3V-7

実時間オペレーティングシステムR²のタスク管理方式

楠田 修三　大久保 英嗣　津田 孝夫　小林 正典(京都大学工学部)
杉村 邦彦　白濱 和人　友田 和伸((株)ダイヘン)

1. はじめに

R²は、実時間システムにおけるソフトウェアの応答性、信頼性ならびに移植性の向上を目標として開発された実時間オペレーティングシステム(以下OSと略す)である。一般に、実時間システムは応答性を向上させるために多くのタスクから構成される。これらのタスクは、OSを介して相互に、あるいは、外部とインタラクションを行いながら活動する。従って、OSのタスク管理方式は実時間システムの構築において重要な意味を持つ。

本稿では、R²におけるタスク管理の特徴ならびに方式について述べる。

2. R²のタスク管理の特徴

R²のタスク管理部を設計するに当たって、我々は応答性、信頼性、分散処理に焦点を当てた。実時間システムのソフトウェアには即応性が要求される。応答性は、割り込みや入出力のアーキテクチャに帰するところが大きいが、活動の主体であるタスクの相互作用、特に、タスク間通信の機構とも密接に関係している。そこでR²では、単に通信オーバーヘッドを短縮するというアプローチは採らず、個々の通信の特性に適した通信形態(同期、非同期、メッセージ長、バッファ容量)を選択させることによってタスク間通信の高速化を図っている。

信頼性は実時間システムにおける重要な問題である。R²ではシステムの信頼性を向上させるための1つの手段として、システムのタスク構成に秩序を持たせることを考えている。このために、タスク相互の作用においてタスクの親子階層に基づく一定の制限を設けている。この階層的アプローチの狙いは、システムの論理構造を簡単にすること、および、プログラムミス等によるタスク間の不当な干渉を排除することにある。信頼性を向上させるためのもう1つの手段として、R²では、稀にしか発生しない例外事象に対する豊富な処理機能を提供している。

近年、小型計算機の価格低下と性能向上に伴い、それらを通信回線で結合することによって低価格で高性能な計算機システム(分散システム)が多数構築されるようになった。R²では、分散システムに必要なCPU間通信の機能をOS核の中に取り入れ、分散透明なタスク間通信(および同期)の機能を実現している。従って、R²は分散OSとしても位置付けられる。

3. R²のタスク管理方式

(1) スケジューリング方式

R²では、多くの実時間OSと同様に、優先度に基づく事象駆動方式によるタスクスケジューリングを行っている。より高度なスケジューリングは、中断状態(後述)を用いて親タスクが複数の子タスクを制御することにより可能である。

(2) タスクの状態遷移

R²におけるタスクの状態遷移を図1に示す。標準的なタスクの状態遷移は、実行状態、実行可能状態、待ち状態の3状態間の遷移であるが、R²では中断状態を設けている。これは、R²では単純なタスクスケジューリングのみを実現し、高度なスケジューリングをユーザが自由に実現できるようにするためである(例えば、特定のタスク集合を時分割スケジュールする等のことは簡単に実現可能である)。

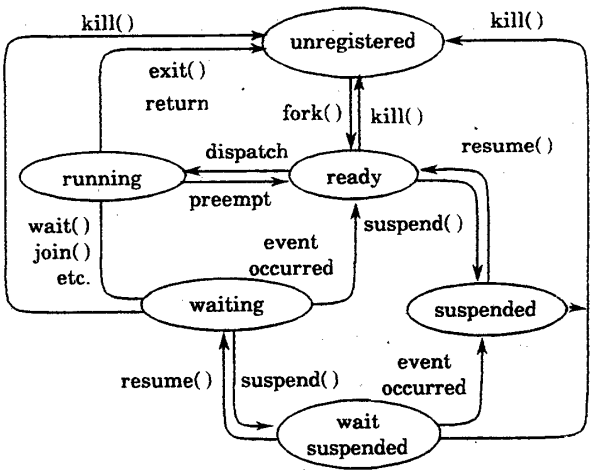

図1 タスクの状態遷移

(3) タスク間の関係仕様

R²では、支配制御、同期および通信、資源の共用、例外事象の波及において図2のような関係仕様を定義している。即ち、支配制御(後述)は親子タスク間でのみ成立する。同期および通信は、基本的にルートタスク間および親子タスク間でのみ成立す

