オペレーティング・システム 49－1
（１９９０．　１２．　７）

分散OS Machとそのインプリメンテーション

乾 和志
オムロン株式会社
OA統轄事業部 EWSシステム事業部
商品開発室

分散OSのMachを、オムロン デスクトップワークステーションLUNA-88Kに移植した。LUNA-88Kは、モトローラ社製RISCプロセッサMC88100を4個搭載可能なワークステーションである。Machは、4.3BSD UNIXとバイナリレベルでのコンパチビリティを持つマルチプロセッサに対応したOSであり、異機種への移植性に関しても特別な配慮がなされている。特にOSの移植には、もっとも大きなネックとなる仮想記憶管理機構に関して異機種への移植を短時間で効果的に行なえるように設計されている。オムロンでは、Machのハードウェア依存部をLUNA-88Kのハードウェアアーキテクチャに合わせるよう設計し、インプリメントを行なった。

Omron LUNA-88K Mach Implementation

Kazushi Inui
OMRON Corporation
Product Development Dpt.
Engineering Work Station System Div.

Mach is a distributed operating system developed at Carnegie-Mellon University which supports multi-processor systems and is binary compatible with Berkeley 4.3 UNIX. Omron Corporation ported Mach to it's LUNA-88K desktop workstation which is based on the 88000 RISC chip set from Motorola. The LUNA-88K system supports up to 4 MC88100 CPU chips with 2 MC88200 CMMU (Cache/MMU) chips per processor. In the design of Mach, considerable thought was given to portability especially in the memory managment system which is normally a bottleneck in porting efforts. Omron ported the hardware-dependent code in the Mach kernel to the LUNA-88K hardware architecture.

1 はじめに

最近、数多くのRISCプロセッサが登場し、話題を集めている。RISCプロセッサは、従来のCISCプロセッサとは別のアプローチを行なうことによって速度向上を図ることに成功している。今後、RISCプロセッサは、ますますその速度を向上させ、将来、RISCプロセッサを用いる技術はコンピュータ技術の核となるだろう。このようなプロセッサの速度の飛躍的な向上はコンピュータの世界の構図を大きく作りかえようとしている。すでに、RISCプロセッサの速度は、メインフレームを越え、スーパーコンピュータに迫る勢いである。そして、このような高速なプロセッサを搭載したワークステーションが既にいくつか登場しており、これらは、机の上に載る程度の大きさ、いわゆるデスクトップタイプで、メインフレーム以上の計算能力を提供している。

オムロンでは、従来よりのLUNAシリーズの上位機種としてLUNA-88Kを開発し、OSとして、米国カーネギーメロン大学の開発した分散OS Machを移植した。

LUNA-88Kは、最大4個のRISCプロセッサMC88100を搭載可能なデスクトップ型のワークステーションである。LUNA-88Kでは、MC88100を25MHzで動作させ、4個のプロセッサ搭載時に最大100MIPSの処理能力を実現している。オムロンが、LUNAの上位機種のワークステーションのメインプロセッサとしてMC88100を選択した最も大きな理由は、MC88100が、マルチプロセッサ構成で使用されることを前提に設計されていたという点である。マルチプロセッサ構成のハードウェアを設計する場合には、バスへのアクセスをどのように軽減するかが大きな問題となってくる。そのため、キャッシュメモリの搭載は、必要不可欠のものになるが、マルチプロセッサのキャッシュ間のコヒーレンシを保つために、バスへのアクセスが増加することになる。MC88100では、このキャッシュ間のコヒーレンシを保つためにスヌーピング機構を搭載しており、最小限のバスアクセスでキャッシュ間のコヒーレンシを保つことができる。

Machは、疎結合と密結合のマルチプロセッサ構成の計算機を想定して設計されている。LUNA-88Kは、メインメモリが一つで、4個のプロセッサが、同一バスを通してメインメモリをアクセスするように設計されている密結合型のマルチプロセッサ構成であり、Machのサポートするハードウェアの条件を満たしている。また、Machは、従来よりの4.3BSD UNIXとオブジェクトレベルで互換性を保つことが可能という、非常に高いレベルでの互換性を有しており、いままでの、4.3BSD UNIXユーザが、容易にMachに移行することができる。LUNA-88Kでは、従来のLUNAとは、プロセッサがことなるためにオブジェクトレベルの互換性は不可能であるが、ソースコードレベルでの互換性を維持している。

