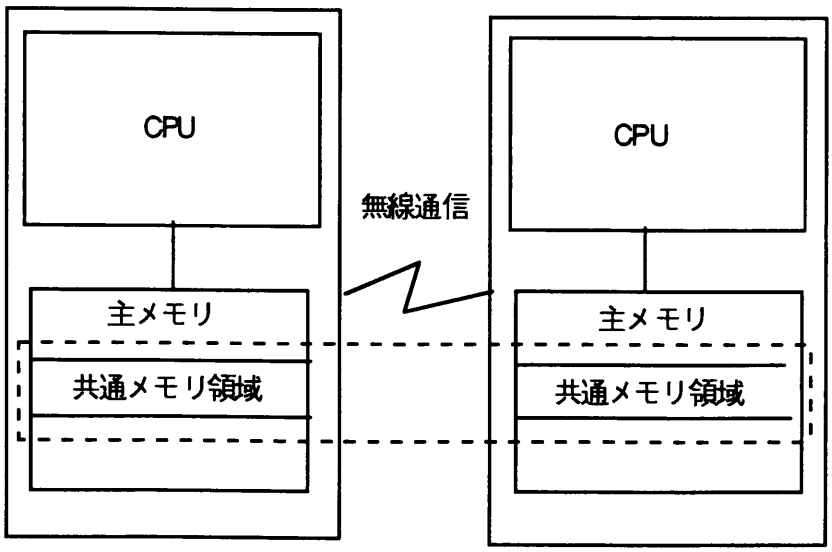


図 5-2 MMM の共通メモリ方式による共有メモリ

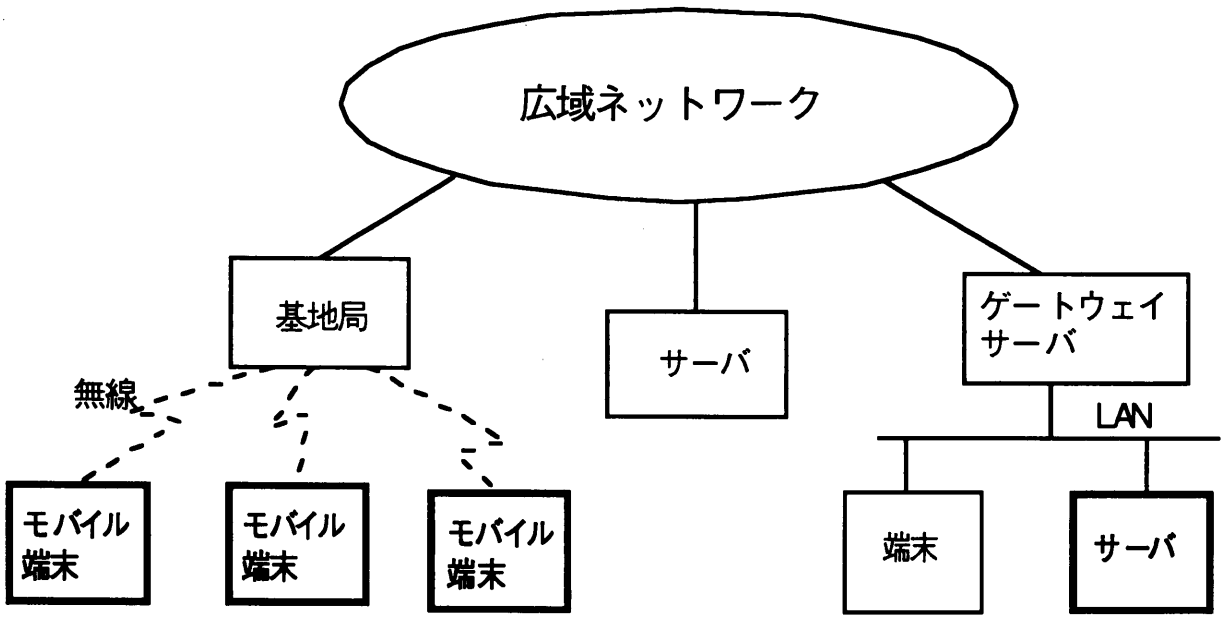


図 5-3 コンピュータネットワークの中でのモバイル
コンピューティングシステム

(c) 通信中の切断への対応及び通信非接続での使用が可能。

(d) MMMでは図5-2に示すように、分散キャッシュ型共有メモリに必須なキャッシュにあたるものがなく、さらにモバイルコンピューティング環境下での利用を前提に、モバイル端末とサーバとの対向するプロセッサ間でのメモリ共有に限定することで管理するステータスの数を減らし、一貫性制御方式を比較的単純化。

(e) 他方のメモリを仮想記憶の二次記憶とすることでメモリ空間の拡大が可能。

5.2.3 MMMとキャッシュメモリの無効化処理について

キャッシュメモリは、メモリアクセスの高速化用に、CPUと主メモリの間に置かれる高速のバッファメモリであり、CPUが当面必要とするメモリの写しを持つ。キャッシュメモリは主メモリの写しのため、何らかの原因で主メモリの内容との不一致が発生する場合があり、この場合に無効化処理が行われる。これは例えばマルチプロセッサで各プロセッサがキャッシュを持ち、一つのプロセッサがキャッシュのラインを変更し、他のプロセッサのキャッシュに同じ内容の写しを持っていた場合に無効化の処理がおこなわれる。

これに対してMMMでは、無効化処理ではなく、Dビットを使用し、ラインではなくブロックの単位で変更管理がおこなわれる。すなわちキャッシュメモリでは一旦無効化処理がおこなわれると、その内容が再利用されることはないが、MMMでは変更を受けたブロック以外のラインの他の部分について再利用がおこなわれる。特にMMMでは、キャッシュメモリに主メモリの写しを転送するという方式ではなく、主メモリのオリジナルを自分の側に転送してから使用するという方式が基本である。

5.3 MMMのモバイルコンピューティングシステムへの適用

5.3.1 MMMの適用

コンピュータネットワークの中で、MMM方式を適用したモバイルコンピューティングシステムの例を図5-3に示す。図中、点線の部分が無線通信区間であり、他は有線の通信ネットワークである。複数のモバイル端末が無線通信を通じて、広域のネットワー

クに接続されたサーバに接続されるモバイルコンピューティングシステムに対してMMMが適用される。モバイル端末は、具体的には携帯情報端末(PDA)やモバイルPCであり、サーバは事務所や家庭の据置型PCを想定する。

5.3.2 通信媒体

通信媒体は、モバイル環境下では一般的には携帯電話やPHSであるが、赤外線通信や無線LAN等の無線通信の他に、公衆電話網やLAN等の有線通信であってもよい。本論文では広域で使用可能ではあるが最も通信速度が遅い携帯電話での利用を前提に検討する。

携帯電話等の無線通信は、通信路が狭くまた通信コストが高価のため、通信は必要最小限にすることが求められる。MMMは、アプリケーションが共通メモリにアクセスしたときに、通信によりサーバまたはモバイル端末から、ライン単位またはブロック単位でデータを書き移す、必要時要求方式（オンデマンド）のメモリ管理方式であり、さらにラインサイズを通信路に合わせ最適に選択可能のため、データ転送を必要最小限とすることができる。

5.3.3 ラインサイズの選択

MMMは一種の分散共有メモリであるが、一方ではキャッシュメモリの側面も持つ[22-27]。すなわちラインは、使う必要が発生した時点で、自CPUに書き写される。書き移されたラインがプログラムの後続のメモリアクセスで利用されれば、通信の効率はよくなる。一般的には、メモリのアクセスには局所性があり、アクセスのあった近傍のデータが次に使われる確率は高い。したがってラインサイズは、ある程度の大きい方がよいが、あまり大きくすると使われない部分が増加し、かえって効率が下がり、適切なラインサイズがあると考えられる。

ラインのサイズを拡大することは、あらかじめ次に使う可能性のあるデータをプリフェッチすることになる。MMMの場合は、ラインの伝送が終了するまでプログラムの実行が中断するため、速度を上げる効果は少なく、逆に待ち時間が増加するため操作感が悪くなる。操作感を優先しプログラムの中断による待ち時間を1～2秒程度以内とす

ると、通信速度が9600bpsの場合、ラインサイズの上限を1～2Kバイト程度までとする必要がある。適切なラインサイズを考える上でのもう一つの制約は、主たる通信媒体が無線のため、ラインサイズを大きくすると通信誤りが多くなった場合に、再送が増加し通信効率が低下することである。

ラインサイズをLバイトとすると、Nバイトのデータを参照するのに必要なデータ転送時間Tは、次のように表すことができる。なお通信誤りによる再送の時間は含まない。

$$T=8N(1+H/L)/(C*F(L))$$

H: 通信オーバーヘッド　バイト数

C: 通信速度　bps

F(L): ラインの使用率

Hは通信オーバヘッドで、コマンド、メモリアドレス、データカウント、チェックコード等のメモリデータ以外のMMMの通信プロトコルで必要とするデータのバイト数である。F(L)はラインの使用率であり、使用したバイト数を転送したバイト数で除したものである。ラインサイズの関数であり、アプリケーションにより異なるが、一般的にはラインサイズが小さければ1に近づき、ラインサイズが大きくなれば値は小さくなる。ラインサイズの最適値については5.4項以降でシミュレーションにより検証する。

5.3.4 ラインのプリフェッチ

プリフェッチ目的で単純にラインサイズを大きくすることは、プログラムの中断時間増大の問題があるが、要求のあったラインの次のライン、またはなんらかのアルゴリズムにより、次に使用する可能性のあるデータを含んでいるラインのプリフェッチがプログラムの実行と並行可能であれば、実行速度の向上が可能である。一方モバイル端末でのアプリケーションは、インタラクティブなものが多いと考えられ、モバイル端末のユーザーからの応答待ち時間をプリフェッチに利用することで性能の向上を図ることができる。プリフェッチは、アプリケーションが端末ユーザーからの応答待ちになった時点で通信を開始し、応答が戻った時点以降、通信中のラインのプリフェッチが終了した

時点で、プリフェッチを終了させる方式とする。MMMでは、プリフェッチしない場合、要求のあったラインの次の1ラインのプリフェッチ、及び応答待ち時間の間を使用し可能な限りプリフェッチをおこなうという3種の方式を選択可能とし、5.4項のシミュレーションで評価する。なおプリフェッチするラインは、要求のあったラインの次のライン以降昇順にまだ転送されていないラインを選択する方式とする。

5.3.5 ブロックサイズの選択

ブロックは、データの書き戻し時のデータ伝送量を減らす目的でラインをさらに分割したものである。メモリの書込が発生した場合には対応するブロックのDビットをセットし、ラインの書き戻しが発生した時にはDビットがセットされているブロックを書き戻す。変更されていないデータ転送の割合を減らすためには、ブロックのサイズは小さいほうが良いが、逆に書き戻し時のオーバーヘッドが増大することと、Dビットのビット数が増大し、ハードウェアコストが増加するため、これも適性値があると考えられる。ブロックサイズの最適値については、5.4項以降でシミュレーションにより検証する。

5.4 シミュレーションによる評価

モバイル端末上、またはモバイル端末とサーバとの双方で、サーバ上に存在するデータを使用し、各種の処理を行うアプリケーションの例によりMMMの評価をおこなう。初期状態においてはアプリケーションのプログラム部は端末、またはモバイル端末とサーバの双方に格納されており、データ部はサーバに格納されているものとし、メモリの割り付けは、プログラム部は非共通領域に、データ部は共通領域に割り付けるものとする。アプリケーションが使用するデータはすべて実行の開始前に外部記憶よりメモリ上にロードされているものとする。MMM方式は特に断わらない限り基本MMM管理方式1とする。

通信速度は9600bps、MMMで使用する通信プロトコルのメモリデータ以外の部分(コマンド2バイト、アドレス4バイト、データカウント2バイト、チェックコード2バイト)を10バイトとし、ラインサイズによらず固定とする。MMMの通信プロトコルの下位の通信プロトコルは特に指定しないが、9600bpsの通信速度が出るものとした。

シミュレーションの結果において、データ伝送時間は通信接続後、実際にモバイル端末とサーバとの間でデータ通信を行った時間の合計、通信接続時間は10秒の呼設定時間を含み通信接続の開始から通信の切断までの時間であり、データ伝送時間は通信接続時間の内数であり、総実行時間はアプリケーションの実行開始からアプリケーションの終了までの時間で、通信接続時間や端末ユーザーからの応答待ち時間を含む時間である。

モバイル端末上の一般的なアプリケーションによるデータアクセスを想定し、メモリの代表的アクセスのパターンを次のように分類し、それに対応したアプリケーションの例を用意してシミュレーションを行った。

a. 1～2Kバイト程度の単位のデータを連続または非連続にリード

b. 4バイト(ワード)単位のインデックステーブルを、ほぼランダムなアドレスで数回リードした後に，数100バイトのデータにリード

c. 数10Kバイトのデータを連続したアドレスでリード

d. 数10Kバイトのデータの中から、ランダムなアドレスで4バイト(ワード)の単位にデータをアクセス

アクセスパターンaの例として、「スケジュール管理」、アクセスパターンbの例として、「住所録」のアプリケーションのモデルを製作し、シミュレーションによる評価をおこなった。アクセスパターンcについては、20Kバイトのデータを連続して読むモデル、アクセスパターンdについては32Kバイトのデータの中から、ランダムに4バイト単位でデータをリードするモデルでシミュレーションを行った。

作成したアプリケーションのモデルを次に示す。

(a)スケジュール管理

スケジュール管理のデータは，30分単位で予約可能とし，用件を24文字以内でライトするとして、1項目あたりのデータ量を48バイト、1日あたり12時間分として48x24=1152バイト、全部で50日分として画面表示データを含め、総データ量は60Kバイトとし、データはアプリケーション開始時点ですべてメモリ上に置かれる。モバイル端末上での表示画面は初期画面と表示入力画面とし、初期画面で日の指定を行い、表示入力画面で指定された日の1日分のスケジュールデータの表示と、スケジュールの追加または変更を行う。日の指定の順序、参照、変更を組み合わせた操

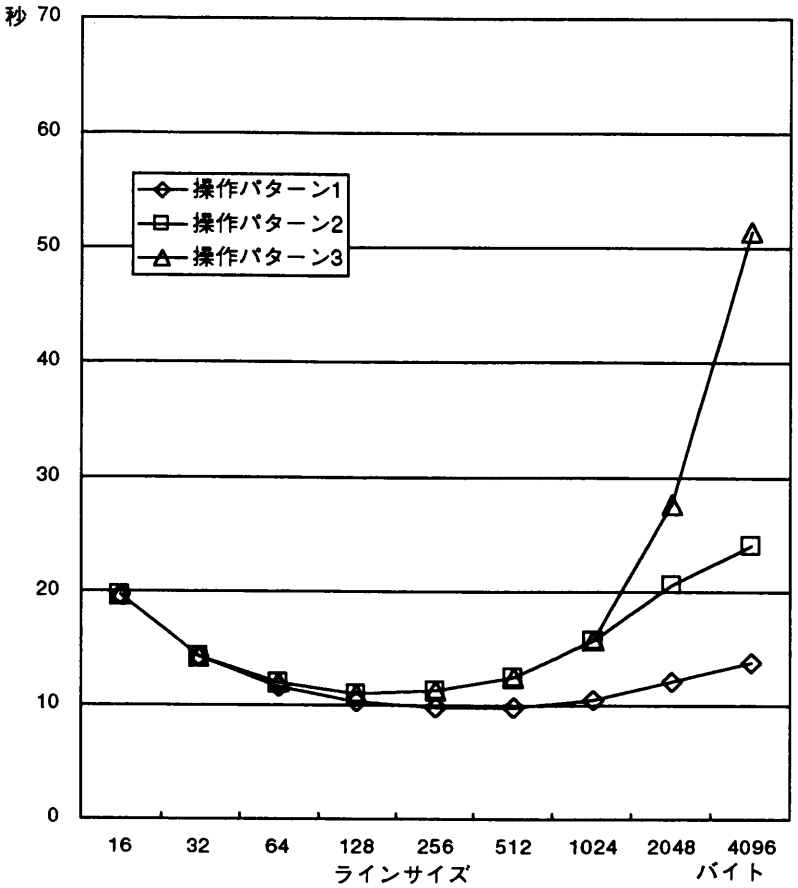


図5-4　ラインサイズとデータ伝送時間
ースケジュール管理ー

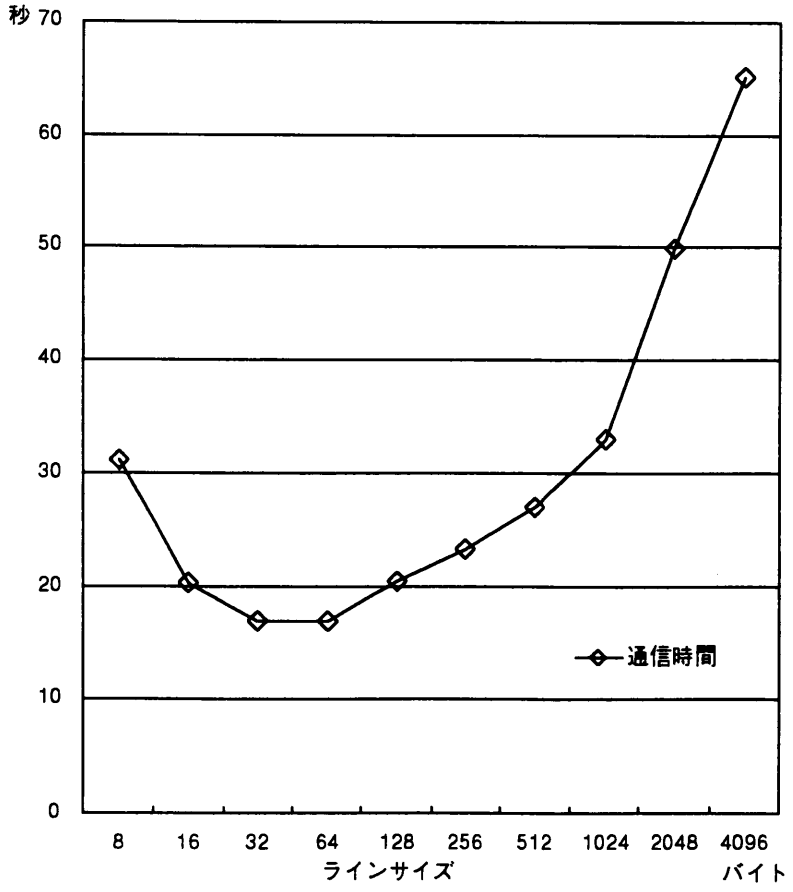


図5-5　ラインサイズとデータ伝送時間
ー住所録ー

作パターンのシナリオを作成してシミュレーションを行った。

(b)住所録管理

住所録のデータは、氏名、住所、電話番号、備考等を合わせて1レコードあたり160バイトとし、500人分のデータと画面データ及びインデックステーブルを含めて総データ量を86Kバイトとし、データはすべてアプリケーションの開始時点でメモリ上に置かれる。モバイル端末上の画面は、初期画面、選択画面、及び表示変更画面から成る。初期画面には名前を選択するための先頭の読みの文字の一覧表があり、最初に検索したい名前の先頭文字を一覧表から選択し、検索を指示することにより選択画面になる。選択画面では選択された先頭文字を持つ名前が一覧表で表示されるので、これらの中から検索したい名前を選択すると表示変更画面となり、検索結果の住所録データが表示される。住所録データの変更及び新規登録は、表示変更画面からおこなう。レコードの追加を行うと、レコードの再ソートを行いインデックステーブルの更新を行う。検索のみの操作と、データの一部の変更を行う操作を組み合わせた操作パターンの例を作成し、シミュレーションを行った。

