CTRON/UNIXインタフェーサにおけるsignalの実現

5P-5

工藤　明彦　　竹内　宏典

（ＮＴＴ情報通信処理研究所）

1　はじめに

ＣＴＲＯＮ*1仕様に準拠したリアルタイムＯＳ（以下、ＣＴＲＯＮと略す）の利用者に対してＵＮＩＸ*2環境を提供するために、ＣＴＲＯＮ／ＵＮＩＸインタフェーサ（以下、ＵＮＩＸインタフェーサと略す）を開発した。ＵＮＩＸ環境とは、ＵＮＩＸ標準システムコールとＵＮＩＸ標準コマンドが使用でき、さらに、利用者プログラムがＵＮＩＸシステムと同様に動作する環境である。

本稿では、ＵＮＩＸのsignal機構をＣＴＲＯＮ上に実現する方法について述べる。

2　signal機構[1][2]

signalは、プロセスに非同期イベントの発生を伝えるものである。プロセスは kill システムコールにより他のプロセスにsignalを送ることができる。また、カーネルからプロセスにsignalが送られることもある。

System Ｖ（リリース２）ＵＮＩＸシステムにおいては、表１に示す１９種類のsignalが定義されている。

プロセスはsignalを受けた時の動作として次の３種類のいずれかを定義することができる。

①プロセス終了
②signal無視（SIGKILL を除く）
③利用者定義ルーチンの実行（SIGKILL を除く）

2．1　signalの送信

プロセスからのsignal送信は、killシステムコールの延長で、カーネルが送信先プロセスのプロセステーブルエントリ内のsignalフィールドに、signalの種類に対応するビットを立てることにより行われる。送信先プロセスは、他プロセスでも自プロセスでもよい。

プロセスがユーザ空間を走行中にプログラム割り込み（メモリアクセス違反、等）を起こした場合など、カーネルから内部的に当該プロセスに対してsignalを送信する場合も、同様にsignalフィールドにビットを立てる。

2．2　signalのハンドリング

signalのハンドリングは、プロセスがカーネルモードからユーザモードに戻る際に行われる。具体的には、次の３つの場合がある。

①システムコールからリターンする時に、カーネルモードからユーザモードに復帰
②ユーザモードで走行中のプロセスが、タイムスライス切れにより待ち状態となった後、再びスケジュールされる時に、カーネルモードからユーザモードに復帰
③ユーザモードで走行中のプロセスに対する割り込みが発生した場合（カーネルからsignalを送信する場合）、その割り込み処理からリターンする時に、カーネルモードからユーザモードに復帰

なお、sleep 状態のプロセスに対してsignalが送信されると、割り込み可能なプライオリティでsleep している場合に限り、カーネルが当該プロセスをwakeupする。そして、signalのハンドリングは、前述のとおり、ユーザモードに復帰する際に行われる。

3　ＵＮＩＸインタフェーサでのsignalの実装

3．1　signal実装上の問題点

ＵＮＩＸインタフェーサでsignalを実装する際に問題となるのは、２．２で述べた②の場合である。すなわち、筆者らが開発したＵＮＩＸインタフェーサでは、プロセスをＣＴＲＯＮのタスクに対応づけており[3]、タスクのスケジュールは基本ＯＳのタスク管理機能に委ねているので、プロセススケジュールの契機を関知できない。したがって、プロセスがユーザ空間で走行している時に他のプロセスから送られたsignalは、そのプロセスがシステムコールを発行するか、割り込まれるまで、ハンドリングできない。

この問題に対して、筆者らは、ＣＴＲＯＮの非同期イベント通知の機能を利用して、signalの発生をプロセスに通知することにより解決した。

3．2　ＣＴＲＯＮの非同期イベント通知

ＣＴＲＯＮの非同期イベント通知の機能には、割り込みと例外がある。割り込みは、Ｉ／Ｏの完了通知など基本ＯＳで処理されるものであり、ＵＮＩＸインタフェーサが関与することはできない。一方、例外は、プログラムに起因して発生するものであり、システムコールにより意識的に発生させることも可能である。