Machの大きな特徴の一つに、その高い移植性がある。ハードウェアに依存する部分と、依存しない部分をモジュール分割しており、他機種への移植を非常に容易に行なえるように設計されている。

オムロンでは、Machのハードウェア依存部をLUNA-88Kのハードウェアに合わせて設計し移植を行なった。以下に、その移植の際に留意した点とその実際のインプリメントについて述べる。

2 Machのマルチプロセッサ制御

Machは、疎結合型のマルチプロセッサと密結合型のマルチプロセッサをサポートするように設計されているが、LUNA-88Kのハードウェアは、密結合型のマルチプロセッサ構成であり、これに合わせてインプリメントを行なった。カーネギーメロン大学から提供されるMach 2.5のソースコードには、疎結合型のマルチプロセッサをサポートするソースコードは含まれていない。しかし、Machカーネルがそのプリミティブを備えているために、ユーザレベルで、外部ページャなどをインプリメントすることによって、疎結合型のマルチプロセッサをサポートすることが可能である。

2.1 プロセッサの識別とMachの起動

Machが制御するマルチプロセッサは、一つのマスタプロセッサとそれ以外のスレーブプロセッサに役割を分担している。

LUNA-88Kの場合には、最小の構成は、1プロセッサであるため、マスタープロセッサが1つということになるが、最大4つのプロセッサを搭載した場合には、マスタープロセッサ1つと、スレーブプロセッサ3つという構成になる。なお、LUNA-88Kでは、ハードウェア上では、マスタープロセッサとスレーブプロセッサということを限定せず、4つのプロセッサは、全く対照に設計されている。また、Machでは、それぞれのプロセッサの番号を必要としているが、LUNA-88Kでは、この番号もハードウェア上では設定されておらず、ハードウェアに電源が投入されたあと、ソフトウェア起動時に、それぞれのプロセッサの番号を決定し、そして、マスタプロセッサを決定している。その時に使用可能なプロセッサの個数もソフトウェアで判別しているため、LUNA-88Kで、ユーザがプロセッサを追加あるいは取り外した場合には、自動的に判別されるようになっている。

Machが起動され、4.3BSDのマルチユーザモードになるまでの処理は、ほとんど4.3BSDのそれと同じである。ただ、Machでは、独自の機能についての初期化やカーネルスレッドと呼ばれる、カーネル内の仕事をする特別なスレッドの起動などを行なっている。カーネルスレッドとしてページャやエクセプションハンドラーが起動されている。デバイスの初期化までは、マスタプロセッサで行ない、その初期化後に他のスレーブプロセッサを起動してマルチプロセッサで動作するようになっている。

2.2 システムコール処理

Mach2.5 は、カーネギーメロン大学が開発した Mach の核となる部分 (Mach3.0 では、この部分以外は、ユーザプロセスとしてカーネル外に出されており、この部分は、マイクロカーネルと呼ばれている。) の回りに 4.3BSD UNIX の層を加えるような形でインプリメントが行なわれている。つまり、4.3BSD とバイナリレベルで互換であるという機能は、この層によって実現されている。4.3BSD とのバイナリレベルの互換機能を提供するために、ユーザプロセス (Mach では、タスクと呼ばれる。) とカーネルのインタフェースは、全く同じになっている。ユーザプロセスが割り込みをかけてカーネルにサービスを要求する際、そのサービスの種類によって番号がつけられているが、当然、この番号も同じである。4.3BSD のシステムコール番号は、1 から正方向に番号を割り付けているが、Mach 独自のシステムコールには、-1 から、負方向に番号を割り付けている。

Mach 独自のシステムコールが発行された場合には、カーネルがサービスする全ての処理が並列に行なわれるように設計されている。このため、マルチプロセッサ構成の計算機の場合には、おのおののプロセッサ上で発行されたシステムコールに関しては、全ての処理が終了するまでそのプロセッサ上で処理可能である。ところが、4.3BSD のシステムコールが発行された場合には、異なる動作をする。4.3BSD のシステムコールは、並列処理可能なシステムコールと並列処理不可能なシステムコールに分類されている。