5.4.1 ラインサイズの評価

最初に、ラインサイズとデータ伝送時間との関係がどのように変化するかを、アプリケーションを変えて調べる。

図5-4は、スケジュール管理について、連続した日にちを順番に参照する操作パターン1，1週間あたり3日参照する操作パターン2、7日飛びに参照する操作パターン3の3種のパターンについてのラインサイズとデータ伝送時間の関係を示す。ラインサイズが64バイトから512バイト近辺が最もデータ伝送時間が短くなることがわかる。ラインサイズが減少するとデータ伝送時間が増大するのは，通信プロトコルの、通信オーバヘッドの割合が増大するためである。通信オーバヘッドは、MMMの通信プロトコルの、メモリデータ以外の10バイトの制御情報を示す。ラインサイズが大きくなったときにデータ伝送時間が増大するのは、再利用しないデータを転送する割合が増加するためである。操作パターン1は、ラインサイズが大きくなっても有効に再利用されるデータの割合が大きく、操作パターン3は、ラインサイズが大きくなると再利用されるデータの

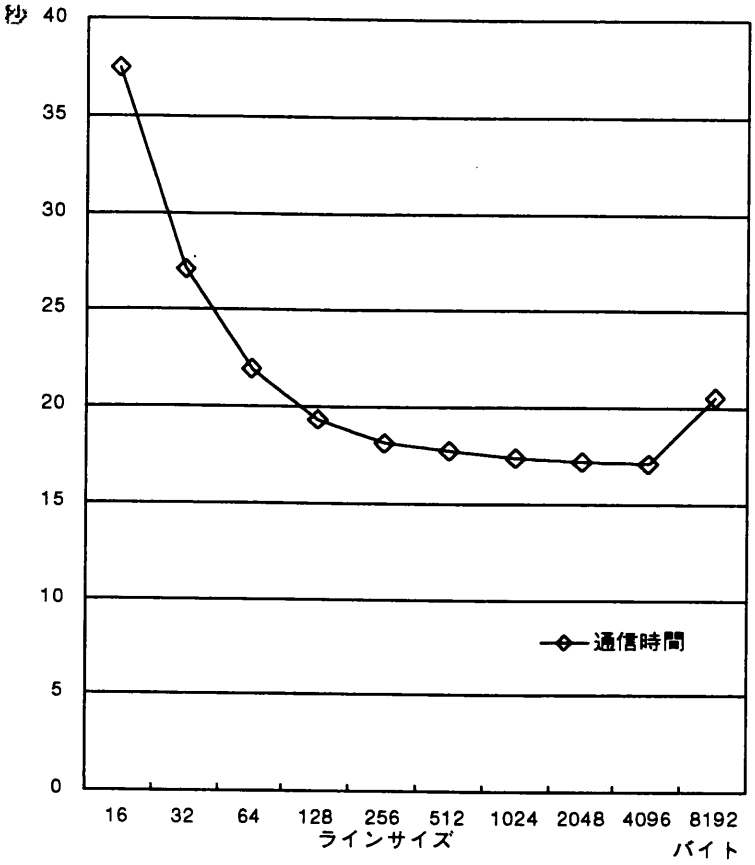


図5-6　ラインサイズとデータ伝送時間
－20Kバイトのデータアクセス－

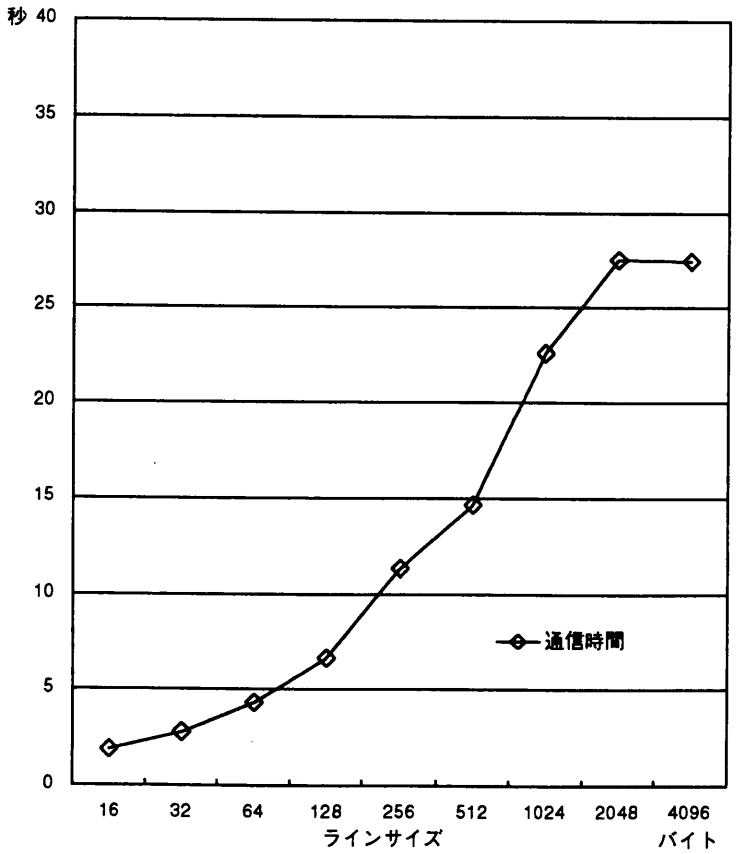


図5-7　ラインサイズとデータ伝送時間
－4バイトデータのランダムアクセス－

割合が少なくなることがわかる。

図5-5は、住所録について、ランダムに10人分のデータを検索したときのラインサイズとデータ伝送時間の関係を示す。ラインサイズが16バイトから128バイト近辺が最もデータ伝送時間が短くなることがわかる。スケジュール管理に比べデータ伝送時間が最小となるラインサイズが小さい方にシフトしているのは、住所録のアクセスするデータ量の単位がスケジュール管理に比べ小さいためである。

図5-6は、20Kバイトのデータを連続したアドレスでリードしたときのラインサイズとデータ伝送時間の関係を示す。連続してリードする場合には，ラインサイズが大きいほどデータ伝送時間が短くなるが、256バイト近辺より改善率が飽和してくる。これはラインサイズが増加するに従い、通信のオーバーヘッドの割合が減少し効率が上がるが、8Kバイトのラインサイズになると、20Kバイトのデータがラインサイズの倍数にならないため、使用されないデータを含むラインを伝送することになり、効率が下がることによる。

図5-7は、32Kバイトのデータの中より、4バイト単位のデータをランダムにアクセスしたときの、ラインサイズとデータ伝送時間の関係を示す。これは局所性が全くない例で、ラインサイズが小さい程データ伝送時間は短く、ラインサイズが大きくなるに従いデータ伝送時間は増加する。

以上まとめると、スケジュール管理は、1Kバイト程度単位のデータを、連続または非連続にアクセスする例であり比較的局所性のあるケース、住所管理は、インデックスでソートされているデータをアクセスするため、4バイトのインデックスを複数回アクセスしたあとに、100バイト～200バイトのレコードをアクセスするという比較的局所性が少ない例であり、20Kバイトの連続データのアクセス例と4バイトのデータのランダムアドレスでのアクセス例は、最も局所性のある場合と無い場合の両極端の例である。これらを総合すると、ラインサイズは、32～256バイト近辺が最適であることがわかる。ただしこれは通信プロトコルのオーバーヘッドが10バイトの場合であり、オーバヘッドが増加すると、最適ラインサイズが大きい方にシフトし、オーバヘッドが減少すると、最適ラインサイズは小さい方にシフトする。

以上の評価では、初期状態から動作を開始し、一連の動作を終了後、初期状態に戻す

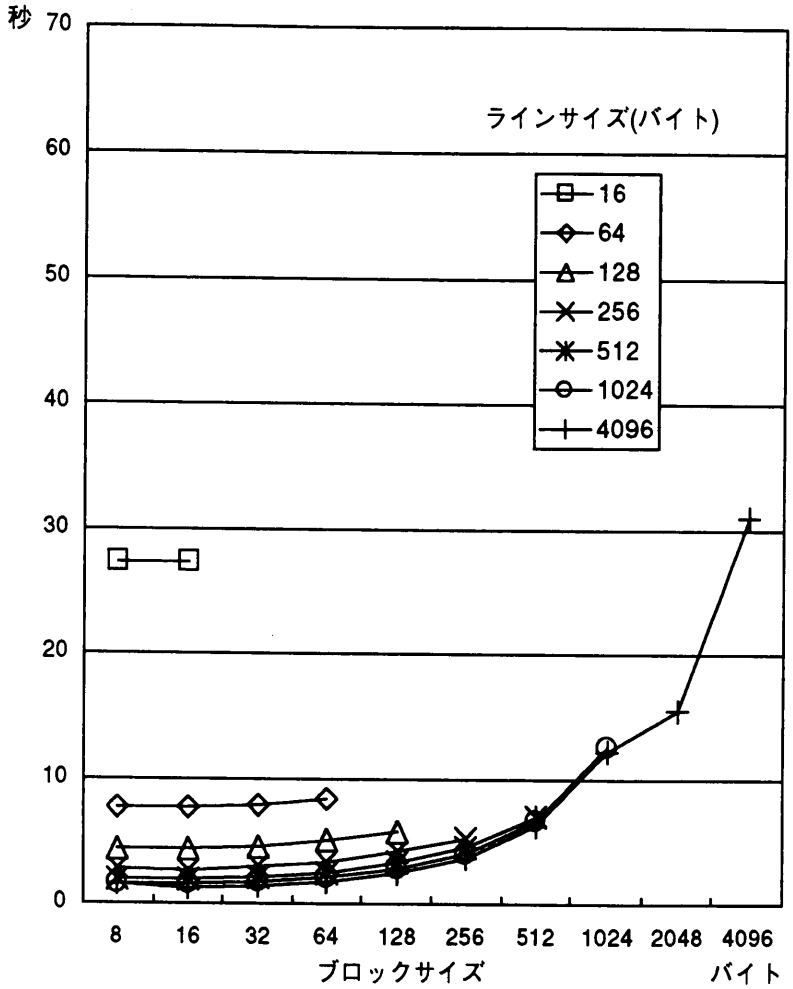


図5-8 ブロックサイズとデータ伝送時間
ースケジュール管理ー

という使用方法を仮定した。異なる複数の使用パターンを組み合わせた継続使用による評価も考えられるが、これはメモリの使用効率が増加し、ラインサイズによる差が減少する方向にあると考えられるため、メモリの使用効率にとっては比較的厳しい、一連の動作で終了させるという使用方法での評価を行った。なおメモリ管理方式は基本MMM管理方式1としたが、以上のシミュレーションは端末側だけの操作のため、他の管理方式でもほぼ同じとなる。ほぼというのは、基本MMM管理方式2、基本MMM管理方式3の場合には、リードの場合リード権の取得を行うが、リードのあとライトが発生するとライト権取得のための制御情報のやりとりの通信が発生するためである。この時間はライトの割合が小さいことと、制御情報のためデータ量が少なく、他の時間に比較するとごく少ない割合である。

5.4.2 ブロックサイズの評価

図5-8にモバイル端末とサーバの相互動作時におけるブロックサイズとデータ伝送時間の関係を、ラインサイズをパラメータにして示す。データ伝送時間は、ラインサイズの影響を大きく受け、ラインサイズは大きい程よく、ブロックサイズは小さい程よいことがわかる。ただし、8バイトのブロックサイズの場合には、ブロック指定のための制御ビットの伝送の割合が増加するため、若干時間が増加する。128バイト程度まではブロックサイズによるデータ転送時間の差は少ないことがわかる。またブロックサイズを小さくすると制御のための変更ビット数が増加するため、ハードウェアコストとの兼ね合いとなる。

5.4.3 MMM方式と従来方式との比較評価 ー その1

次にスケジュール管理を利用し、初期状態の時にサーバ上にあるデータを、モバイル端末上のアプリケーションが使用する場合について、MMMと従来方式との比較評価を行う。シミュレーションの条件を次に示す。

(a)ブロックサイズは16バイトの固定とした．これは、ブロックサイズは小さい方が効率がよくなるが、ブロックを制御するハードウェア(ビット数)が増加するため16バイトとした。