例外には、内部例外と外部例外がある。内部例外は自タスクに起因する場合に発生する例外であり、外部例外は他タスクに起因する場合に発生する例外である。外部例外は、タスク間の非同期イベントの通知に使用することができる。以上より、ＵＮＩＸインタフェーサでは、他プロセスからのsignal発生を通知するために、外部例外を利用した。

表１　signalの種類

signal名称	signalの意味	備　考
SIGHUP	ハングアップ	
SIGINT	割り込み	
SIGQUIT	クィット	①
SIGILL	不正命令	①、②
SIGTRAP	トレーストラップ	①、②
SIGFPE	浮動小数点例外	①
SIGKILL	強制終了	③
SIGSYS	システムコールに対する無効パラメータ	①
SIGPIPE	readするプロセスが存在しないパイプへのwrite	
SIGALRM	アラームクロック	
SIGTERM	ソフトウェア終了シグナル	
SIGUSR1	ユーザ定義シグナル１	
SIGUSR2	ユーザ定義シグナル２	
SIGIOT	ＩＯＴ命令	①、④
SIGEMT	ＥＭＴ命令	①、④
SIGBUS	バスエラー	①、④
SIGSEGV	セグメンテーション違反	①、④
SIGCLD	子プロセスの終了	④
SIGPWR	電源断	④

①プロセス終了時コアダンプ生成
②ユーザ定義ルーチン呼び出し時リセットしない
③ユーザ定義ルーチンは定義できない、無視できない
④ＳＶＩＤでは処理系定義

Implementation of UNIX Signals on CTRON Architecture

Akihiko KUDOH and Hironori TAKEUCHI
NTT Communications and Information Processing Laboratories

3．3　外部例外を利用したsignal機構

ＵＮＩＸインタフェーサでは、signalを他のプロセスに送る場合に、対象プロセスのsignalフィールドにビットを立てるとともに、対象プロセスに対応するタスクに対して外部例外を発生することとした。この例外発生により、対象タスクがスケジュールされる際に、例外ハンドラが実行されるので、例外ハンドラの処理の延長で、signalのハンドリングを行うことが可能となる。

4　実現方式

4．1　signalの送信

signalの送信は、ＵＮＩＸインタフェーサがプロセス毎に持つプロセステーブルエントリのsignalフィールドに、signalの種類に対応するビットを立てることにより行う。さらに、他プロセスへのsignal送信の際には、送信先プロセスに対応するタスクに対して、外部例外を発生する。また、プロセスがユーザモードで走行中にプログラム割り込みを起こした場合などは、内部例外が発生するので、例外ハンドラの中で、ＵＮＩＸインタフェーサカーネルから当該プロセスへのsignal送信と、signalハンドリングを行う。本方式により、ＵＮＩＸと同等のsignal送信機能を実現した。

4．2　signalのハンドリング

signalのハンドリングは、次の２通りの契機で行う。

（１）プロセスがカーネルモードからユーザモードに復帰する際に、ＵＮＩＸインタフェーサがプロセス毎に持つプロセステーブルエントリのsignalフィールドを参照し、対応するsignalのハンドラを呼び出す。

（２）例外ハンドラの中で、例外発生地点がユーザ空間であった場合に、ＵＮＩＸインタフェーサがプロセス毎に持つプロセステーブルエントリのsignalフィールドを参照し、対応するsignalのハンドラを呼び出す。なお、例外発生地点がカーネル空間であり、かつ、内部例外であった場合には、ＵＮＩＸインタフェーサのpanic 処理を行う。

上記の（１）はＵＮＩＸと同等である。また、（２）については、ＣＴＲＯＮが外部例外により例外ハンドラを呼び出す契機が、タスクスケジュールの契機となることから、実効上はＵＮＩＸと同等と見なすことができる。

4．3　signalハンドラの登録

signalのハンドリングは、signalの種類に対応するsignalハンドラを呼び出すことにより行う。ＵＮＩＸインタフェーサでは、ＵＮＩＸと同様に、デフォルトのsignalハンドラを持ち、また、利用者のルーチンをsignalシステムコールにより、signalハンドラとして登録することもできる。これは、ＵＮＩＸインタフェーサ内で、プロセス毎にsignalハンドラのエントリアドレスを保持することにより可能とした。

