6G－3

密結合マルチプロセッサ用階層構造OS MUSTARDにおけるUNIX核の実現

桃井武* 関美鈴** 広屋修一*
* 日本電気(株) ** 日本電気マイコンテクノロジー(株)

1. はじめに

MUSTARDは，V60/V70で構成される密結合型マルチプロセッサ・システムをターゲットとする．本システムは，シングルプロセッサの交換機システムなどでの利用実績をもつ組込みシステム用リアルタイム核 RX616 [1] と，UNIX 核を組み合わせた RX-UX を，マルチプロセッサ化することにより実現されている．

MUSTARD も RX-UX と同様，UNIX 核をリアルタイム核 RX616 の上に構築しており，基本部分を下位 OS である RX616 に依存し，ファイルサーバ機能等を上位 OS である，UNIX 核に依存している階層構造 OS である．

MUSTARD のリアルタイム核部の特徴とマルチプロセッサ上での実現については，すでに報告 [3-5] されており，本稿では，UNIX 核を，マルチプロセッサ対応に機能拡張した時に採用した UNIX 核の排他制御の実装について述べる．

2. RX-UX の概要 [2]

MUSTARD のベースとなった RX-UX は，階層構造になっており，以下の特徴を持っている．

i) リアルタイムタスクと UNIX プロセスの同期，通信，ファイルの共有

ii) リアルタイム核側のリアルタイム性の維持

iii) UNIX 核による標準インタフェースの提供

さらに，UNIX プロセスに対し，UNIX のシステムコールだけでなく，リアルタイム核のシステムコールを呼び出せるように拡張し，リアルタイム核の提供するタスク間通信機能 [2] により，リアルタイムタスクと UNIX プロセスの同期・通信 [4] を可能とした．

また，リアルタイムの世界の事象に対する，応答性能を得るために，スケジューリング方式を変更 [5] して，UNIX 核のプリエンプションのうち，リアルタイムタスクによる UNIX 核のプリエンプションは許可することにより，UNIX 核の排他制御機構には手を加えずに，リアルタイム側の応答性能を維持した．

3. MUSTARD の概要

3.1 MUSTARD の構成

MUSTARD は以下の構成をとっている．

i) 排他制御機能，ディスパッチ機能，プロセッサ間通信機能は，リアルタイム核で実現し，UNIX 核はそれを利用する．

ii) ファイルは UNIX 核のデバイスドライバが管理し，リアルタイムタスクは UNIX 核のデバイスドライバを利用する．

iii) リアルタイム核，UNIX 核，共にシングル・スレッドの対称型マルチプロセッサ・カーネルとして実現されている．

なお，リアルタイム核に関しては，将来マルチ・スレッド方式に移行する予定である．

iv) 複数のデバイスドライバが並行動作可能である．

v) デバイスドライバは UNIX 核と並行動作可能である．

3.2 UNIX 核の排他制御

MUSTARD においては，UNIX 核の排他制御に関して，以下の方式を採用している (図1参照)．

step1) システムコールが発行されカーネル内を実行しようとする場合，プロセスはどのプロセッサで走行していても，他にカーネル内を走行しているプロセスがなければ，そのままそのプロセッサ上でカーネル内を走行する．

step2) 他にカーネル内を実行中のプロセスが存在する場合，UNIX 核待ちキューにつないで待つことにし，他のプロセスにディスパッチする．

step1) については，TOP-1 [6] に見られるように，カーネル専用のプロセッサを設ける方法も考えられ，以下のような長所と欠点を持つ．

(長所) 割り込み制御を行なうプロセッサと，割り込みが入るプロセッサとが同じであるため，割り込みコントローラを制御することで排他制御を行なうことが可能であり，シングルプロセッサのデバイスドライバに手を加えることなくマルチプロセッサ化することが可能である．

A Method for UNIX Kernel on the Multiprocessor Operating System MUSTARD
Takeshi MOMOI*, Misuzu SEKI**, Shuuichi HIROYA*
*NEC Corporation, **NEC Microcomputer Technology Ltd.