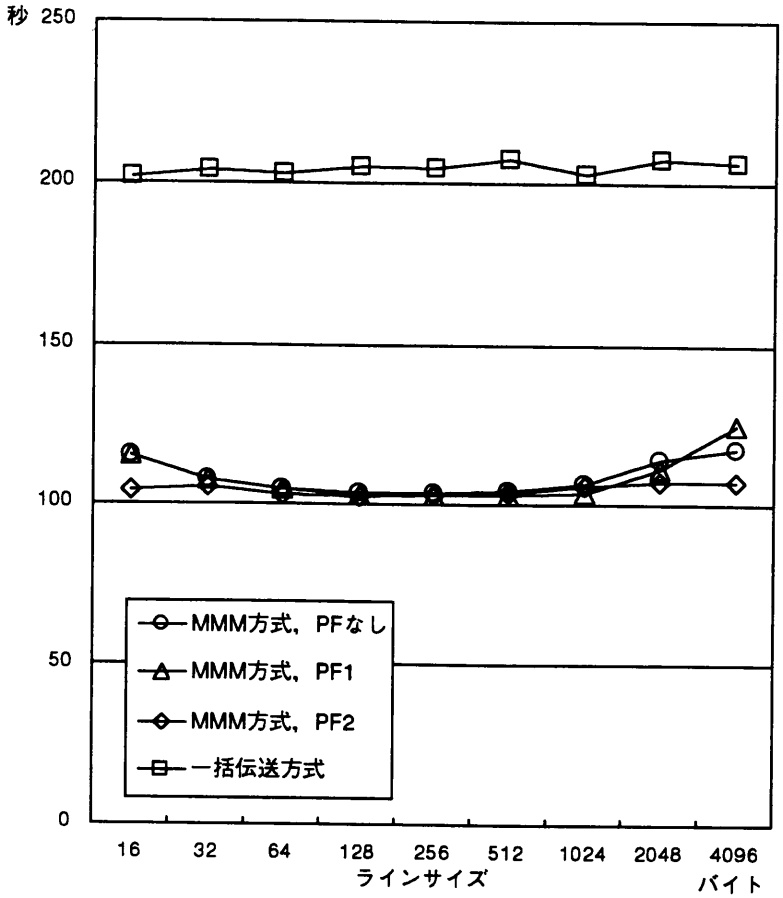


図 5-9 ラインサイズと総実行時間
ースケジュール管理ー

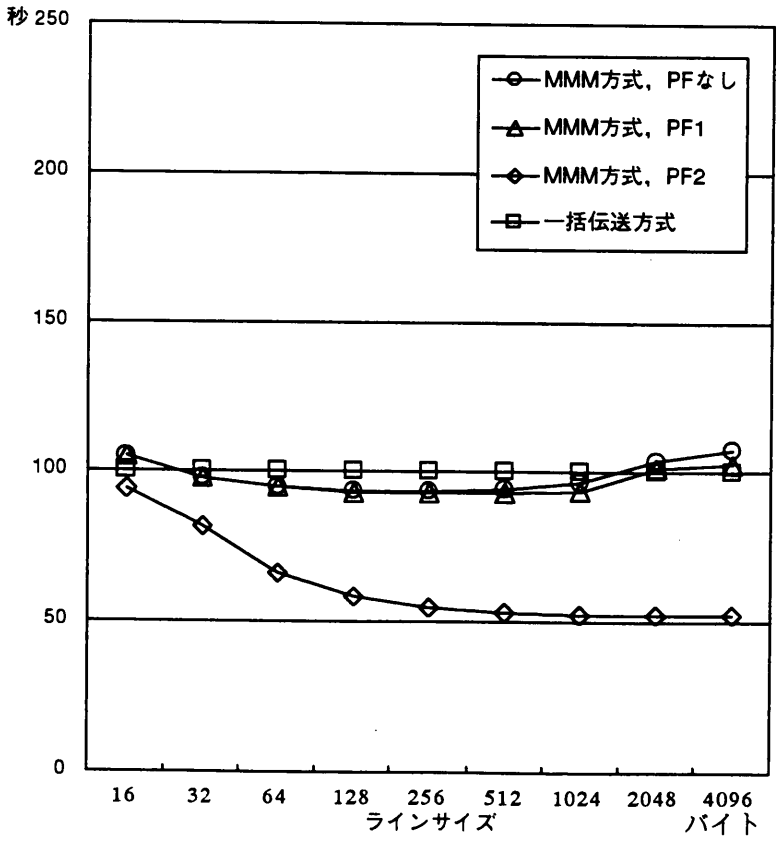


図 5-10 ラインサイズと通信接続時間
ースケジュール管理ー

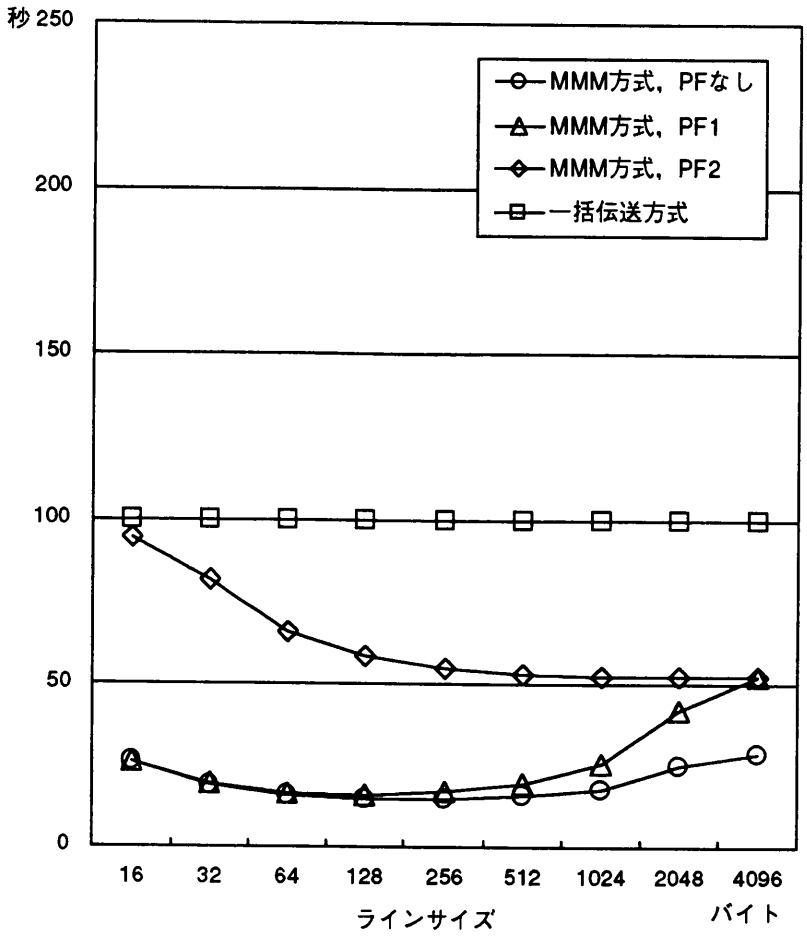


図 5-11　ラインサイズと通信時間
ースケジュール管理ー

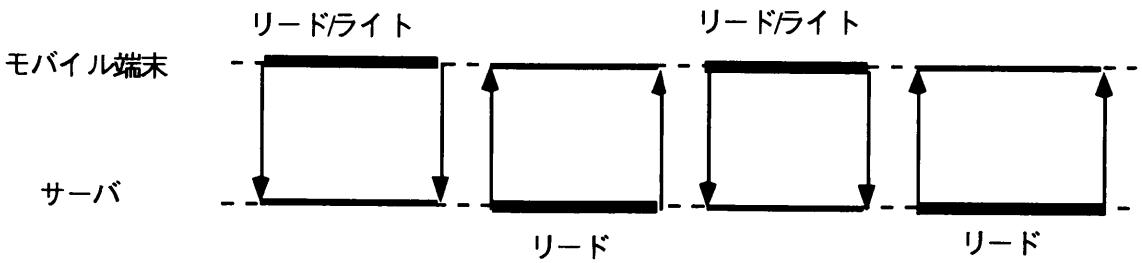


図 5-12 モバイル端末とサーバの相互動作のシナリオ

(b)従来方式でも実際に使用するデータだけを伝送する方式が可能であるが、モバイル端末のアプリケーションがデータの伝送を意識する必要があり、またサーバ側でもデータハンドリング用のプログラムが必要になる等、アプリケーションの構築が複雑になる。このため従来方式としては、アプリケーションの実行の始めに全データを読み込み、実行の最後に全データを書き戻す、一括伝送方式を比較対象とする。

(c)モバイル端末の動作のシナリオは、10日分のスケジュールデータを参照し、その内4日については、データの変更をおこなう操作パターンとする。

(d)1日分のデータが画面に表示された後、参照だけの場合は1～5秒、平均3秒の一様分布を持つ時間を要するものとし、データに変更のある場合には、前記平均3秒の参照時間に加えて、24文字分で12秒の書込時間を要するものとした。

(e)総実行時間には、プログラムの実行時間の他に、データの参照時間、書込時間、および通信の呼設定時間を含む。通信の呼設定時間は10秒とした。

(f)MMMの場合には、プリフェッチする場合とプリフェッチをしない場合を示す。プリフェッチは通信接続中に、データ転送が行われていない時に、直前に転送がおこなわれたラインに続くラインを転送する方式とし、1ラインプリフェッチする場合と(図中PF1で示す)、通信の空き時間のあるかぎりプリフェッチをおこなう場合の(図中でPF2で示す)、2とおりの場合について示す。

(g)通信接続は、一括伝送方式の場合では、最初のデータの書き移しの終了後一旦切断し、最後の書き戻し時に再度接続するものとした。MMMの通信接続は、実行中に全データが書き移されればその時点で通信を切断し、最後の書き戻し時に再接続し、実行中に全データの書き移しが終了しなければ、最後のデータの書き戻しまで通信の切断は行わないものとした。

評価結果を以下に示す。

(a)図5-9は、ラインサイズと総実行時間の関係を示す。一括伝送方式の総実行時間はMMMに較べ倍程度になっているが、この差はほぼ一括伝送方式の全データの書き移しと書き戻しの時間分に相当する。MMMの場合、ラインサイズによる総実行時間の差の割合が少なく見えるのは、画面を参照している時間や書込の時間の

占める割合が大きいためである。一括伝送方式の場合には、一括伝送のデータ伝送時間が総実行時間の中に単純に加算されるが、MMMでは必要とするデータの伝送時間だけが加算され、その時間がその他の時間に比較し割合が少なく、またプリフェッチをおこなうことにより総実行時間がさらに減少することがわかる。

(b)図5-10は、ラインサイズと通信接続時間の関係を示す。通信の空き時間を利用してできる限りプリフェッチするPF2の場合、ラインサイズが大きくなるに従い、時間が少なくなるのは、途中で全データの書き移しが終了し、通信接続を切断するためであり、ラインサイズが大きくなるに従い、早く全データの転送が終了するためである。通信料が接続時間による時間課金の場合には、空き時間になるべくプリフェッチをおこなうPF2の方式が有利であることがわかる。

(c)図5-11は、ラインサイズとデータ伝送時間の関係を示す。実際に必要とするデータだけを伝送するプリフェッチをおこなわない場合が最も通信時間が少なくなるため、通信料がデータ量による従量課金の場合には、この方式を選択するのが有利であることがわかる。

(d)以上総合すると、MMMの場合には、通信接続時間の中で、実際にデータ伝送をおこなう時間だけが総実行時間に加算され、他の通信接続時間は、端末ユーザーの操作時間や待ち時間とオーバーラップしており、さらに、待ち時間を利用したプリフェッチは、総実行時間を減少させることがわかる。これに対して、従来方式は、全データの伝送時間がすべて総実行時間に単純に加算されるため、MMM方式に比べ大きな値になる。

5.4.4 MMMと従来方式との比較評価ーその2

次に、初期状態の時にサーバ側にあるデータを、モバイル端末とサーバ上のアプリケーションが相互に使用する場合について、スケジュール管理を使用し、MMMと従来方式とで比較評価する。シミュレーションの条件を次に示す。

(a)ブロックサイズは16バイトの固定とした。

(b)従来方式は、5.4.3で述べた一括伝送方式を比較対象とした。

(c)モバイル端末とサーバの動作のシナリオは、たとえば、営業マンが客先での打合

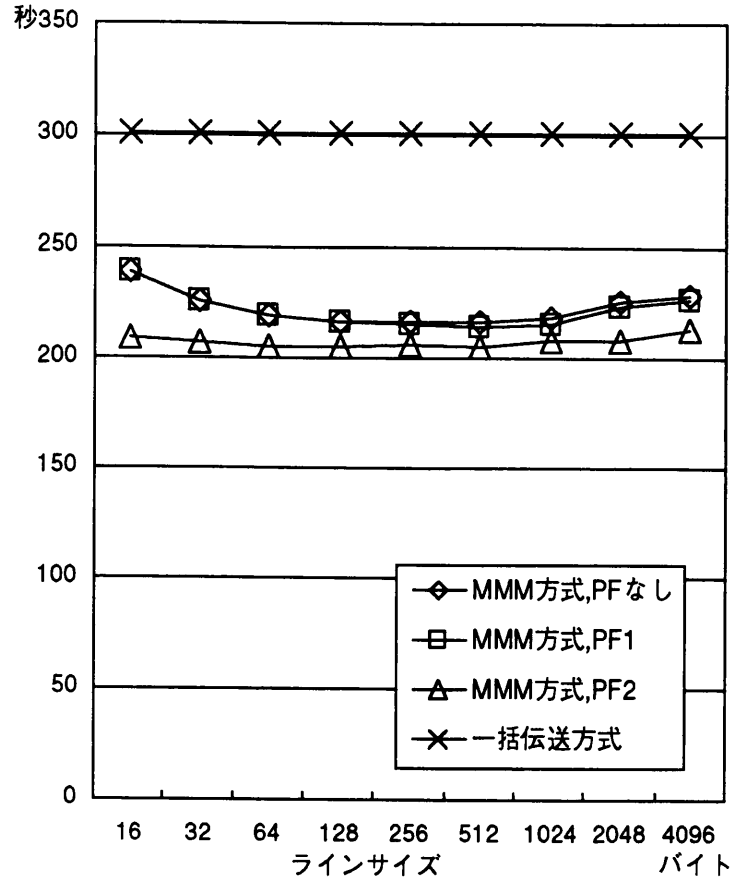


図5-13 ラインサイズと総実行時間
ーモバイル端末上のスケジュール管理ー

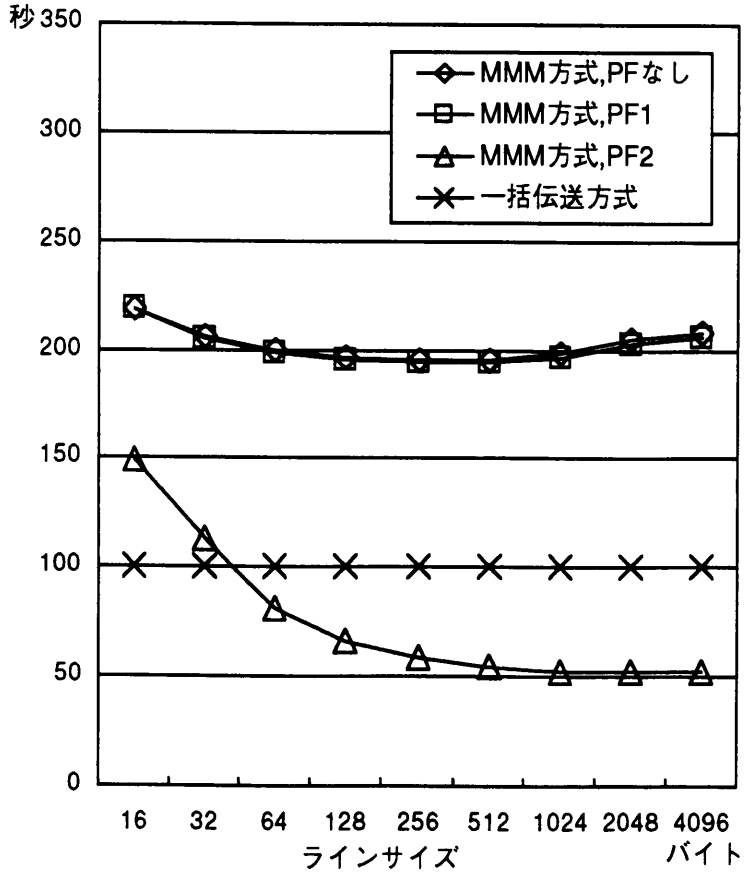


図5-14 ラインサイズと通信接続時間
ーモバイル端末上のスケジュール管理ー

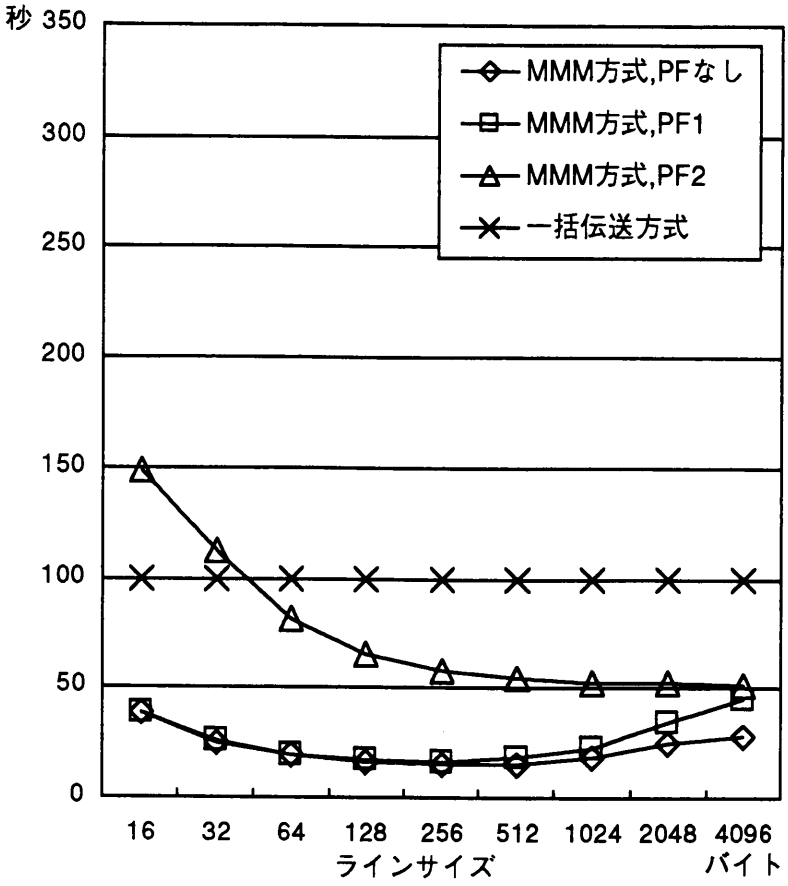


図5-15 ラインサイズとデータ伝送時間
ーモバイル端末上のスケジュール管理ー

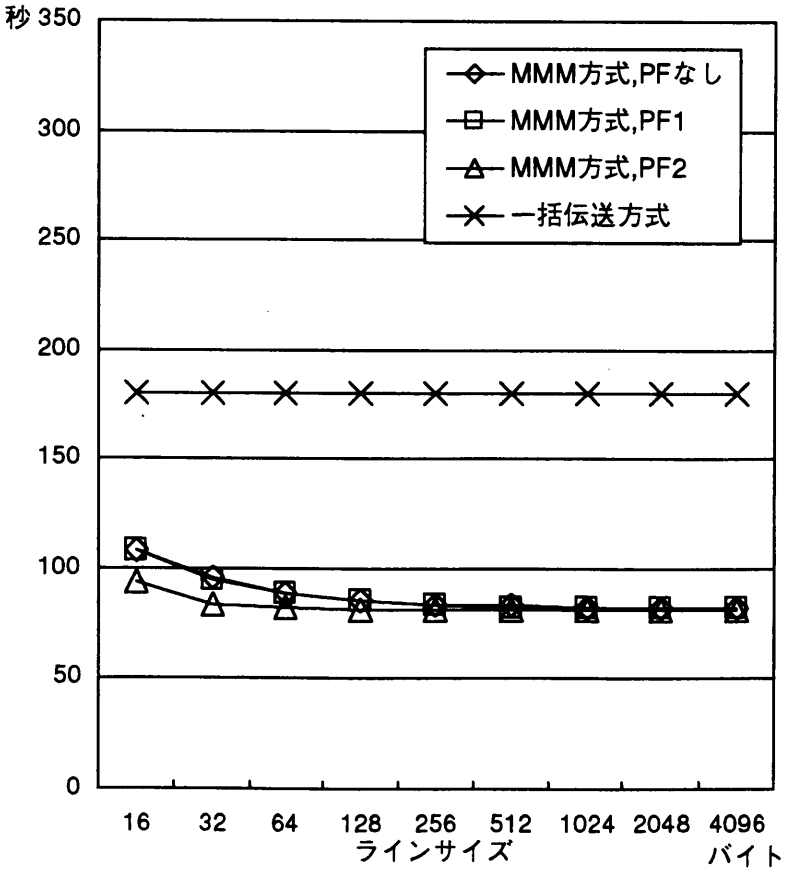


図5-16 ラインサイズと総実行時間
ーサーバ上のスケジュール管理ー

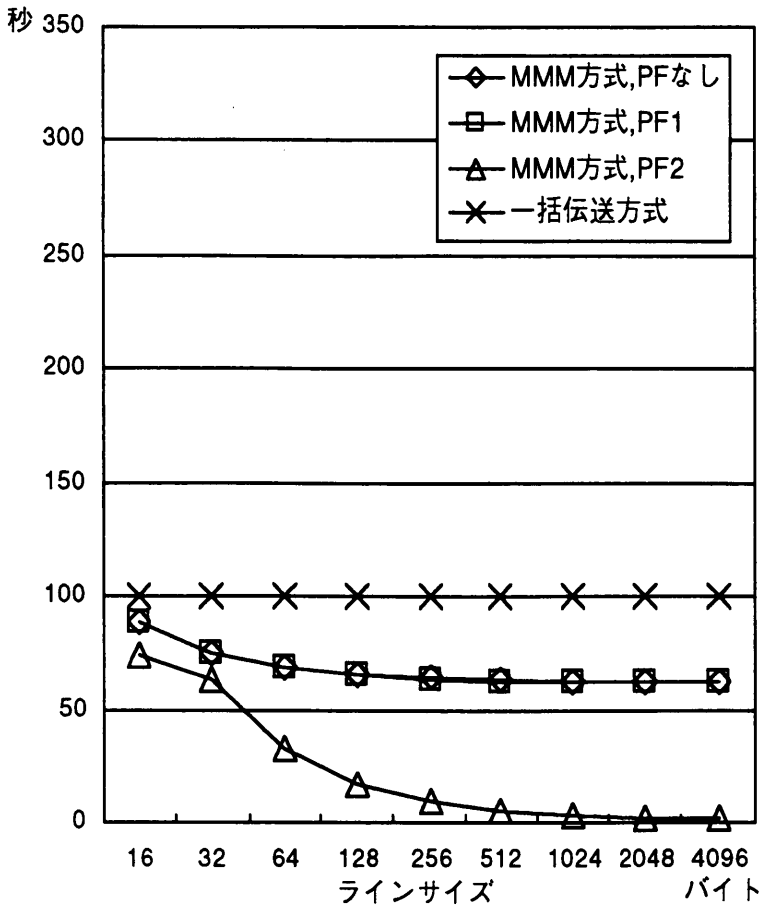


図5-17 ラインサイズと通信接続時間
ーサーバ上のスケジュール管理ー

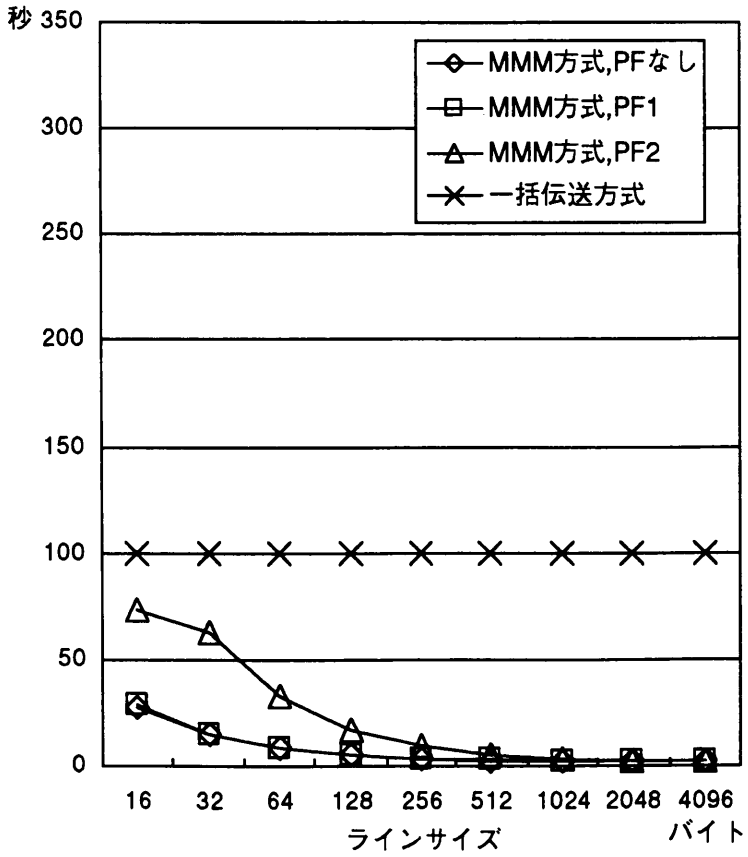


図5-18 ラインサイズとデータ伝送時間
サーバ上のスケジュール管理ー

せで、スケジュールの予約をモバイル端末で行い、事務所のサーバで秘書がスケジュールのチェックを行うという使用方法を想定し、それを図5-12に示す。図において横軸が時間の経過を示し、上段がモバイル端末、下段がサーバを示し、各時間軸上で、太線で示す方の側が、アプリケーションを使用していることを示す。最初にモバイル端末上で10日分のデータが参照され、その内5日分の一部のデータを更新、その後サーバで、モバイル端末上で参照・更新された10日分のデータが参照される。次にモバイル端末上で10日分のデータが参照され、その内5日分の中のデータを更新、最後にサーバ側で、モバイル端末上で参照・更新された10日分のデータが参照される。