4．4　端末割り込み

端末から割り込み文字、クイット文字が入力された場合や回線が切断されたりログアウトした場合にも、signalが発生する。ＵＮＩＸインタフェーサでは、図１に示すように、端末からのデータを受信するためにＲＥＡＤタスクが存在し、signalはＲＥＡＤタスクからの外部例外により通知される。

4．5　signalキックタスク

ＵＮＩＸインタフェーサ上のプロセスのスケジュールはＣＴＲＯＮに委ねているため、図２に示すように、タイミングによりsignalをチェックできない場合がある。この場合、プロセスがユーザ空間に戻った後にユーザ空間で走りきる場合など、そのsignalがハンドリングされないことがある。

これを防ぐために、各プロセスのsignalフィールドを定期的に走査するsignalキックタスクを設ける。signalキックタスクは、ハンドリングされていないsignalを見つけると、そのプロセスに外部例外を発生し、通知する。

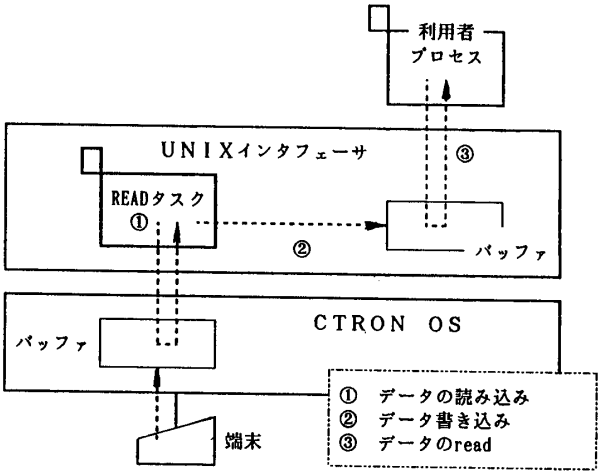

図１　ＲＥＡＤタスク

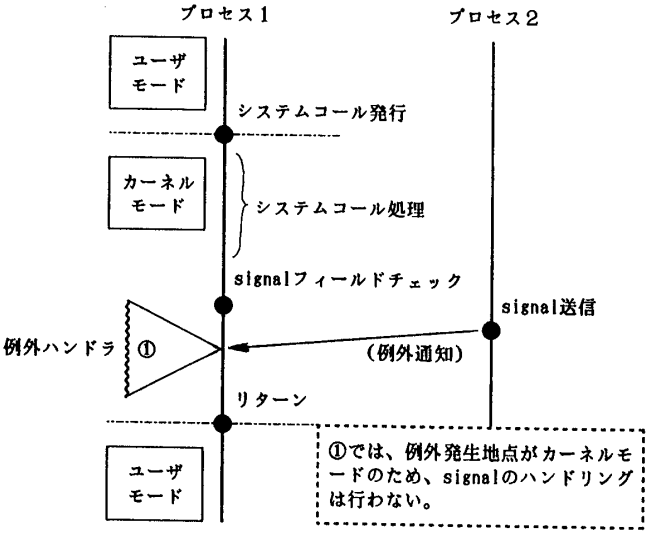

図２　signalの取りこぼし

5　まとめ

ＣＴＲＯＮの例外機能を利用して、ＵＮＩＸのsignal機構を実現することができた。この結果、全てのsignalとsignalに係わるシステムコールをＣＴＲＯＮ／ＵＮＩＸインタフェーサ上に提供可能とした。

参考文献

[1]M.J.Bach; The Design of the UNIX Operating System Prentice-Hall, Inc. (1986)

[2]System V Interface Definition Issue 2, AT&T (1986)

[3]竹内、工藤；「ＣＴＲＯＮ／ＵＮＩＸインタフェーサにおけるプロセス制御」本大会

---

*1 ＣＴＲＯＮは、Central and Communication TRON (the realtime operating system nucleous) の略称である。

*2 ＵＮＩＸはＡＴ＆Ｔが開発し、ライセンスしているオペレーティングシステムである。