並列処理不可能なシステムコールというのは、このシステムコールをカーネルがサービスする際に、4.3BSD のシステムコール処理関数内で、大域データに対し読み書きを行なった場合、他のプロセッサ上で同じ大域データにたいしてアクセスを行なうときに整合性がとれなくなることがあるシステムコールを意味している。4.3BSD では、マルチプロセッサ上で実行されることを考慮して設計が行なわれていないため、ほとんどのシステムコールは、この並列処理不可能なシステムコールである。

並列処理可能なシステムコールに関しては、Mach のシステムコールと同様の処理が行なわれるが、並列処理不可能なシステムコールに関しては、マスタープロセッサーにその処理を委ねることによって、大域変数へのアクアセスは、同時期には必ず一つのプロセッサのみが行なうことを実現している。

2.3 周辺デバイスからの割り込み

マルチプロセッサ構成にした場合、周辺デバイスからの割り込みなどの IO 処理をどのプロセッサ上で処理するかという問題がある。4.3BSD のデバイスドライバは、直接、カーネルの大域変数のアクセスなどを行なっており、もともと、マルチプロセッサを意識しては設計されていない。このため、今回のインプリメントでも、外部デバイスからの割り込みに関しては、マスタプロセッサで処理している。ただし、プロセッサ間の同期をとるためのソフトウェアで発生させる割り込み (後述) と、クロック割り込みに関しては、各々のプロセッサで受け付けられるようにしている。クロック割り込みは、スケジューリングタイミングの決定や統計情報をとるためにそれぞれのプロセッサで処理を行なう必要がある。

2.4 プロセッサ間割り込み

LUNA-88K では、プロセッサ間の同期をとるためにソフトウェアによって指定したプロセッサに発生させることのできる割り込みを用意している。これは、各々のプロセッサ上の Mach カーネルが管理しているメモリ管理情報に変更を加えたい場合、そのプロセッサが他のプロセッサに対し、現在そのメモリ管理機構に対して行なっているオペレーションを中断し、変更が終るまで待たせる、という処理をする場合に使用している。プロセッサ間のメモリ管理機構に関する整合性は、全てこのソフトウェア割り込みによって保たれている。ソフトウェア割り込みによるメモリ管理機構の整合性に関しては、後述する。

2.5 ロックパッケージ

Mach では、シンプルロックとコンプレックスロックという 2 種類のプロトコルによってカーネル内の大域変数の保護などを行なっている。シンプルロックは、相互排他のために使われる、いわゆるスピンロックである。コンプレックスロックは、以下のような非常に高度な機能を提供している。

- リードライトロック (複数読みだし単数書き込みロック)
  これによって複数のプロセッサが読みだしアクセスのロックを同時にアクセスできる。読みだしロックを書き込みロックに対して上位にする機能も提供されている。
- 休眠可能ロック
  休眠可能ロックに設定されたロックは、他のプロセッサがそのロックを確保していると休眠 (sleep) 操作が呼ばれる。ロックが確保できるようになると起床 (wakeup) 操作が呼ばれる。
- 再帰的ロック
  スレッドが再帰的なスタイルで透過にロックを得ることができるようにロックを設定することを許す。特に、ロックを確保しているスレッドが同じロックを取得しても、再帰ロックはデッドロックを発生しない。

LUNA-88K では、プロセッサに RISC プロセッサである MC88100 を使用しているためにテストアンドセットなどの、ロックプリミティブとして使用できる高級なアセンブラ命令を持っていない。そのため、MC88100 のアセンブラ命令である、xmem という命令を使用し、テストアンドセットの相当する機能を実現している。この命令は、一命令で、指定したレジスタと、指定したアドレスの内容を交換できる命令である。シンプルロックなどのロックパッケージは最終的にこのロックプリミティブを使用している。

3 Mach のハードウェア依存部とその対策

Mach は、4.3BSD UNIX と比較して、かなり異機種への移植性を考慮している。これは、特に、仮想記憶管理を行なっている部分のインプリメントに見られる。仮想記憶管理部は、MMU 依存部と非依存部の 2 層にわかれており、新しい MMU アーキテクチャのハードウェアに移植する場合には、MMU 依存部のみをインプリメントすれば良いようになっている。MMU 依存部以外のハードウェア依存部には、エクセプション処理部、signal/ptrace システムコール処理部、その他のアセンブラルーティンがあげられる。