(d)1日分のデータが画面に表示された後、参照だけの場合は3秒の時間、データに変更のある場合には前記の参照時間に加えて、24文字分で12秒の書込時間を要するものとした。

(e)総実行時間には、プログラムの実行時間の他に、データの参照時間、書込時間、および通信の呼設定時間を含む。通信の呼設定時間は10秒とした。

(f)MMMの場合には、プリフェッチする場合とプリフェッチをしない場合を示す。プリフェッチは、直前に転送がおこなわれたラインに続くラインを転送する方式とし、1ラインプリフェッチする場合(図中PF1で示す、図5-13～図5-18)と、通信の空き時間のあるかぎりプリフェッチをおこなう場合(図中で、PF2で示す、図5-13～図5-18)の、2とおりの場合について示す。

(g)通信接続は、一括伝送方式の場合は、最初のデータの書き移しの終了後一旦切断し、最後の書き戻し時に再度接続するものとした。MMMの場合は、実行中に全データが書き移されればその時点で通信を切断し、最後の書き戻し時に再接続し、実行中に全データの書き移しが終了しなければ、最後まで通信の切断はおこなわないものとした。

評価結果を以下に示す。

(a)図5-13と図5-16は、モバイル端末とサーバ上でのラインサイズと総実行時間の関係を示す。MMMの時間が一括伝送方式に比較し大幅に少なくなっているが、その差は、ほぼ一括伝送方式の全データの書き移しと書き戻しの時間分に相当する。

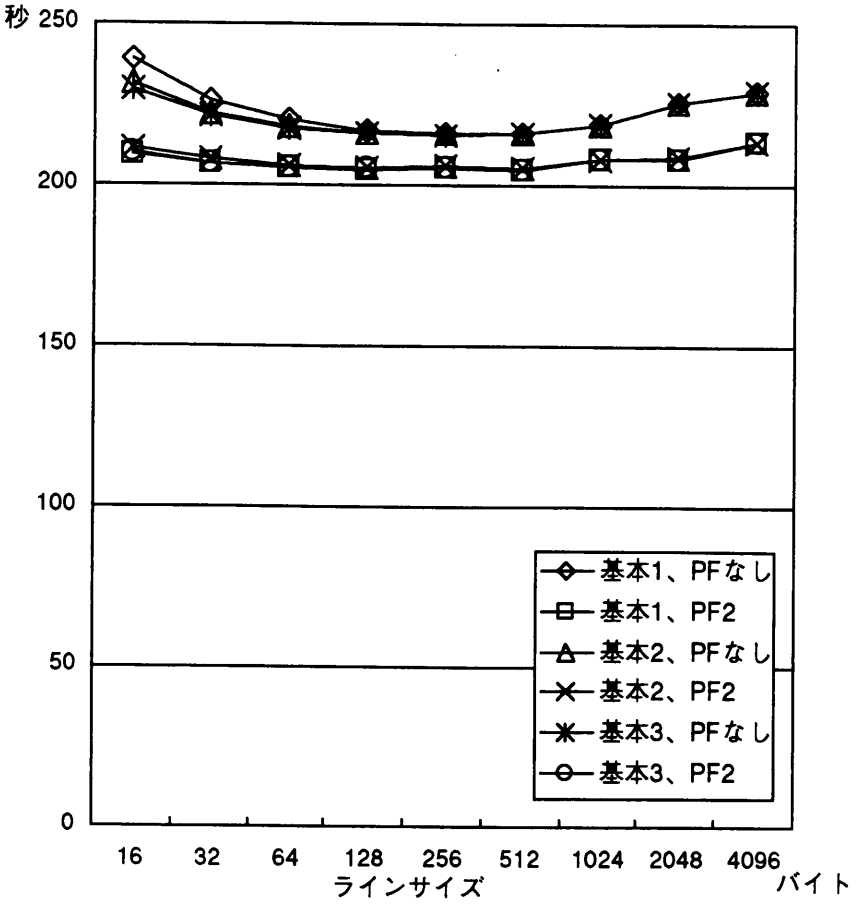


図 5-19 一貫性管理方式についての
ラインサイズと総実行時間
ーモバイル端末上のスケジュール管理ー

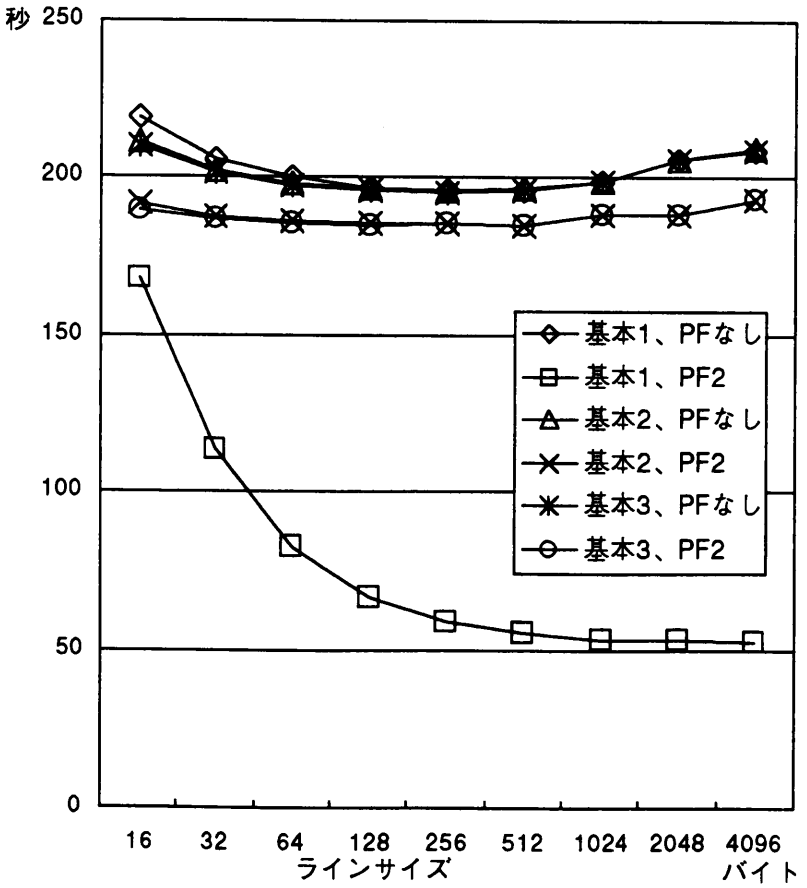


図 5-20 一貫性管管理式についての
ラインサイズと通信接続時間
ーモバイル端末上のスケジュール管理ー

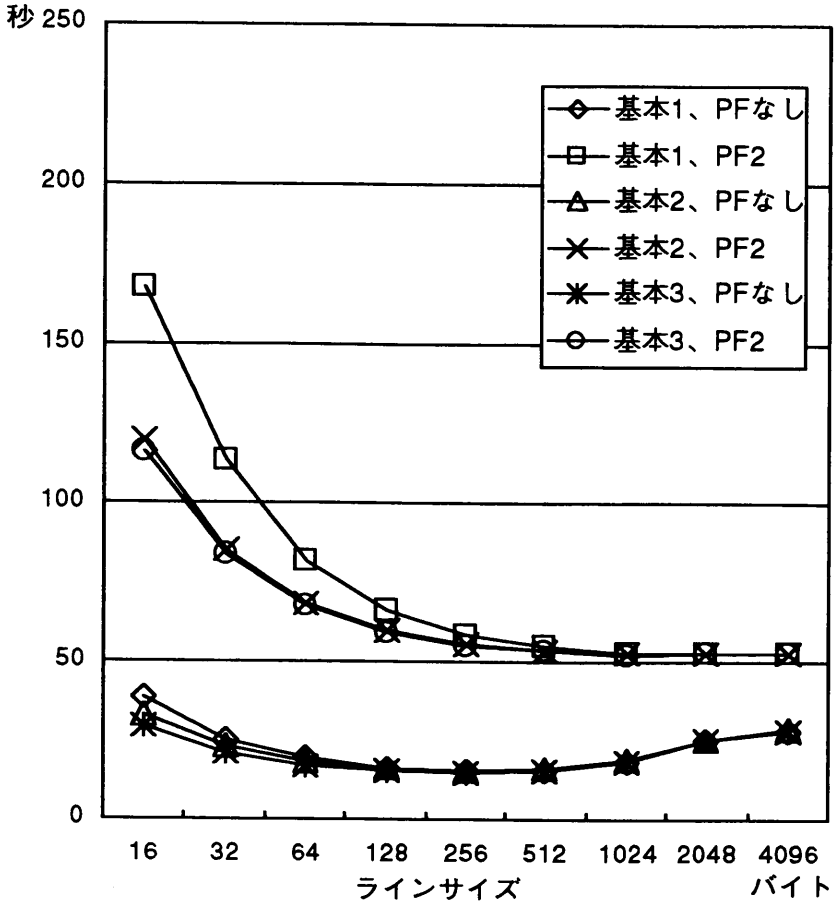


図5-21 一貫性管理方式についての
ラインサイズとデータ伝送時間
ーモバイル端末上のスケジュール管理ー

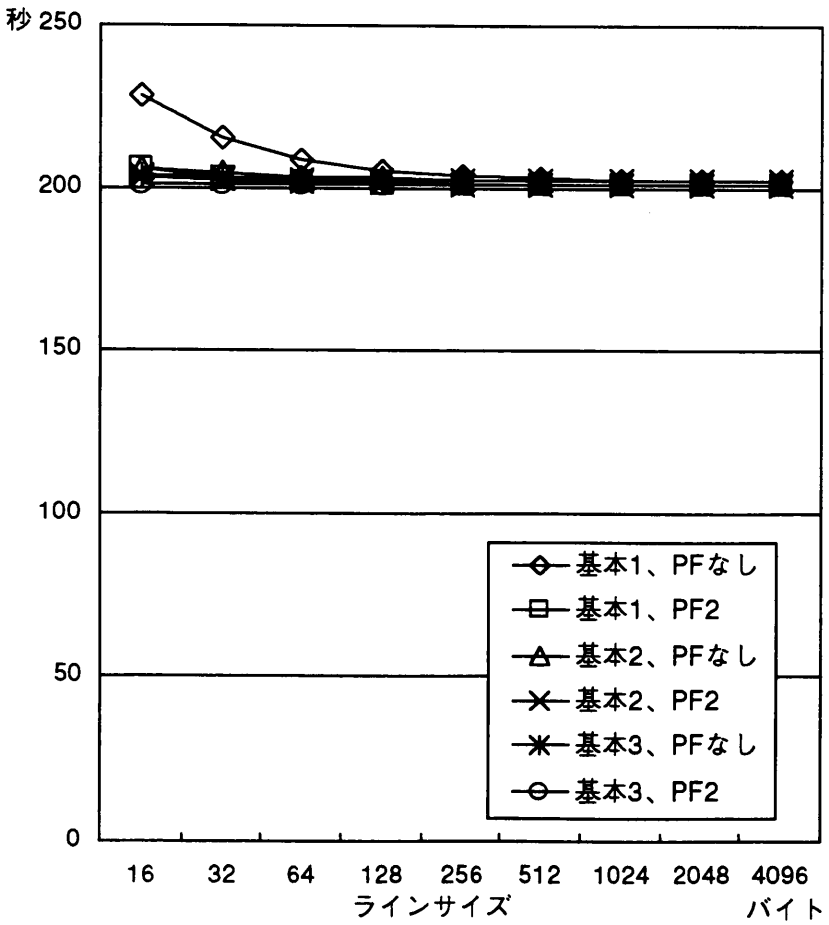


図5-22 一貫性管理方式についての
ラインサイズと総実行時間
ーサーバ上のスケジュール管理ー

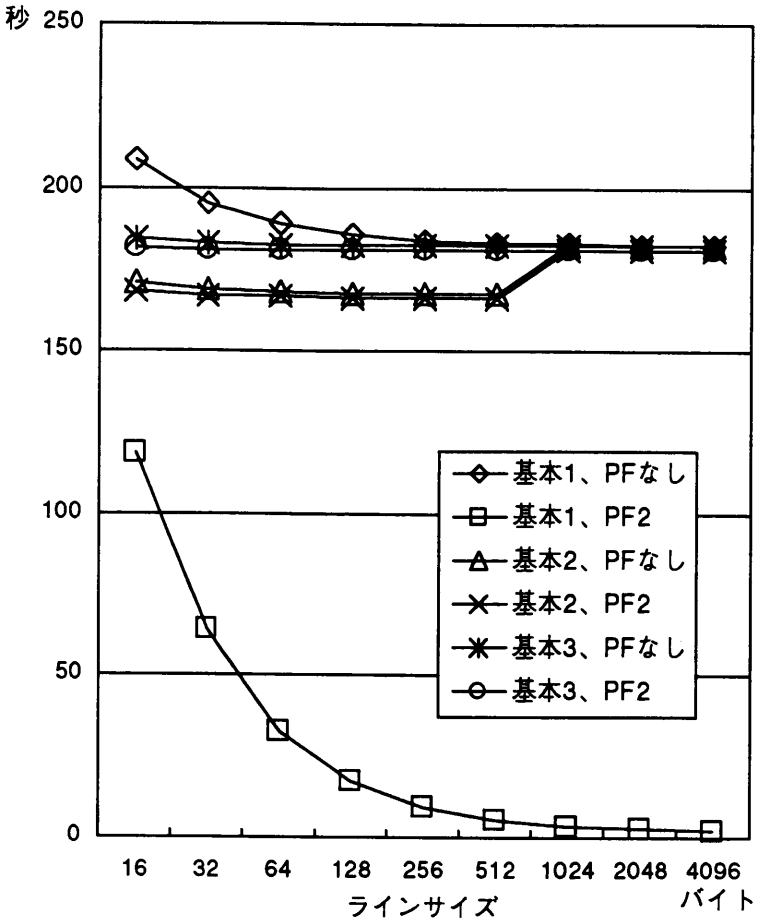


図5-23 一貫性管管理式についての
ラインサイズと通信接続時間
ーサーバ上のスケジュール管理ー

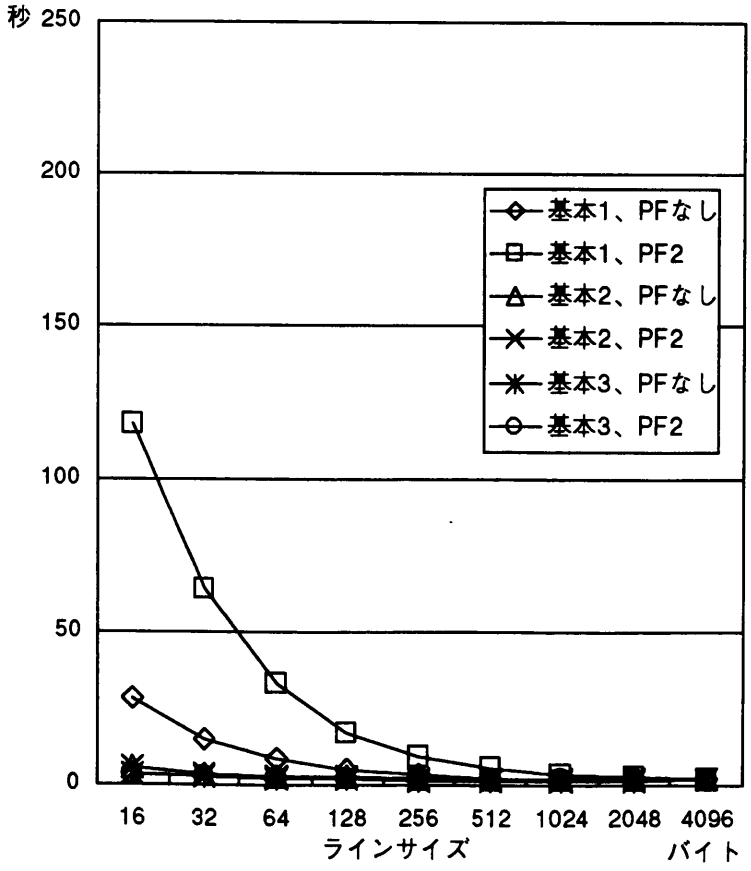


図5-24 一貫性管理方式についての
ラインサイズとデータ伝送時間
ーサーバ上のスケジュール管理ー

総実行時間は、できるだけプリフェッチをおこなうPF2の場合が最も少なくなることがわかる。MMMの場合ラインサイズによる差の割合が少なく見えるのは、画面を参照している時間や書込の時間の占める割合が大きいためである。