UNIX は AT&T 社の Bell研究所で開発された OS です．

(欠点) カーネル内を走行しようとするたび，及び，カーネル内から戻るたびにディスパッチする必要があり， ディスパッチの分だけオーバーヘッドになる上，特定のプロセッサに，あるプロセスを割り当てる機能をディスパッチ機能に追加する必要がある

step2) については，カーネルを獲得できなかったプロセスは，そのプロセッサ上で busy wait により待つ方式も考えられるが，UNIX 核の連続走行時間は，リアルタイム核と比較して非常に長く（およそ数千倍），リアルタイム・システムとしての性能を，著しく低下させることになるので採用しなかった．

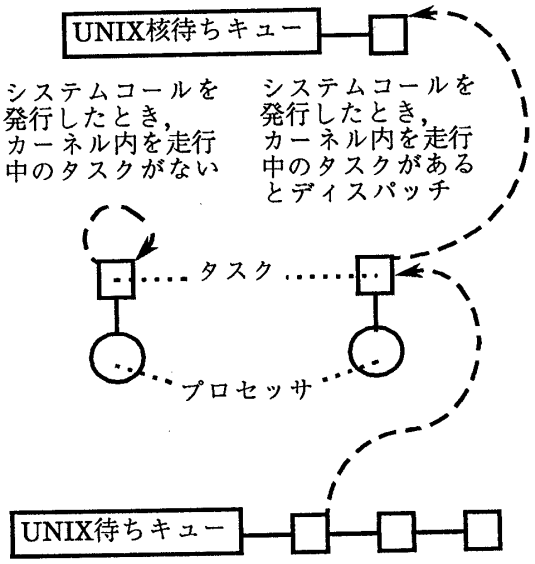

図1 UNIX核の排他制御

4. MUSTARD の実現上の得失

RX-UX をマルチプロセッサ・システム対応に機能拡張するに当たり，階層構造になっていない通常の UNIX をマルチプロセッサ化する時と比較して，発生した問題の違いを以下に列挙する．

4.1 階層構造であったことによる利点

以下の機能に関しては，リアルタイム核ですでに実現されており，UNIX 核では，それらの機能を流用することができ，改めて実現する必要がなかった．

i) 排他制御機能

ii) ディスパッチ機能

iii) プロセッサ間通信機能

iv) プロセッサ管理機能

v) 初期設定機能

vi) 耐故障機能

4.2 下位 OS のリアルタイム性維持に関する問題

i) リアルタイムタスクの実時間処理の保証

リアルタイム性を維持するために，カーネル内を走行中の UNIX プロセスも，より高い優先度を持つリアルタイムタスクによるプリエンプションを可能にする必要がある．

しかし，プロセッサローカルな割込制御は，その制御区間内でプリエンプションがおきると割込制御の解除ができなくなる．制御区間内をプリエンプション禁止で走行することが望ましいが，割込制御区間は長いもので数ミリ秒に及ぶものもあり，そのような区間に対し，リアルタイムタスクの実時間性をどう保証するかが問題である．

ii) クロック処理の実時間性

RX-UX では UNIX のクロック処理が重いことによって，リアルタイム核のクロック処理が実時間性を失うことを避けるため，クロックをもっとも優先度の高いリアルタイムタスクとして実現している．

UNIX 核は最大約 1 秒連続走行することが実測されているおり，クロックタスクを UNIX 核内で走行させようとすると，クロックとしての機能を果たせなくなるので，UNIX 核を獲得せずに走行させないといけない．

5. おわりに

本稿では，UNIX 核の実現の一部，及び，階層構造になっている OS を，マルチプロセッサ・システム対応に機能拡張する上で，発生し得る問題について考察した．現在，MUSTARD は UNIX 核の実現を終え，検査及び評価に入ろうとしているところである．MUSTARD の実現の詳細と評価については，今後報告していく予定である．

【参考文献】

[1] 古城他，"V60リアルタイムOSの設計"，情処第33回全国大会1C-4, 1986.

[2] 下島他，"V60/V70リアルタイムUNIX -概要-"，情処第35回全国大会3D-3, 1987.

[3] 広屋他，"種々のハードウェアに適用可能な密結合マルチプロセッサ用OS MUSTARD"，情処第39回全国大会4P-5, 1989.

[4] 渡辺他，"密結合マルチプロセッサ用OS MUSTARDのプロセッサ間通信"，情処第39回全国大会4P-6, 1989.

[5] 川口他，"密結合マルチプロセッサ用OS MUSTARDの耐故障性の強化と障害処理機構"，情処第39回全国大会4P-7, 1989.

[6] 山崎他，"TOP-1 オペレーティング・システム (2) プロセス・スケジューリング"，情処第39会全国大会3P-5, 1989.