これらのハードウェア依存部をまとめると以下のようである。

- メモリ管理部
- エクセプション処理部
- Signal/ptrace システムコール部
- その他、アセンブラルーチン

3.1 メモリ管理部

従来の 4.3BSD UNIX では、システム全体が、VAX の MMU のアーキテクチャに大きく依存していて、カーネル内部の随所に VAX の MMU のアーキテクチャに依存したコードが見られた。Mach では、これを大幅に改善し、仮想記憶管理機構の部分をハードウェア非依存部とハードウェア依存部 (pmap モジュール) の 2 階層に分けることによって、ハードウェア非依存部より上位に位置する部分から完全にハードウェアに依存したコードを排除している。この際に Mach では、一般的な MMU の機能を想定しており、非常に抽象化されている。通常のプロセッサの MMU を使用する限りにおいてハードウェア非依存部は、全く変更をする必要がない。LUNA-88K への Mach の移植においても VM のコードは一切変更していない。 物理メモリに対するアクセスは、すべてハードウェア依存部によって行なわれるように設計されている。ハードウェア非依存部は、物理メモリの取り扱いについては全く関知していない。Mach では、この物理メモリを二次記憶領域のキャッシュとして使用している。このキャッシュは Mach カーネルが直接管理を行なわず、仮想記憶サブシステムにこれらの管理を任せている。このため Mach カーネルは完全に仮想記憶管理部分とは独立しており、Mach カーネルが記憶領域を使用する場合には、仮想記憶サブシステムに対して、記憶領域を要求するに過ぎない。この仮想記憶サブシステムは、一般的にページャと呼ばれるものである。4.3BSD のページャは、カーネル内に組み込まれカーネル内処理の重要な一部分となっていたが、Mach では、これをカーネル外で実行し、カーネル自体もページャに依存しないようにインプリメントされている。また、このページャへのインターフェースは、ユーザに公開されており、ユーザタスクにおいてこのインターフェースを使用してページャの機能を実現することが可能になっている。デフォルトのページャは、カーネルスレッドとして実行される。

LUNA-88K の MMU である MC88200 のための pmap モジュールの設計とインプリメントを行なうためにカーネギーメロン大学からリリースされた Mach 2.5 のソースコードの中に含まれている、VAX 用のものを参考にした。このソースコードをもとに MC88200 のために設計し直した。

VAX では、ユーザテキスト,データ,BSS 空間 (P0 リージョン)1G バイト、スタック空間 (P1 リージョン)1G バイト、システム空間 1G バイト、IO 空間 1G バイトというように既に 32bits で表現される仮想アドレス空間の使用方法があらかじめ予約されている。このため、ユーザ仮想アドレス空間とカーネル仮想アドレス空間を一つの仮想アドレス空間上に見ることができるが、その大きさが限定されている。LUNA-88K では、カーネル仮想アドレス空間とユーザ仮想アドレス空間を全く別に用意した。これによってユーザタスクは、4G バイトのアドレス空間を使用することができる。カーネル仮想アドレス空間の中には、物理アドレスと、仮想アドレスとを透過に見せる領域を設けており、この領域を、ページテーブル、セグメントテーブルのアクセスや、IO へのアクセスを行なうために使用している。ページテーブル、セグメントテーブルは、プロセッサごとに用意している。

pmap モジュールは、この上位の階層のメモリ管理部の中のハードウェア非依存部から参照される。そのため、この 2 層間のインターフェースはあらかじめ決められている。pmap モジュールでは、pmap と呼ばれる物理メモリの固まりとしてとり扱われるマップに対して何らかの操作を行なうことが基本となっている。例えば、このマップの生成、破壊などをおこなう操作などが、含まれている。

LUNA-88K では、MMU である MC88200 が、ページテーブル、セグメントテーブルを持ち、ユーザ空間とカーネル空間を指しているエリアレジスターが存在する。LUNA-88K での pmap は、このページテーブルの集合体となっており、この pmap に対する操作は、ページテーブルに対しての操作となっている。なおその操作には、セグメントテーブルの書換えやエリアレジスターの書換えをともなっている。pmap は基本的に一つの仮想メモリ空間に一つ存在している。