(b)図5-14は、モバイル端末側のラインサイズと通信接続時間の関係を示す。MMMのプリフェッチなしと1ラインプリフェッチするPF1の時間が、一括伝送に比較し大きくなっている。これは全ライン転送が終了しないため最後まで通信接続が切断できないことによる。通信の空き時間を利用して、できる限りプリフェッチするPF2の場合、ラインサイズが大きくなるに従い時間が少なくなるのは、実行の途中で全データの書き移しが終了し、通信接続を切断するためである。通信料が接続時間による時間課金の場合には、PF2の方式が有利であることがわかる。

(c)図5-15は、モバイル端末側のラインサイズとデータ伝送時間の関係を示す。実際に必要とするデータのみを伝送するプリフェッチをおこなわない場合が最もデータ伝送時間が少なくなり、さらにラインサイズが128バイトから512バイトの近辺で最小になることがわかる。通信料がデータ量による従量課金の場合にはこの方式を選択するのが有利であることがわかる。

(d) 図5-17と図5-18のサーバ側の通信接続時間とデータ伝送時間は、モバイル端末側の傾向とは異なっている。サーバ側では通信のほとんどがステータス情報の伝送のため通信オーバーヘッドの占める割合が大きく、時間はラインサイズが小さくなるにつれ、大きな値になっている。データの書き戻し時間は、ブロックサイズを16バイトに固定しているため、ラインサイズによらずほぼ一定であり、その占める割合は小さい。初期状態においてデータはすべてサーバ側にあるため、サーバ側でのデータ伝送は、モバイル端末側で変更のあったブロックだけとなる。

5.4.5 一貫性管理方式についての評価

図5-19～図5-24は、基本MMMの管理方式1、基本MMM管理方式2、基本MMM管理方式3の各管理方式に関し、モバイル端末とサーバ上でのスケジュール管理のシミュレーションによる、ラインサイズと総実行時間、通信接続時間およびデータ伝送時間の関係を示す。スケジュール管理のアプリケーションは、5.4.4で用いたモバイル端末と

サーバとで相互にアプリケーションを動作させるシナリオに対して、サーバ上でリード動作に加え、ライト動作をおこなったものである。図において、基本1、基本2、基本3はそれぞれ基本MMM管理方式1、2、3を示し、PFなしは、プリフェッチなし、PF2は、可能な限りプリフェッチする方式を示す。つぎに評価結果を示す。

(a) 図5-19のモバイル端末上の総実行時間では、プリフェッチしないグループとプリフェッチするグループに分かれており、プリフェッチすることにより総実行時間が少なくなることがわかる。ラインサイズが減少すると基本MMM管理方式1の場合、他より時間が増加している。これは基本MMM管理方式1の場合、アクセス権が片方のみに与えられるため、モバイル端末とサーバとで交互に使用する場合、データ転送は、1回ラインを転送すれば次は変更のあったブロックのみの転送でよいが、アクセス権については、アクセス毎に転送の必要があるため、ラインサイズが小さくなると、このステータス情報転送のオーバーヘッドが増大するためである。

(b) 図5-20のモバイル端末上の通信接続時間では、通信が途中で切断できる場合と、切断できない場合に分かれ、基本MMM管理方式1の、できる限りプリフェッチをおこなうPF2のケースでは、途中でデータ伝送を完了し通信接続が切断できるため、ラインサイズが増大するにつれ通信接続時間が減少する。基本MMM管理方式2と基本MMM管理方式3では、通信接続を途中で切らない制御としている。これは、切断後も通信が必要になる場合があり、再接続には呼設定時間がかかるためである。

(c) 図5-21のモバイル端末上のデータ伝送時間は、プリフェッチをする場合としない場合とでグループに分かれていることがわかる。プリフェッチを行うことにより、ラインサイズが減少するに従い、オーバーヘッドが急速に増大する。これはライト権またはリード権をモバイル端末側に転送するための時間である。メモリ管理方式の違いによる比較をすると、プリフェッチする場合としない場合の両者とも、基本MMM管理方式1、基本MMM管理方式2、基本MMM管理方式3の順にデータ伝送時間が少なくなることがわかる。

(d) 図5-22は、サーバ上の総実行時間で、基本MMM管理方式1のプリフェッチなし

の場合、ラインサイズが減少するに従い時間が増大するのは、アクセス権の転送のための、オーバーヘッド時間が増大するためである。その他の場合は、ラインサイズによらずほとんど時間は同じである。

(e) 図5-23は、サーバ上の通信接続時間であり、基本MMM管理方式1のプリフェッチを行う場合には、途中ですべてのデータを伝送し、通信接続の切断をするため、ラインサイズの増加するにつれ通信接続時間は減少する。基本MMM管理方式2の場合、512バイトと1024バイトの部分で段差が出ている。これは、サーバ側の場合、初期データがもともとサーバ側にあるため、リードの場合には、モバイル端末が変更したブロックをリードするまで通信接続の必要はなく、ライトの場合には、モバイル端末がライトしたラインに対してライトするまで通信接続を開始する必要がないため、サーバ側では、使用を開始してからこれらの条件が成立するまで通信接続が行われないためである。ブロックサイズが16バイトと固定のため、リードの場合の条件はラインサイズによらないが、ライトの場合にはラインサイズが小さい方が、モバイル端末がライトしたラインにライトする確率が小さくなるため、それだけ通信接続の開始時間が遅くなり、通信接続時間が減少することによる。

(f) 図5-24のデータ伝送時間では、基本MMM管理方式1の場合、アクセス権の転送が多いため、ラインサイズが減少するに従いオーバーヘッドによる時間が増大する。基本MMM管理方式2と基本MMM管理方式3では、データが最初にサーバ側にある設定のため、アクセス権の転送はライトでのみで発生し、このため実際のデータ転送を合わせても、ひじょうに小さな値となっていることがわかる。

(g) 以上比較したように、基本MMM管理方式1では、ラインサイズが減少するとアクセス権の転送によるオーバーヘッドが増大するが、基本MMM管理方式2と基本MMM管理方式3では、この点に関しては非常に効率が良いことがわかる。しかしながら4.4項で述べたように、それぞれ利害得失があるため、アプリケーションにより使い分けが必要である。

5.5 結言

以上、モバイルコンピューティングシステム向きメモリ管理方式である、基本MMMの提案とその評価について述べた。MMMは、分散共有メモリの側面を持つが、共通メモリによる共有方式、低速度の通信路に合わせたラインサイズの選択が可能、データ伝送量を減らすためのブロック単位の書き戻し制御,通信路の切断への対応や非接続での利用等に関し、従来のものとは異なっている。本方式により、モバイル端末とサーバとの間で共通データの同期が可能となり、アプリケーションの構築では複雑な通信手順が不要となり、実行時間の短縮，通信の効率化を図ることができ、さらにモバイル端末からサーバのリソースの利用が容易となることを示した。通信の効率化は、必要とするデータのみを伝送することの他に、モバイル端末の利用形態を利用したプリフェッチ方式の提案により、通信と他の動作を並行動作させることで、さらに大きな効果が出ることを示した。

つぎの課題は、同一メモリ領域を共有する複数モバイル端末環境への拡張である。これまでの提案の基本MMMは、モバイル端末とサーバとで同一時間では1対1に対応させる方式であり、同一データを他のモバイル端末が使用する場合には、直列化して使用する必要がある。同時に複数端末で使用可能とするためには、MMMの拡張が必要であり、次章でこの拡張について述べる。

6. モバイルコンピューティング環境向きメモリ管理方式の拡張

6.1 緒言

4章では、基本MMMの3種のメモリ管理方式について述べたが、これらの方式は、同一タイミングでは、1台のモバイル端末と1台のサーバとの間で共通メモリ空間を共有する方式である。複数のモバイル端末とサーバとでメモリ空間を共有するためには、サーバ上でモバイル端末毎に異なるメモリ空間を使用するか、時間を分けてシーケンシャルに使用することが必要である。この章では、基本MMMを拡張し共通空間を複数のモバイル端末で共有可能な、拡張MMMの提案をおこなう。

6.2 拡張MMMのアーキテクチャ

6.2.1 概要

拡張MMMは、複数端末とサーバとが同一メモリ空間を共有する方式である。メモリの一貫性管理方式の違いにより、拡張MMM管理方式1と、拡張MMM管理方式2の2つの方式がある。拡張MMM管理方式1では、常に最新のメモリの内容が参照されるように制御される。拡張MMM管理方式2では、メモリを参照する場合、必ずしも最新の内容でない場合がある。最新の内容にするには、参照する直前に同期命令を実行することが必要である。

通常の状態では、端末側はリード権、サーバ側はライト権が与えられる。リード権はメモリリードだけが可能、ライト権はメモリリードおよびライトが可能である。ライト権は同一時間内では、ただ1つのラインにのみ与えられ、他の共通ラインはすべてリード権となる。端末側でライトを行う場合には、ライト権を取得してからライトをおこなう。リード権のある端末でライトをおこなうと、ライト権取得のための割込が発生し、サーバと通信をおこなう。ライト権がサーバにあれば、ただちにアクセス権を取得し、ライト権が他の端末にあれば戻るまで待ち、戻った時点でライト権を取得し、その後ライトを行う。端末が取得したライト権は、一続きのライトが終了後ただちにサーバに戻される。拡張MMMのメモリ制御は、基本MMMで使用するメモリ制御ステータスメモ

リに加えて、新たにサーバ側に設けるライン管理テーブルにより行う。ライン管理テーブルは、各モバイル端末の共通メモリの、各ラインに対応したエントリを持ち、ラインステータスの写しを保持する。

モバイル端末のライト権の保持時間は、可能な限り最短とする必要がある。これは他のモバイル端末がライトを要求したときに、速やかにライト権を得られるようにするためである。このため共通メモリへのライトは、原則として連続したライト命令で行う。アプリケーションによっては、ループ制御でライトを実行する必要がある可能性もあるため、ライトとライトの間隔を一定の決められた時間内とし、この時間を超えた場合には、ライト権をサーバに返還する制御を行う。

6.2.2 メモリ制御ステータスメモリとライン管理テーブル

サーバ及び各モバイル端末は、基本MMMと同じメモリ制御ステータスメモリを使用する。拡張MMMでは、これに加えて、サーバに各端末のメモリ制御ステータスの写し

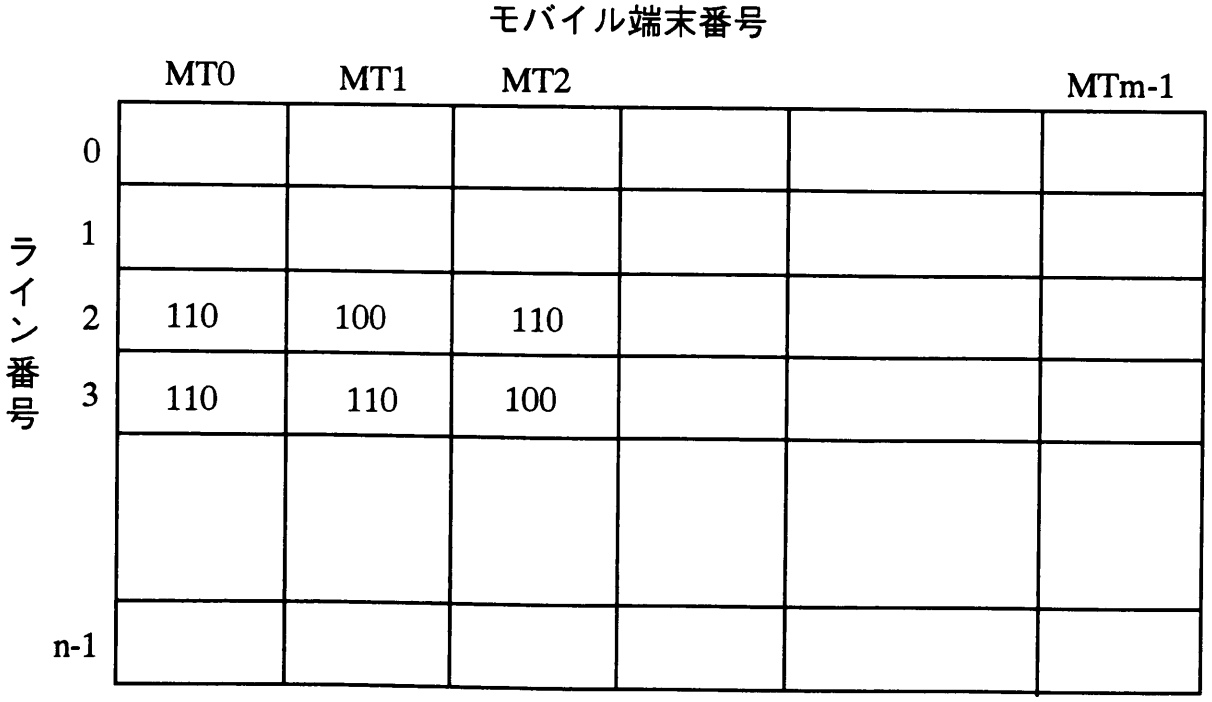


図 6-1 ライン管理テーブル

を保持するライン管理テーブルが設けられる。図6-1にライン管理テーブルを示す。図において、各行が端末のメモリの共通ラインに対応し、各列が各モバイル端末に対応する。各エントリは、モバイル端末のラインステータスの写しを保持し、有効ビット(V)、コピービット(C)、変更ビット(D)の3ビットで構成される。モバイル端末のメモリ制御ステータスの変更ビットは、ラインが複数のブロックで構成される場合、複数のDビットで構成されるが、これらの論理和をとり、1ビットに変換したものをライン管理テーブルのDビットに対応させる。ライン管理テーブルは、サーバの主メモリ領域に置かれる。

6.2.3 拡張MMM管理方式1のメモリ制御

拡張MMM管理方式1では、メモリはアクセス時に常に最新となるように管理される。表6-1にメモリ制御ステータスフィールドで示されるメモリのラインステータスを示す。

表6-1 拡張メモリ管理方式1のメモリステータス

サーバ

VCD	ラインステータス
0--	無効
100	アクセス権，最新，モバイル端末側と一致
101	アクセス権，最新，モバイル端末側と不一致
110	リード権，最新，モバイル端末側と一致
111	リード権，最新でない，モバイル端末側が最新

モバイル端末

VCD	ラインステータス
0--	無効
100	アクセス権，最新，サーバ側と一致
101	アクセス権，最新，サーバ側と不一致
110	リード権，最新，サーバ側と一致
111	リード権，最新でない，サーバ側が最新

(1) モバイル端末でのライト制御

端末でライトが行われると、まずメモリ制御ステータスが調べられる。ステータスがリード権の場合には、メモリ割込を発生させ、サーバにライト権取得の要求をおこなう。サーバ側では、モバイル端末からの要求に対し、ライト権を他の端末に与えている場合には、戻るまで待つ。ライト権が戻るかすでに自身にある場合には、最初にモバイル端末のラインの最新化を行う。モバイル端末のメモリ制御ステータスのDビットを調べ、セットされている場合には、サーバ側のDビットを調べ、Dビットが1にセットされているブロックをモバイル端末に書き写し、モバイル端末の対応するラインのDビットをすべて0にリセットし、同時にライン管理テーブルの対応する端末のDビットを0にリセットするとともに、他の端末のDビットを調べ、すべて0にリセットされていれば、サーバ側の対応するラインのDビットをすべて0にリセットし、いずれか1にセットされていればそのままとし、サーバ側をリード権に、端末側をライト権に設定する。端末側がライト権を取得すると、ライト動作に移り、ライン管理テーブル上で、他のモバイル端末のラインステータスの写しがあるかどうか調べ、写しがあり、Dビットがすでに1にセットされていれば、その端末とは通信せず、写しがあり、Dビットが0にリセットされていれば、その端末と通信しメモリ制御ステータスの対応するラインのすべてのブロックのDビットを1にセットし、同時にその端末のライン管理テーブルのDビットも1にセットする。この処理を、写しを持っているすべてのモバイル端末について行う。要求のあった端末のライン管理テーブルについてもDビットを1にセットする。この後サーバの対応するブロックのDビットを1にセットし、端末ではライト命令を実行し、対応するDビットを1にセットする。さらにライト命令の実行と同時に、ライト権返還リクエストのタイマーを起動する。すでにタイマーが起動されていれば、タイマーをリセットし再起動する。タイマーのタイムアウト時間は約0.5秒程度とする。この時間は、一連の意味ある動作による共通領域へのライト動作の、ライト命令を実行する最大の間隔以上に設定される。一連の意味ある動作とは、スケジュール管理であれば、変更箇所を一旦非共通領域のバッファに書き込み、入力値を確認後、変更指示により共通領域に書き込む動作、銀行預金の引き出しのような動作であれば、引き出したい金額を入力し

た後、確認ボタンを押されたことにより共通領域に書き込みを行うという動作を示し、アプリケーションの組み方によるが、ほぼ連続したライト命令の実行が可能である。

メモリ制御ステータスがライト権の場合には、割込を発生させ、前記のリード権の場合のライト権取得以降のライト動作の処理と同じ処理をおこなう。端末側ではライト命令を実行し、ライト権変換リクエストタイマーを再起動する。

ライト権返還リクエストのタイマーがタイムアウトすると、メモリ割込を発生させ、モバイル端末側のライト権をサーバ側に戻す処理を行う。これは、端末側ではDビットが1にセットされているブロックをサーバ側に書き戻し、端末側のメモリ制御ステータスをリード権にセットし、Dビットを0にリセットする。サーバ側ではメモリ制御ステータスをライト権に戻し、ライン管理テーブルを調べ、対応する端末のDビットを0にリセットすると共に、他の端末の対応するラインのDビットを調べ、すべて0にリセットされていれば、サーバ側のラインのDビットをすべて0にリセットする。

サーバ側のDビットは、いずれかのモバイル端末の対応するラインのDビットが、1にセットされていることで、1にセットされ、すべてのモバイル端末の対応するラインのDビットが、0にリセットされた時点で、0にリセットされる。すなわちサーバ側のDビットの0は、対応するラインの内容がすべて一致していることを示し、サーバ側のDビットの1は、いずれかのモバイル端末のラインに最新でないものがあることを示す。

(2)モバイル端末でのリード

モバイル端末でリードが行われるとメモリ制御ステータスが調べられる。リード権の場合にはさらにDビットが調べられる。Dビットが0にリセットされていればリードが行われる。Dビットが1にセットされている場合には、ラインを最新化するための割込を発生させる。モバイル端末は、サーバ側のメモリ制御ステータスのDビットが1にセットされているブロックをサーバからモバイル端末に書き写し、Dビットを0にリセットしてリードを実行し、サーバ側では、ライン管理テーブル上の対応する端末のDビットを0にリセットするとともに、他の端末のDビットを調べ、すべて0にリセットされていれば、サーバのメモリ制御ステータスの対応するラインのDビットをすべて0にリセットする。ライト権の場合には、そのままリードが行われる。