具体的なハードウェア非依存部とハードウェア依存部 (pmap モジュール) とのインターフェースは以下のようになっている。

- pmap の作成や破壊のような pmap ハンドルの管理
  - pmap の生成

    pmap_t pmap_create(size)
    vm_size_t size;
    新しい pmap の生成し、その pmap をかえす。***size*** が 0 の場合は、マップは真の物理マップであり、ハードウェアから参照されても良い。***size*** が 0 でない場合は、マップはソフトウェアでのみ用いられ、***size*** までの範囲のみアクセスされる。
  - pmap の参照

    pmap_reference(pmap)
    pmap_t pmap;
    pmap への参照カウンターをインクリメントする。複数のハードウェア独立なマップが ***pmap*** を使用中であることを示すために用いられている。これによって ***pmap*** が間違って解放されることを防いでいる。
  - pmap の参照の放棄

    pmap_destroy(pmap)
    pmap_t pmap;
    pmap への参照カウンターをデクリメントする。この参照カウンターが 0 になると、***pmap*** は破壊され、それが使用していた資源は全て解放する。
- pmap 領域操作-pmap 内の領域のマッピングを変更するプリミティブ
  - マップへの登録

    pmap_enter(pmap, v, p, prot, wired)
    pmap_t pmap;
    vm_offset_t v;
    vm_offset_t p;
    vm_prot_t prot;
    boolean_t wired;
    物理ページ ***p*** を指定された ***pmap*** の指定された仮想アドレス ***v*** に、指定された保護属性 ***prot*** とともに格納する。***wired*** が指定されていれば、ページは ***wire down*** される。
  - マップからの削除

    pmap_remove(pmap, s, e)
    pmap_t pmap;
    vm_offset_t s, e;
    pmap から指定したアドレスの領域を取り除くスタートアドレスとエンドアドレス (***s*** と ***e***) は、ページ境界で丸められることが保証されている。
  - 保護属性の変更

    pmap_protect(pmap, s, e, prot)
    pmap_t pmap;
    vm_offset_t s, e;
    vm_prot_t prot;
    pmap の指定した領域上に保護を設定する。スタートアドレスとエンドアドレスは、***s*** と ***e*** で指定される。
- pmap の大域的操作
  - 無効化

    pmap_remove_all(phys)
    vm_offset_t phys;
    物理アドレス ***phys*** を含む物理ページを、それが存在している全ての pmap から取り除く。
  - コピーオンライト

    pmap_copy_on_write(phys)
    vm_offset_t phys;
    物理アドレス ***phys*** を含む物理ページの、書き込みの権限を全ての pmap から削除する。
- 物理メモリを操作するルーチン

- 物理メモリをゼロで埋める

  pmap_zero_page(phys)
  vm_offset_t phys;
  物理アドレス *phys* を含む物理ページを 0 で埋める。

- 物理ページのコピー

  pmap_copy_page(src_addr, dst_addr)
  vm_offset_t src_addr, dst_addr;
  物理アドレス *src_addr* を含む物理ページを、物理アドレス *dst_addr* を含む物理ページにコピーする。

- マップの使用

  pmap_activate(pmap, thread, cpu)
  pmap_t pmap;
  thread_t thread;
  int cpu;
  pmap をプロセッサ *cpu* 上でスレッド *thread* が使用できるようにする。

- マップの使用の中止

  pma_deactivate(pmap, thread, cpu)
  pmap_t pmap;
  thread_t thread;
  int cpu;
  pmap をプロセッサ *cpu* 上で使わないようにする。

- その他

  - 同期制御

    pmap_update()
    実際に pmap に対して加えられてきた変更をここで反映する。この関数が呼ばれるまで、変更は、*pmap_enter* を除いて遅らせることができる。

  - ページング許可設定

    pmap_pageable(pmap, start, end, pageable)
    pmap_t pmap;
    vm_offset_t start;
    vm_offset_t end;
    boolean_t pageable
    pmap 中のアドレス *start* から *end* の範囲内の領域にページングを許可するか許可しないかをしてする。