表6-2 拡張メモリ管理方式2のメモリステータス

サーバ

VCD	ラインステータス
0--	無効
100	アクセス権，最新，モバイル端末側と一致
101	アクセス権，最新，モバイル端末側と不一致
110	リード権，最新，モバイル端末側と一致
111	リード権，最新でない，モバイル端末側が最新

モバイル端末

VCD	ラインステータス
0--	無効
100	アクセス権，最新，サーバ側と一致
101	アクセス権，最新，サーバ側と不一致
110	リード権，ノーマルモード：最新でない可能性がある 同期モード:最新，サーバ側と一致
111	リード権，ノーマルモード：最新でない可能性がある 同期モード：最新でない，サーバ側が最新

(3) サーバでのライト

サーバでライトが行われるとメモリ制御ステータスが調べられる。リード権の場合にはライト権が戻るまで待つ。ライト権が戻るか、すでに自身にある場合には、他のモバイル端末のラインステータスの写しがあるかどうかをライン管理テーブルで調べる。写しがあり、Dビットが1にセットされていれば、その端末のDビットはすでに1にセットされていることを示すため通信はしない。写しがあり、Dビットが0にリセットされていれば、その端末と通信し、対応するラインのメモリ制御ステータスのすべてのブロックのDビットを1にセットし、同時にその端末のライン管理テーブルのDビットも1にセットする。この処理を，写しを持っているすべてのモバイル端末について行う。この後サーバではライトを実行し、対応するブロックのDビットを1にセットする。

(4) サーバでのリード

サーバでリードが行われるとメモリ制御ステータスが調べられる。リード権の場合にはライト権が戻るまで待つ（リード権の場合にはDビットが1にセットされており、ラインの内容は最新ではない)。ライト権が戻るか、すでに自身にある場合には、リードを実行する。

6.2.4 拡張MMM管理方式2のメモリ制御

拡張MMM管理方式2では、ラインステータスがライト権の場合には、メモリの内容が最新となるように制御されるが、リード権の場合には必ずしも最新の内容とはならない。表6-2にメモリ制御ステータスフィールドで示されるメモリのラインステータスを示す。リード権には、ノーマルモードと同期モードがあり、ノーマルモードでは、内容は必ずしも最新ではないが、同期モードでは最新となるように管理される。同期モードは同期開始命令でセットされ、同期終了命令でリセットされる。

(1) モバイル端末でのライト

モバイル端末側の、ライトの場合の動作は、割込を発生させ、サーバ側のライン管理テーブルで他の端末にラインの写しがあるかないかを調べ、他の端末にラインの写しがある場合、ライン管理テーブルの対応するDビットを1にセットするが、他の端末と通信を行わず、したがって他の端末のメモリ制御ステータスの更新を行わないことを除き、拡張MMM管理方式1のメモリ制御と全く同じ動作を行う。

(2) モバイル端末でのリード

リード権のノーマルモード、またはライト権の場合には、そのままリードする。同期モードの場合には、拡張MMM管理方式1の端末のリードの場合と全く同じ動作をおこなう。

(3) サーバのライト

他の端末にラインの写しがあるかないかをライン管理テーブルで調べ、他の端末にラインの写しがある場合、ライン管理テーブルの対応するDビットを1にセットするが、

他の端末と通信を行わず、したがって他の端末のメモリ制御ステータスの更新を行わないことを除き、拡張MMM管理方式1のメモリ制御と全く同じ動作を行う。

(4) サーバのリード

拡張MMM管理方式1のサーバのリードの場合と全く同じ動作を行う。サーバでは、常にライト権でリードを行うため最新の内容が読まれる。

(5) 同期制御

同期制御のため、同期開始命令と同期終了命令が用意される。モバイル端末で同期開始命令が実行されると同期モードとなり、割込を発生させサーバと通信をおこない、ラインステータスの最新化要求をおこなう。サーバ側では、要求のあったモバイル端末を同期モードに設定し、ライン管理テーブルに対して、要求のあった端末に対応するラインの写しを調べ、Dビットが1にセットされているラインがあれば、モバイル端末に知らせ、モバイル端末側ではメモリ制御ステータスの対応するラインのDビットをすべて1にセットすることにより、端末側のメモリ制御ステータスを最新化する。この後サーバは、同期モードとなったモバイル端末に対しては、拡張MMM管理方式1と同一のメモリ制御をおこなう。これはサーバまたは他のモバイル端末でライトが実行されたときに、同期モードのモバイル端末にラインの写しがある場合には、端末側までDビットの最新化をおこなう。モバイル端末で同期終了命令が実行されると、割込を発生させサーバに同期モードの終了を通知する。

6.2.5 排他制御

複数のモバイル端末が共有メモリを使用しているとき、共有データを排他的に使用するためには排他制御が必要であり、排他制御のため、占有命令を設ける。占有命令は指定されたアドレスの内容を調べ、0の場合には条件コードを0にセットすると共に、指定されたレジスタの内容を、指定されたアドレスにストアし、0でない場合には条件コードを1にセットする。すなわち指定されたアドレスの内容がすでに0でない場合には、共有データが他で使用されていることを示し、0の場合には、他で使用されていないため、

自身が排他的に使用することを示すため、0以外の値をセットする。

プログラムが共有データを排他的に使用するためには、占有命令を実行し、占有権(排他使用権)を得た後に、共有データを使用する。占有権が他にあれば占有権が返還されるまで待つ。共有データの使用後は占有権を解除する。

6.3 拡張MMMの応用事例

拡張MMMが有効なアプリケーションの事例を次に示す。

(a) 会議室予約システム

会議室の予約希望者は、会議室予約表を見て予約状況を調べ、自分の要求に合った会議室が空いていれば、そこを予約するシステムである。ここで必要となる基本機能は、同時に複数の予約の要求があり、場所と時間が同一の予約要求があるときの調停方法であり、拡張MMMの機能により実現可能である。

(b) 座席予約システム

必要となる基本機能は会議室予約システムと同様である。

(c) 図書貸出予約システム

在庫している書籍の検索と貸出状況を表示し、貸し出されている場合には貸出の予約が可能なシステム。

(d) 在庫管理システム

商品の入荷、出荷および在庫を管理するシステム。

6.4 拡張MMMのシミュレーションによる評価

アプリケーションの例として次の会議室予約システムを作成し、拡張MMMの評価を行う。

(a) 会議室予約システムの概要

5つの会議室があり、予約可能時間は8:00から20:00までとし、予約単位は30分とし、4週間先まで予約可能とする。モバイル端末より会議室予約システムを呼び出すと、最初に初期画面が表示される。初期画面には、予約可能日の一覧が表示されるので、予約希望日を指定すると、指定した日の5つの会議室の予約状況の一覧が表

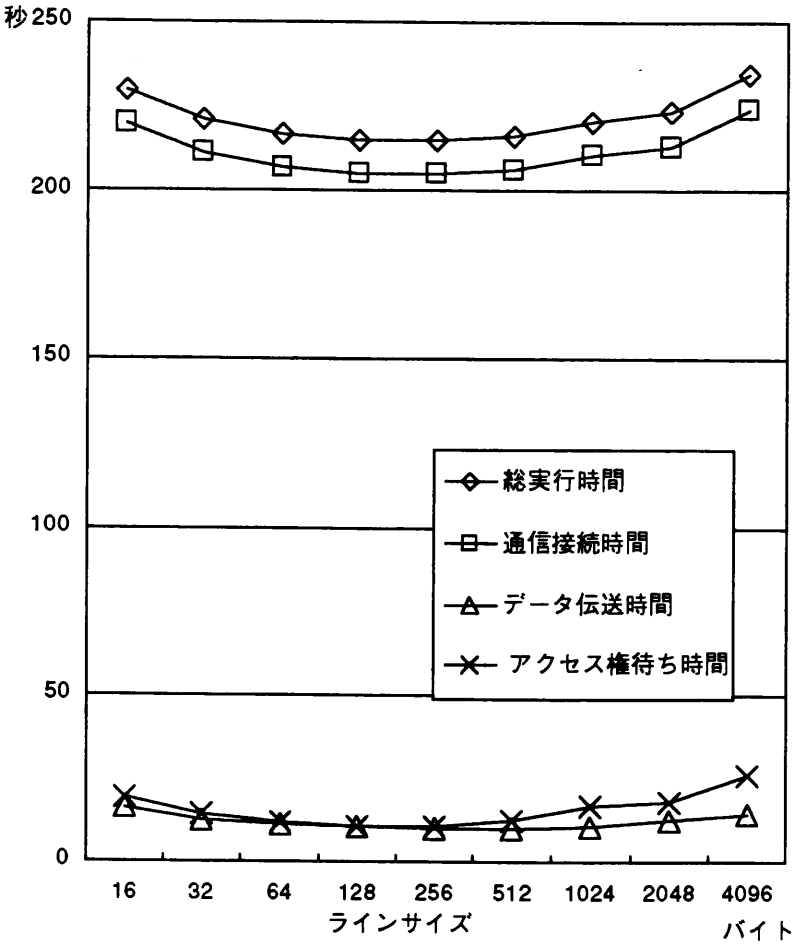


図 6-2 －拡張 MMM 管理方式 1－
モバイル端末 1 ラインサイズと総実行時間、
通信接続時間、データ伝送時間、アクセス権待ち時間

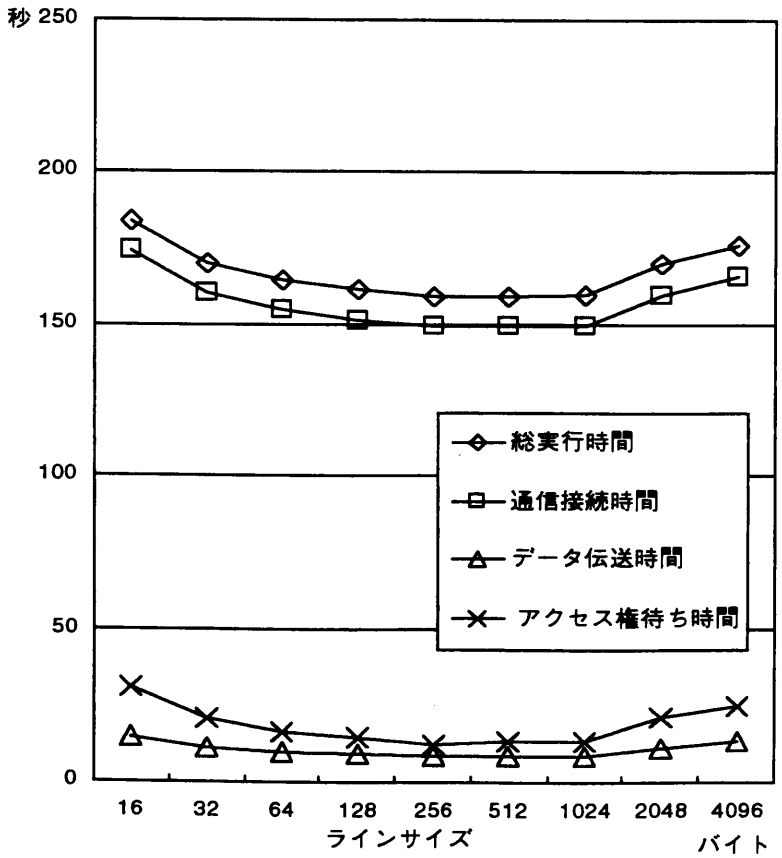


図 6-3 －拡張 MMM 管理方式 1－
モバイル端末 2 ラインサイズと総実行時間、
通信接続時間、データ伝送時間、アクセス権待ち時間

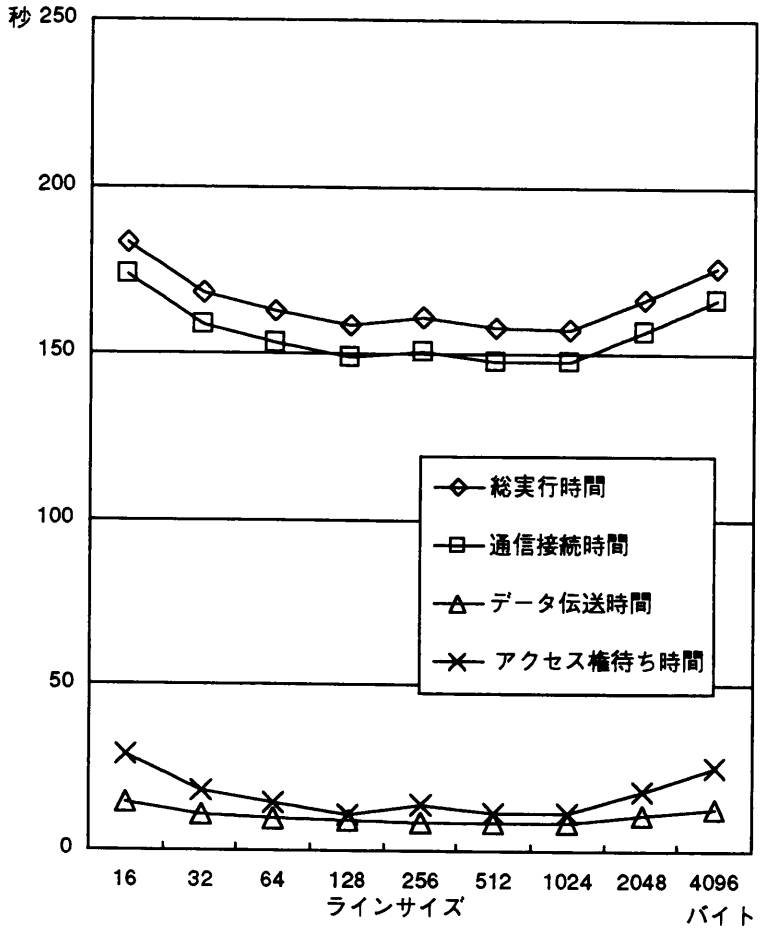


図 6-4 －拡張 MMM 管理方式 1－
モバイル端末 3 ラインサイズと総実行時間、
通信接続時間、データ伝送時間、アクセス権待ち時間

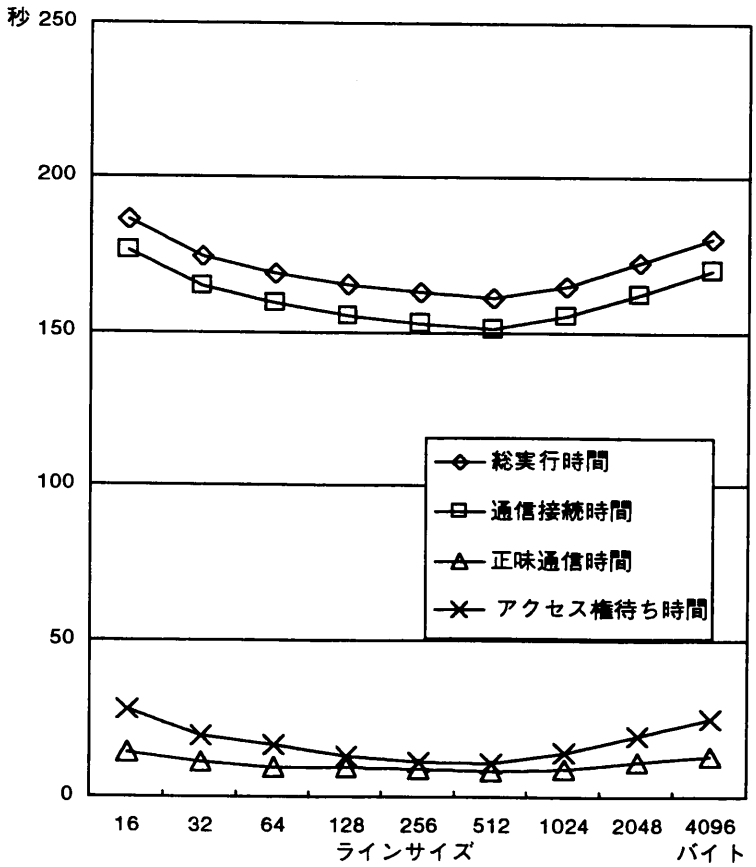


図 6-5 －拡張 MMM 管理方式 1－
モバイル端末 4 ラインサイズと総実行時間、
通信接続時間、データ伝送時間、アクセス権待ち時間

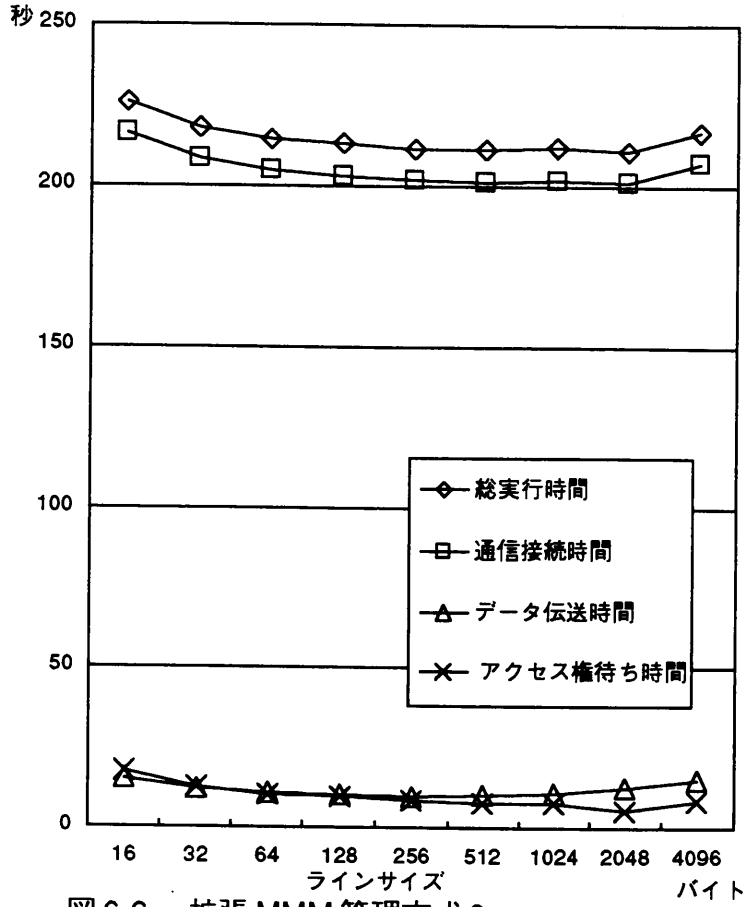


図 6-6 －拡張 MMM 管理方式 2－
モバイル端末 1 ラインサイズと総実行時間、
通信接続時間、データ伝送時間、アクセス権待ち時間

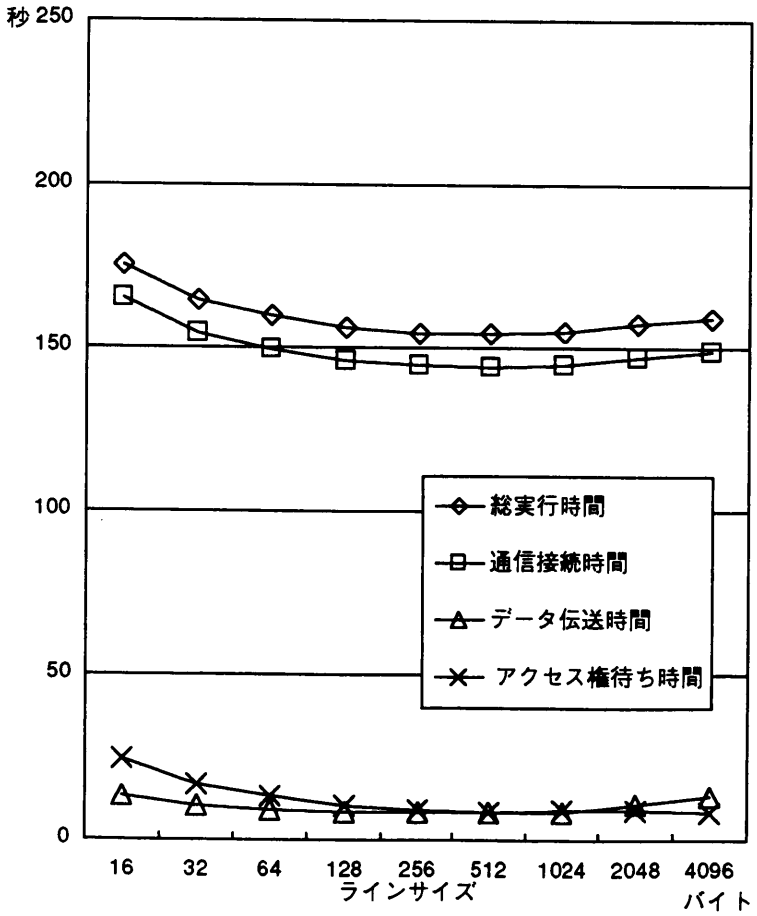


図 6-7－拡張 MMM 管理方式 2－
モバイル端末 2 ラインサイズと総実行時間、
通信接続時間、データ伝送時間、アクセス権待ち時間

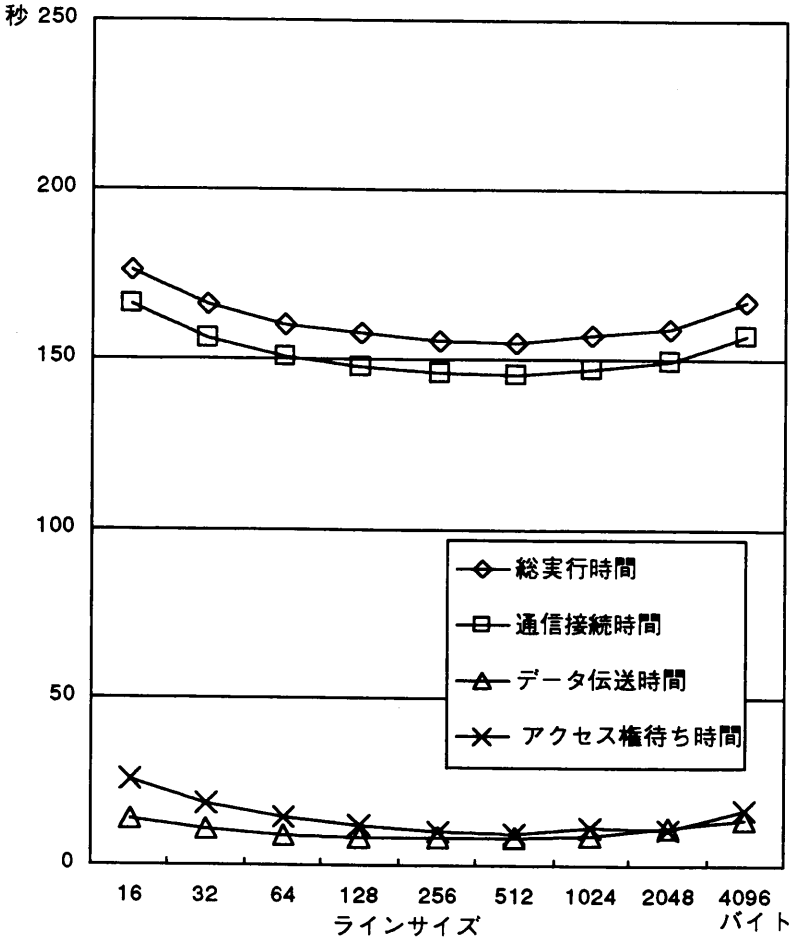


図6-8 －拡張MMM管理方式2－
モバイル端末3 ラインサイズと総実行時間、
通信接続時間、データ伝送時間、アクセス権待ち時間

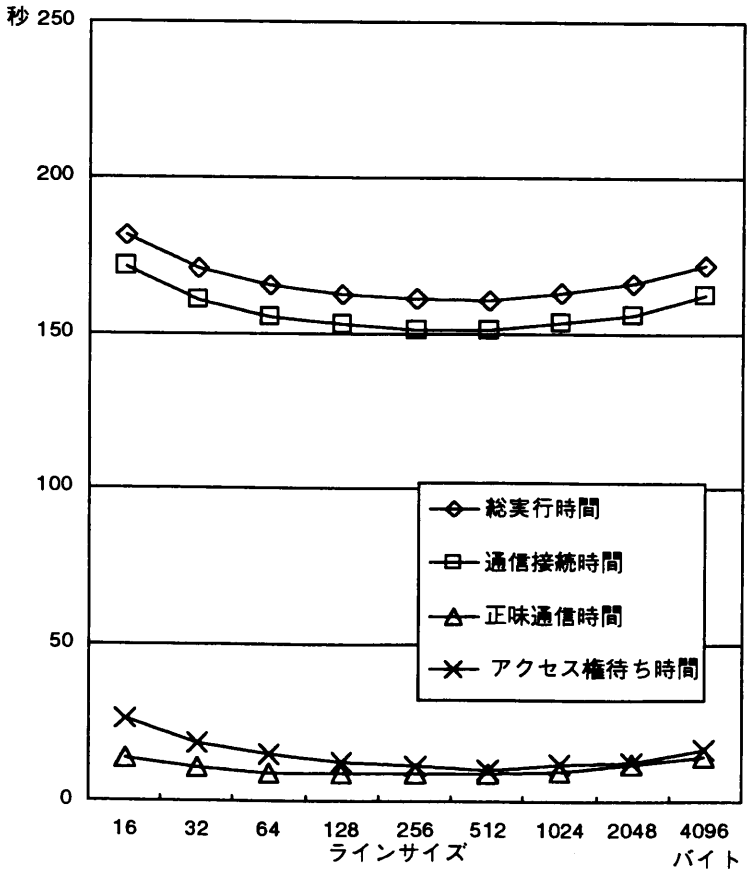


図6-9 －拡張MMM管理方式2－
モバイル端末4 ラインサイズと総実行時間、
通信接続時間、データ伝送時間、アクセス権待ち時間

示される。予約を希望する会議室名、会議時間、予約者名及び会議名を入力し、予約設定ボタンを押す。アプリケーションは、まず予約テーブルを排他使用するために占有命令を実行する。排他使用権を得ると、指定された会議室と会議時間がすでに予約されているかどうか調べ、すでに予約されていれば、他ですでに予約済を表示するとともに、最新の予約状況を再表示する。他で予約されていなければ、指示された予約を行い、最新の予約状況を再表示する。端末使用者は、予約できなかった場合には最新の予約状況を見て、再度予約を行う。

(b) 拡張MMMの会議室予約システムによる評価

4台のモバイル端末が1つの会議室を4週間にわたり予約するものとし、競合条件を作るため各モバイル端末に対して同時に予約作業を開始させる。最初に、週に1から2回の同一曜日の会議を4週間にわたり予約する。この後約10秒後に各モバイル端末は1から2回の会議の予約をランダムに行い、一部ではすでに他で予約済の会議時間を指定し、予約不可の通知を受け、時間を変更後再予約する。以上のシナリオにより、各モバイル端末の総実行時間、通信接続時間、データ伝送時間、アクセス権(ライト権)待ち時間を、拡張MMM管理方式1と、拡張MMM管理方式2の各メモリ管理方式について評価する。

最初に、拡張MMM管理方式1について、図6-2～図6-5に4台のモバイル端末が、前記シナリオに基づいて会議室の予約を行うシミュレーションの結果を示す。モバイル端末1の総実行時間および通信接続時間が、他の端末より長いが、これは予約する会議の数が単純に他より多いためである。4台のモバイル端末をほぼ同時に使用し、競合状況を作っているが特に異常な時間はなく、良好に動作していることを示す。

図6-6～図6-9は、拡張MMM管理方式2について、前記と同じシナリオによるシミュレーションの結果である。ほぼ同様の傾向の結果が得られているが、拡張MMM管理方式1と比較し、256バイトのラインサイズの場合に総実行時間では約2%、アクセス権(ライト権)待ち時間では約20%減少している。データ伝送時間については、ほとんど差はない。これは、拡張MMM管理方式2が、メモリの内容の同期をとるための通信が拡張MMM管理方式1に比較して少なく、このためサーバとの通信で、モバイル端末間の競合がより少なくなるためである。拡張MMM管理方式2の方が、端末間のメモリアク

セスのための競合は減るが、逆にその代償として、端末ユーザーにとって最新の予約状況を見る時間が若干遅くなる。このため、他の端末ユーザーの状況を知る時間が遅れ、予約が競合する機会が増大する可能性はあるが、ごく短時間であり、実用的には問題にならないと考えられる。

6.5 結言

本章では、第4章で提案したモバイルコンピューティング環境向き共通メモリ管理方式の基本MMMを拡張し、複数のモバイル端末とサーバとで同一のメモリ空間を共有可能にする拡張MMMを提案し、その評価を行った。基本MMMでは、モバイル端末とサーバとは共通メモリの管理に関し対称であったが、拡張MMMでは複数のモバイル端末とサーバとは非対称となる。さらにモバイルコンピューティング環境下、すなわち通信接続が切断される可能性のある環境下で、複数のモバイル端末がサーバと同一メモリ空間を共有することを考慮し、共通メモリを変更する権利であるライト権は、常時はサーバ側にあり、モバイル端末はそれが必要になったときのみ取得し、必要な一連のライトの終了後はただちにライト権をサーバ側に戻す制御とする方式とした。一般の共有メモリ方式で採用されているのと同様の、同期命令を導入することにより、複数のモバイル端末が、会議室予約システムを同時に、それぞれの分散制御で使用可能であることを示した。

7. 結論

本研究では、モバイルコンピューティング環境における各種課題の中で、アプリケーション構築の容易化、サーバとモバイル端末間でのデータの同期の容易化、および通信の効率化等を目的として、モバイルコンピューティング環境向きメモリ管理方式(MMM)を提案し、アーキテクチャの詳細検討、具体的適用方法、シミュレーションによる評価を行った。

第1章では、本研究の背景及び目的と、本研究の概要について述べた。

第2章では、モバイルコンピューティング環境を構成するモバイル端末と、無線通信システムについて概観し、各種のモバイル端末の特徴と課題、各種の無線通信システムの概要と特徴を述べ、モバイルコンピューティング環境における課題は、モバイル端末については大きさ、質量、バッテリーの持続時間、無線通信については、データ伝送が低速であること、通信の切断の可能性があること、広域通信についてはコストが高いことを述べた。

第3章では、モバイルコンピューティング技術に関する従来研究について概観した。ハードウエア構成方式では、バッテリーの電力消費を節減する方法や、モバイル端末の機能を単純化することで大きさや質量を軽減し電力消費を縮減する端末のハードウェア方式を紹介し、ソフトウェアによる方式では、通信の切断のある環境下でのファイル管理方式やアプリケーションの構築法について述べた。メモリ管理方式は、モバイルコンピュータに限らず、すべてのコンピュータの基本であり、MMMの基礎を成すため、基本的な方式について概観した。

第4章では、本研究で提案するモバイルコンピューティング環境向きメモリ管理方式の中で、基本MMMのアーキテクチャについて述べ、メモリの一貫性管理方式が異なる3種の管理方式を提案した。MMMは、共通メモリ領域をラインと呼ぶ一定の大きさの単位で分割し、内容の一貫性管理をラインで行う。ラインをさらにブロックに分割することによりデータ転送量を減らす。基本MMMでは、モバイル端末とサーバとが1対1の対応で共通メモリの内容の一貫性を管理し、メモリ管理に関しては、モバイル端末とサーバとは全く対等である。3種の管理方式はその特徴により、アプリケーションの種

別に応じて、適切に使い分けられることを示した。

第5章では、MMMと他のメモリ管理方式との差異、モバイルコンピューティングシステムへの適用について述べ、アプリケーションとしてスケジュール管理や住所録等を使用したシミュレーションによる評価を行った。ラインサイズには適切なサイズがあること、ブロックサイズは制御ビットの増加の問題を除くと小さいほうが良いことを示した。モバイルコンピューティング環境でのモバイル端末利用が、インタラクティブの場合が多いことを利用したプリフェッチの導入により、実行時間の短縮ができることや、時間課金または従量課金に応じて、MMMの適切な管理方式の選択により、通信の効率化が図れることを示した。

第6章では、基本MMMを拡張した、拡張MMMの2種の方式について、アーキテクチャとその応用について述べ、シミュレーションにより評価した。基本MMMではモバイル端末とサーバとが1対1の関係であったのに対し、拡張メモリ管理方式では、複数のモバイル端末とサーバとの構成が可能であり、共通メモリ領域の複数端末間の排他制御が可能であることを示した。

なお本研究では、モバイルコンピューティング環境下でのモバイル端末とサーバ向けの、メモリ管理についての基本アーキテクチャを提案した。しかしながら、基本アーキテクチャを実現するためのオペレーティングシステム(OS)との関係も整理する必要があるが、OSも日進月歩で進化しているため、今後の課題である。

1999年以来インターネット接続機能を持つ携帯電話が急速に普及している。これらのデータ通信は、主としてパケット通信方式が採用され、接続時間ではなく通信したデータ量による従量課金方式と、常時接続での利用が可能のため、利用分野が拡大している。近距離通信では、無線LANが事務所や家庭内でのコードレス通信方式として普及が広がっており、さらに1999年に規格化されたBluetoothも各種機器間のコードレス通信方式としての普及が期待されている。一方モバイル端末としてのモバイルコンピュータは、据置型と同一機能を持つノートＰＣと、機能は限定されているがモバイルでの利用に特化した機能を持つパームサイズPCに2極化していくように思われる。パームサイズPCは今後事務所や家庭のメインPCと、または各種コンテンツサーバとの連携機能が進展

するものと考えられる。MMMはこれらモバイルコンピューティング環境の中で有効な技術であると考える。

謝辞

本研究において、直接種々懇切なるご指導とご鞭撻をいただいた静岡大学情報学部助教授 渡辺尚博士に深く感謝申しあげる。また、懇切なるご指導とご鞭撻をいただいた静岡大学情報学部教授 水野忠則博士に深く感謝申し上げる。

本論文をまとめる課程で、種々適切なるご指導とご鞭撻をいただいた、静岡大学情報学部教授 梅谷征雄博士、静岡大学情報学部教授 富樫敦博士、静岡大学情報学部助教授 塩見彰睦博士に深く感謝申し上げる。

また、本研究を進める上で、種々適切なご指導とご鞭撻をいただいた、岩手県立大学ソフトウェア情報学部教授 曽我正和博士、静岡大学情報学部助教授 佐藤文明博士、静岡大学情報学部助手 石原進博士に深く感謝申し上げる。

静岡大学大学院博士課程への社会人入学の機会を与えていただいた前三菱電機株式会社情報システム製作所所長 長沢一嘉氏、三菱電機株式会社情報システム製作所所長 香取和之氏、三菱電機株式会社情報通信システムセンターセンター長 肥田木誠氏に深く感謝申し上げる。

三菱電機株式会社情報システム製作所からの静岡大学社会人学生の先輩として種々助言とご指導をいただいた三菱電機株式会社情報システム製作所 田窪昭夫博士、公立はこだて未来大学システム情報科学部教授 宮西洋太郎博士、三菱電機株式会社情報システム本部 青野正弘博士、浜松大学助教授 飯田登博士に深く感謝申し上げる。

研究を進めるにあたり、ご意見ご討論をいただいた三菱電機株式会社技術研修所吉田幸二氏、三菱電機株式会社通信開発センター 上田尚純氏、株式会社エヌ・ティ・ティ・ドコモマルチメディア研究所 太田賢博士に深く感謝申し上げる。

本研究に直接協力いただいた静岡大学大学院 清水正貴氏、静岡大学大学院 奥田隆弘氏をはじめとして静岡大学渡辺研究室の学生諸氏ならびに静岡大学水野研究室の学生諸氏に深く感謝申し上げる。

最後に、在宅の研究作業に協力を強いた家族に感謝する。

参考文献

[1] M. Weiser, R. Gold, J. S. Brown, "The origins of ubiquitous computing research at PARC in the late 1980s," IBM SYSTEM JOURNAL, Vol. 38, No. 4, pp. 693-696, 1999.

[2] 水野忠則，田窪昭夫，"モーバイルコンピューティング，" 情報処理 , Vol. 36, No. 9, pp. 822-826, September 1995.

[3] 水野忠則，太田賢，"モバイルコンピューティングの現状と将来像，" 電子情報通信学会誌 , Vol. 80, No. 4, pp. 318-323, April 1997.

[4] 移動通信研究会編，"移動通信システムガイド'99 ー陸上移動通信のすべてー，" (株)クリエイト・クルーズ，1999.

[5] 田原康生，歌野孝法，沖中秀夫，丸山辰夫，"IMT-2000 のサービスとシステム要求条件，" 電子情報通信学会誌，Vol. 82, No. 2, pp. 108-115, February 1999.

[6] Robert C. Dixon, "SPREAD SPECTRUM SYSTEMS WITH COMMERCIAL APPLICATIONS Third Edition," John Wiley & Sons, Inc., New York, USA 1994.

[7] ANSI/IEEE Std 802.11, 1999 Edition [ISO/IEC 8802-11: 1999], Information Technology - Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.