LUNA-88K の pmap モジュールを設計する際、マルチプロセッサ構成の計算機の場合に重要となる、他のプロセッサとの MMU の整合性を考慮する必要があった。複数のプロセッサが同じ pmap に対してアクセスを行なう可能性があるからである。

例えば、カーネルは、それぞれのプロセッサ上で動作しているので、カーネルの pmap は、複数のプロセッサから同時にアクセスされる危険性を持っている。また、Mach では、スレッドという同じメモリ空間上で動作する仕事の単位毎にプロセッサを割り当てるため、一つのメモリ空間を表現している pmap を、複数のプロセッサが同時にアクセスする可能性がでてくる。この可能性を排除するために、pmap を更新する場合には、他のプロセッサにソフトウェア割り込みで、自分が、pmap を更新することを知らせ、他のプロセッサがその pmap を放棄するのを待つという手法を用いている。

pmap を更新する際、以下のような手順で更新を行なっている。

1. 自分が更新したい pmap を使用している他のプロセッサがあるかどうか調べる。
2. もしあれば、そのプロセッサに対してソフトウェア割り込みをかけて自分がその pmap を使用したいことを伝える。
3. ソフトウェア割り込みをかけたプロセッサが pmap を解放するのを待つ。
4. 処理続行

このような手順で pmap を更新することによって他のプロセッサの MMU との整合性を保っている。

3.2 エクセプション処理部

OS では、外部デバイスからの割り込みやプロセッサの内部割り込みを処理する必要があるが、これらはプロセッサのアーキテクチャに依存している、特に RISC プロセッサは、複数段のパイプラインを持ちパイプライン上でインス

トラクションを処理中にも割り込みを許すため、エクセプションハンドリングにおけるシステムソフトウエアの処理に対する負担が大きくなっている。例えば、エクセプション処理からの復帰の際にはエクセプション発生時のパイプライン内にあるメモリアクセス動作をすべてソフトウェアでエミュレーションしなければならない。

エクセプション処理部は 88K の場合、大別すると次の様になる。

1. システムコール処理
2. 割り込み処理
3. 例外処理
4. データアクセスエミュレーション
5. 浮動小数点演算例外処理

3.3 signal/ptrace システムコール

通常カーネルがユーザにサービスするシステムコールに関しては、ハードウェア依存というものはないが、4.3BSD の場合、signal と ptrace の 2 つのシステムコールに関しては、ハードウェアに依存している。

signal システムコールは次のように動作する。signal をユーザタスクが受けた際にその処理ハンドラが起動されるが、signal 発生から処理がユーザタスクへ渡されるまでと、処理ハンドラが終って signal 発生時のアドレスに戻るまでの処理は、ユーザのコンテキストのスタック上で実行される。これらのプログラムは、シグナル発生時にカーネルが、ユーザスタックへコピーを行なう。この部分は、プロセッサのアーキテクチャに大きく依存している。

ptrace システムコールは、プロセスのステップ実行を可能にする。ほとんどの CISC プロセサでは、インストラクションレベルのステップ実行を CPU レベルでサポートしているが、RISC プロセッサである MC88100 では、これらの機能を有していない。そのため、この機能をソフトウェアで実現した。

4 Mach と LUNA-88K の今後の問題点

LUNA-88K は、4 つのプロセッサが一つのバス上に搭載されるという、もっとも一般的なハードウェア構成であるが、このような構成のハードウェアの場合、バスのトラフィックが最も OS の性能、システムの性能に影響する。バスのトラフィックを軽減させるために、我々は、MC88100 プロセッサを採用し、同プロセッサのスヌーピング機構を利用しているが、その上でキャッシュのヒット率が重要な問題となる。また、Mach カーネルのスケジューリングにより、タスクは、プロセッサ間を移動しながら仕事を実行することになるが、その移動のスケジューリングを如何に最適化するかで、キャッシュのヒット率に影響してくる。我々が現在、市場にリリースしている Mach カーネルでは、このプロセッサ間のスケジューリングが十分最適化されておらず、ある特定の仕事を実行した場合、プロセッサを増やすことによって逆に速度の低下が見られることがある。次回にオムロンからリリースするバージョンでは、この部分を改善する予定である。