[8] IEEE Std 802.11b, 1999, Local and Metropolitan Area Networks, Wireless LAN MAC and PHY specifications: Higher speed Physical Layer (PHY) extension in the 2.4 GHz band.

[9] 相原達, “近距離無線通信規格 Bluetooth,” **bit**, Vol. 32, No. 10, pp. 8-14, October 2000.

[10] Anantha P. Chandrakasan, Samuel Sheng, and Robert W. Brodersen, “Low-Power CMOS Digital Design,” IEEE Journal of Solid-State Circuits, Vol. 27, No. 4, pp. 473-484, April 1992.

[11] Arif Merchant, Benjamin Melamed, Eugen Schenfeld, and Bhaskar Sengupta, “Analysis of a Control Mechanism for a Variable Speed Processor,” IEEE Transactions on Computers, Vol. 45, No. 7, pp. 793-801, July 1996.

[12] 古市実裕, 相原達, 下遠野享, “ノートパソコンにおけるプロセッサ計算速度の自動制御”、マルチメディア、分散、協調とモーバイル(DICOMO'98)シンポジウム、pp. 669-676, July 1998.

[13] David P. Helmbold, Darrell D. E. Long and Bruce Sherrod, “A Dynamic Disk Spin-down Technique for Mobile Computing,” MOBICOM ’96, pp. 130-142, 1996.

[14] 岡村寛之, 土井正, 尾崎俊治, “コンピュータシステムの自動スリープ機能による省電力効果1-再生過程によるモデル化,” 情報処理学会論文誌, Vol.39 No.6, pp. 1858-1869, June 1998.

[15] C. F. Chiasserini, R. R. Rao, “Pulsed Battery Discharge in Communication Devices,” MOBICOM ‘99, pp. 88-95, 1999.

[16] Robin Kravets and P. Krishnan, “Power Management Techniques for Mobile Commu-

nication," MOBICOM '98, pp. 157-168, 1998.

[17] Pravin Bhagwat, Ibrahim Korpeoglue, Chatschik Bisdikian, Chatschik Bisdiskian Mahmoud Naghshineh and Satish K. Tripathi, "BlueSky: A Cordless Networking Solution for Palmtop Computers," MOBICOM '99, pp. 69-76, 1999.

[18] Paul Lettieri, Christina Fragouli, and Mani B. Srivastava, "Low Power Error Control for wireless Links," MOBICOM '97, pp. 139-150, 1997.

[19] Thomas E. Truman, Trevor Pering, Roger Doering and Robert W. Brodersen, "The InfoPad Multimedia Terminal: A Portable Device for Wireless Information Access," IEEE Transactions on Computers, Vol.47, No.10, pp. 1073-1087, 1998.

[20] Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, Phoenix Technologies Ltd., and Toshiba Corporation, "Advanced Configuration and Power Interface Specification," Revision 2.0, July 27, 2000.

[21] Alexander Klaiber, "The Technology Behind Crusoe Processors," Transmeta Corporation, January 2000 (http://www.transmeta.com/).

[22] Alan Jay Smith, "Cache Memories," Computing Surveys, Vol. 14, No. 3, pp. 473-530, 1982.

[23] Josep Torrellas, Chun Xia, and Russell L. Daigle, "Optimizing the Instruction Cache Performance of the Operating System," IEEE Transaction on Computers, Vol. 47, No. 12, pp. 1363-1381, December 1998.

[24] 森真一郎，福島直人，五島正裕，中島浩，富田真治，“Self-Cleanup Cache の提案，” 情報処理学会論文誌，Vol. 38, No.2, pp. 321-331, February 1997.

[25] Sinisa Srbljic, Zvonko G. Vranesic, Michael Stumn, and Leo Budin, “Analytical Prediction of Performance for Cache Coherence Protocols,” IEEE Transaction on Computers, Vol. 46, No. 11, pp. 1155-1173, November 1997.

[26] Francois Bodin, and Andre Seznec, “Skewed Associativity Improves Program Performance and Enhances Predictability,” IEEE Transactions on Computers, Vol. 46, No. 5, pp. 530 544, May 1997.

[27] Vidyadhar Phalke and B. Gopinath, “Compression-Based Program Characterization for Improving Cache Memory Performance,” IEEE Transactions on Computers, Vol. 46, No. 11, pp. 1174-1186, November 1997.

[28]寺澤卓也，井上敬介，黒澤飛斗矢，天野英晴，“シングルマルチプロセッサのためのスヌープキャッシュの検討，” 電子情報通信学会論文誌 D-Ⅰ, Vol. J79-D-Ⅰ, No. 4, pp. 177-187, April 1996.

[29] Hong Wang, Tong Sun, Qing Yang, “Minimizing Area Cost of On-Chip Cache Memories by Caching Address Tags,” IEEE Transactions on Computers, Vol. 46, No. 11, pp. 1187-1201, November 1997.

[30] David A. Patterson, John L. Hennessy, “Computer Architecture A Quantitative Approach Second Edition,” Morgan Kaufmann Publishers, Inc., San Francisco, California, 1996.

[31] 森岡道雄，黒澤憲一，“分散メモリ型マルチプロセッサ用キャッシュ一致保証方式の設計と評価，” 情報処理学会論文誌，Vol. 39, No.4, pp. 1088-1097, April 1998.

[32] Lucien M. Censier and Paul Feautrier, “A New Solution to Coherence Problems in Multicache Systems,” IEEE Transactions on Computers, Vol. C-27, No. 12, pp. 1112-1118, December 1978.

[33] Chun Xia and Josep Torrellas, “Comprehensive Hardware and Software Support for Operating Systems to Exploit MP Memory Hierarchies,” IEEE Transactions on Computers, Vol. 48, No. 5, pp. 494-505, May 1999.

[34] 鈴木健一，大庭信之，中田武男，宮崎晃一郎，小林広明，中村維男，“RICE による２次キャッシュメモリの性能評価，” 電子情報通信学会論文誌 D-Ⅰ，Vol. J80-D-Ⅰ，No. 10, pp. 793-802, October1997.

[35] Daniel Lenoski, James Laudon, Kourosh Gharachorloo, Anoop Gupta, and John Hennessy, “The Directory-Based Cache Coherence Protocol for the DASH Multiprocessor,” 17th ISCA, pp. 148-251, 1990.

[36] Jeffrey Kuskin, David Ofelt, Mark Heinrich, John Heinlein, Richard Simoni, Kourosh Gharachorloo, John Chapin, David Nakahira, Joel Baxter, Mark Horowitz, Anoop Gupta, Mendel Rosenblum, and John Hennessy, “The Stanford FLASH Multiprocessor,” Proc. 21st ISCA, pp. 302-313, 1994.

[37] Tom Lovett and Russell Clapp, “STiNG: A CC-NUMA Computer System for the Commercial Marketplace,” Proc. 23rd ISCA, pp. 308-317, 1996.

[38] Takashi Matsumoto, Katsunobu Nishimura, Tomohiro Kudoh, Kei Hiraki, Hideharu Amano and Hidehiko Tanaka, "Distributed Shared Memory Architecture for JUMP-1 a general-purpose MPP prototype," Proc. IEEE 1996 International Symposium on Parallel Architectures, Algorithms and Networks, pp. 131-137, 1996.

[39] Kai Li, "IVY: A Shared Virtual Memory System for Parallel Computing," Proc. 1988 ICPP, pp. 94-101, 1988.

[40] 市川明弘，小野航，中條拓伯，工藤知宏，天野英晴，"Home Proxy Cache による分散共有メモリの高速化，" 情報処理学会論文誌，Vol. 40, No. 5, pp. 2016-2024, May 1999.

[41] Steven K. Reinhardt, Robert W. Pfile, and David A. Wood, "Hardware Support for Flexible Distributed Shared Memory," IEEE Transaction on Computers, Vol. 47, No. 10, pp. 1056-1072, October 1998.

[42] Mark Heinrich, Vijaraghavan Soundararajan, John Hennessy, and Anoop Gupta, "A Quantitative Analysis of the Performance and Scalability of Distributed Shared Memory Cache Coherence Protocols," IEEE Transaction on Computers, Vol. 48, No.2, pp. 205-217, February 1999.

[43] Frederic T. Cong, Beng-Hong Lim, Ricardo Bianchini, and John Kubiatowicz, "Application Performance on the MIT Alewife Machine," IEEE Computer, pp. 57-64, December 1996.

[44] Yeikuan Chang, and Laxmi N. Bhuyan, "An Efficient Tree Cache Coherence Protocol for Distributed Shared Memory Multiprocessors," IEEE Transactions on Computers,

Vol. 48, No. 3, pp. 352-360, March 1999.

[45] 田中清史，松本尚，平木敬，“軽いハードウェアによる分散共有メモリ機構，” 情報処理学会論文誌，Vol. 40, No. 5, pp. 2025-2036, May 1999.

[46] 斎藤彰一，國枝義敏，大久保英嗣，“広域分散環境に置ける分散共有メモリの実現とその性能評価，” 情報処理学会論文誌，Vol. 40, No. 6, pp. 2563-2572, June 1999.

[47] 安生健一朗，井上浩明，佐藤充，工藤知宏，天野英晴，平木敬，“超並列計算機JUMP-1における分散共有メモリ管理プロセッサMBP-light，” 情報処理学会論文誌，Vol. 39, No. 6, pp. 1632-1643, June1998.

[48] 南里豪志，佐藤周行，島崎眞昭，“分散共有メモリシステム上にソフトウェアによって構築されたキャッシュシステムの静的制御，” 情報処理学会論文誌，Vol. 38, No. 9, pp. 1859-1868, September1997.

[49] 西村克信，工藤知宏，天野英晴，“Pruning Cacheを用いた分散共有メモリのディレクトリ構成法，” 情報処理学会論文誌，Vol. 39, No. 6, pp. 1644-1654, June 1998.

[50] Ravishankar R. lyer and Laxmi N. Bhuyan, “Design and Evaluation of a Switch Cache Architecture for CC-NUMA Multiprocessors,” IEEE Transactions on Computers, Vol. 49, No. 8, pp. 779-797, August 2000.

[51] 岡本一晃，松岡浩司，廣野英雄，横田隆史，佐藤三久，坂井修一，“並列計算機のための同期処理機構とその評価，” 情報処理学会論文誌，Vol. 40, No. 3, pp. 1245-1256, March 1999.

[52] 上原敬太郎，猪原茂和，益田隆司，“分散共有メモリのキャッシュ一貫性制御プロトコルのカスタマイズ，” 情報処理学会論文誌，Vol.38, No. 10, pp. 2040-2052, October 1997.

[53] A．S．タネンバウム，水野他訳，“分散オペレーティングシステム，” プレンティスホール出版，1996.

[54] 平木敬，丹羽純平，松本尚，“分散共有メモリに基づく計算機クラスタ，” 情報処理，Vol.39, No. 11, pp. 1078-1083, November 1998.

[55] Sarita V. Adve, and Kourosh Gharachorloo, “Shared Memory Consistency Models: A Tutorial,” IEEE Computer, Vol.29, No. 12, pp. 66-70, December 1996.

[56] Liviu Iftode, Cezary Dubnicki, Edward W. Felten, and Kai Li, “Improving Release-Consistent Shared Virtual Memory using Automatic Update,” Proc. of the 2nd Inter. Symp. on HPCA, pp. 14-25, 1996.

[57] Pete Keleher, Alan L. Cox, and Willy Zwaenepoel, “Lazy Release Consistency for Software Distributed Shared Memory,” Proc. of the 19th ISCA, pp. 13-21, 1992.

[58] Pete Keleher, Alan L. Cox, Sandhya Dwarkadas and Willy Zwaenepoel, “TreadMarks: Disributed Shared Memory on Standard Workstations and Operating Systems,” Proc. of the 1994 Winter USENIX, pp. 115-131, 1994.

[59] Guang R. Gao and Vivek Sarkar, “Location Consistency - A New Memory Model and Cache Consistency Protocol,” IEEE Transactions on Computers, Vol. 49, No. 8, pp. 798-813, August 2000.

[60] 高橋大介，金田康正，“分散メモリ型並列計算機による円周率の515億桁計算，”情報処理学会論文誌，Vol. 39, No.7, pp. 2074-2083, July 1998.

[61] 佐藤友実，加藤聡彦，鈴木健二，“共有メモリ型マルチプロセッサを対象としたレイヤ単位の並列処理による通信プロトコルの実装方式，”情報処理学会論文誌，Vol. 39, No. 9, pp. 2727-2740, September 1998.

[62] R. P. Parmelee, T. I. Peterson, C. C. Tillman, and D. J. Hatfield, “Virtual storage and virtual machine concepts,” IBM Systems Journal No. 2, pp. 99-130, 1972.

[63] M. A. Auslander and J. F. Jaffe, “Functional structure of IBM virtual storage operating systems Part 1: Influences of dynamic address translation on operating system technology,” IBM Systems Journal, Vol. 12, No. 4, pp. 368-381, 1973.

[64] Jin Jing, Abdelsalam (Sumi) Helal, and Ahmed Elmagarmid, “Client-Server Computing in Mobile Environments,” ACM Computing Surveys, Vol. 31, No. 2, pp. 117-157, June 1999.

[65] James J. Kistler, and M. Satyanarayanan, “Disconnected Operation in the Coda File System,” ACM Trans. on Computer Systems, Vol.10, No. 1, pp. 3-25, 1992.

[66] Lily B. Mummert, Maria R. Ebling, and M. Satyanarayanan, “Exploiting Weak Connectivity for Mobile File Access,” Proc. 15th Symp. Operating Systems Principles, pp. 143-155, 1995.

[67] L. Mummert, M. Satyanarayanan, “Large Granularity Cache Coherence for Intermit-

tent Connectivity," 1994 Summer USENIX, pp. 279-289, June 1994.

[68] Douglas B. Terry, Marvin M. Theimer, Karin Petersen, Alan J. Demers, Mike J. Spreitzer and Carl H. Hauser, "Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System," Proc. 15th Symp. Operating Systems Principles, pp. 172-183, 1995.

[69] Anthony D. Joseph and M. Frans Kaashoek, "Building Reliable Mobile-Aware Applications using the Rover Toolkit," MOBICOM '96, pp. 117-129, 1996.

[70] Anthony D. Joseph, Joshua A. Tauber and M. Frans Kaashoek, " Mobile Computing with the Rover Toolkit, IEEE Trans. on Computers," Vol.46, No.3, pp. 337-352, March 1997.

[71] Brian D. Noble, Morgan Price, and M. Satyanarayanan, "A Progamming Interface for Application-Aware Adaptation in Mobile Computing," In Proceedings of the 2nd USENIX Symposium on Mobile and Location-Independent Computing, pp. 57-66, 1995.

[72] Barron C. Housel, David B. Lindquist. "WebExpress: A System for Optimizing Web Browsing in a Wireless Environment" MOBICOM '96, pp. 108-116, 1996.

[73] Mads Haahr, Raymond Cunningham and Vinny Cahill, "Supporting CORBA Application in a Mobile Environment," MOBICOM '99, pp. 36-47, 1999.

筆者発表論文

1. 論文誌への発表

[1] 横山繁盛，渡辺尚，水野忠則，“モバイルコンピューティング環境に適した共通メモリ管理方式，”情報処理学会論文誌，Vol. 41, No. 9, pp. 2423-2433, September 2000.

2. 国際会議への発表

[1] Shigemori Yokoyama, Takahiro Okuda, Tadanori Mizuno and Takashi Watanabe, “A Memory Management Architecture for a Mobile Computing Environment,” Proceedings of the Seventh International Conference on Parallel and Distributed Systems: Workshops, pp. 23-28, Iwate, Japan, July 2000.

[2] Shigemori Yokoyama, Tadanori Mizuno and Takashi Watanabe, “A Memory Management Architecture for Mobile Computing Environments and its Evaluation,” World Multiconference on Systemics, Cybernetics and Informatics, Proceedings, Volume 7, Computer Science and Engineering: Part 1, pp. 113-118, Orlando, Florida, USA, July 2000.

[3] Shigemori Yokoyama, Tadanori Mizuno and Takashi Watanabe, “A Proposal of a Memory Management Architecture for Mobile Computing Environment,” Proceedings of the Eleventh International Workshop on Database and Expert Systems Applications, pp. 28-32, Greenwich, London, UK, September 2000.

3. 研究会／全国大会等への発表

[1] 横山繁盛，渡辺尚，水野忠則，“モバイルコンピューティングシステムにおけるメモリ管理方式，”情報処理学会第 57 回全国大会 講演論文集 (3)，pp. 597-598, October 1998.

[2] 横山繁盛，清水正貴，渡辺尚，水野忠則，“モバイルコンピューティングにおけるメモリ管理方式，”情報処理学会研究報告，情処研報，Vol. 99, No. 13, pp. 33-38, February 1999.

[3] 清水正貴，横山繁盛，渡辺尚，水野忠則，“モバイルコンピューティングシステムのメモリ管理の一考察，”情報処理学会第58回全国大会　講演論文集(3)，pp. 455-456, March 1999.

[4]清水正貴，横山繁盛，渡辺尚，水野忠則，“モバイル環境に適したメモリ管理方式の評価，”マルチメディア，分散，強調とモーバイル(DICOMO’99)シンポジウム，vol. 99, No. 7, pp. 459-464, July 1999.

[5] 奥田隆弘，横山繁盛，水野忠則，渡辺尚，“モバイル環境に適したメモリ管理方式MMMの評価について，”平成11年度電気関係学会東海支部連合大会，講演論文集，pp. 293, September 1999.

[6] 奥田隆弘，横山繁盛，水野忠則，渡辺尚，“モバイル環境に適したメモリ管理方式MMMの考察，”情報処理学会第59回全国大会，講演論文集(3)，pp. 259-260, September 1999.

[7] 横山繁盛，奥田隆弘，水野忠則，渡辺尚，“モバイルコンピューティング環境に適した共通メモリ管理方式について，”情報処理学会研究報告，情処研報，Vol. 2000, No. 14, pp. 33-40, February 2000.

4. 技術誌への発表

[1] 横山繁盛，“コンピュータCOSMO-700の仮想記憶方式，”日経エレクトロニクス，1974年12月30日.

[2] 田渕謹也，横山繁盛，渡辺照久，老田清五，“《MELCOM-COSMO 900－2》の本体装置，”三菱電機技報，Vol. 54, No. 9, pp. 609-613, September 1980.

[3] 横山繁盛，坂本巍，有賀幾夫，渡辺照久，中村俊一郎，“汎用電子計算機《MELCOM EXシリーズ》のハードウェアシステム，”三菱電機技報，Vol. 59, No. 7, pp. 481-485, July 1985.

5. 特許出願

[1] 横山繁盛，渡辺尚，水野忠則，「メモリ管理方法及びコンピュータ」特 2000-293434 [H12. 10. 20] (特許公開).