The Architecture of Task Management in Real-Time Operating System R²

SYUZO KUSUDA, EIJI OKUBO, TAKAO TSUDA and MASANORI KOBAYASHI (Kyoto University)
KUNIHIKO SUGIMURA, KAZUTO SHIRAHAMA and YASUNOBU TOMODA (Daihen Corporation)

る。資源の共用は親タスクの所有する資源に関して親とその子孫タスクの間でのみ成立する。例外事象の波及は親タスクに対してのみ成立する。

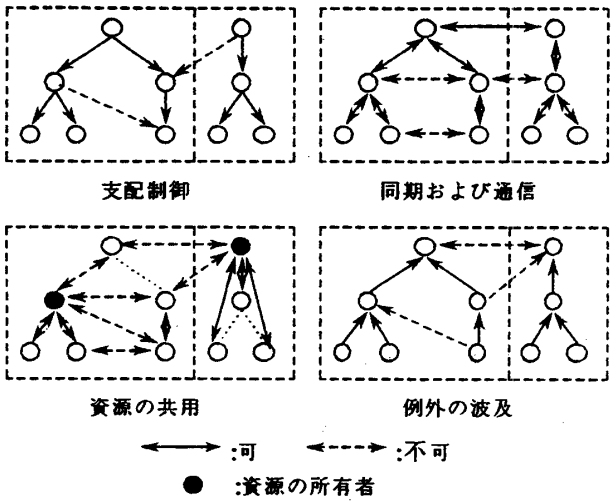

図2 タスク間の関係仕様

(4) 支配制御

R²におけるタスクには、システム立ち上げ時に生成されるルートタスクとシステム走行中に生成される子孫タスクとがある。これらのタスクはルートタスク毎に木構造の親子関係を形成する。親タスクは子タスクに対して、(生成、)削除、終了待ち、中断および再開を行うことが可能である。我々はこれらの制御を支配制御と呼んでいる。子タスクは、関数を単位としてシャムタスクの形態(親タスクとコード領域およびデータ領域を共有する)で生成する。

(5) 同期および通信

R²では、直接指名方式に基づく分散透明な同期と通信を実現している。(3)の関係仕様において、同期や通信の成立するタスク間の関係には一定の制限があることを示した。しかし、祖先のタスクを経由して相手のタスク識別子が得られれば、自由に当該タスクと同期や通信を行うことが可能である。

(a) 分散透明な環境

R²では、相手のタスクがいずれのCPU上に存在しようと同一の形式により同期や通信を行うことが可能である。即ち、アプリケーションレベルでは、CPU間の通信制御が隠蔽されていることになる。我々このような環境を分散透明な環境と呼んでいる。

(b) 同期

各タスクは1語(16ビット)の信号バッファを持つ。信号バッファの各ビットが当該タスク固有の事象に対応している。タスク間で同期を取るには、互いに信号を送り合えばよい。この信号バッファはOSとの同期、即ち、I/O終了待ちにも使用可能である。

(c) 通信

R²における通信は次のように分類される。

(イ) 送信(同期、非同期)

(ロ) 受信(同期、非同期)

(ハ) 返信

(ニ) 中継

ここで、同期送信とは返信を待つということを意味し、同期受信とはメッセージが到着するまで待つということを意味している。返信と中継は同期送信とペアで用いる。前者は同期送信に対する応答を返すためのものであり、後者はその応答を他のタスク(子タスクに限る)に委託するためのものである。

受信用バッファは各タスクの空間に設ける。バッファのサイズと容量(バッファの数)は各タスク毎に設定可能である。

(6) 資源の共用

R²では、共有ロック方式(バイナリセマフォに相当)による資源管理を行っている。各資源は論理的な識別子により識別される。デッドロックを回避するためと処理を高速化するために、1つのタスクによる複数資源の占有を禁止している

(7) 例外処理

R²における例外には次の3種類がある。

(a) CPU例外(零除算、オーバーフロー等)

(b) ソフトウェア例外(SVCエラー、raise() SVC)

(c) 外部例外(外部事象を例外として登録したもの)

例外が発生すると、CPU例外とソフトウェア例外の場合には、カレントタスクに例外が通知される。もし当該例外に対する登録があればその処理へ移行し、なければシステム標準の処理へ移行する。外部例外の場合には、その登録を行ったタスクに例外が通知され、当該例外処理へ移行する。

例外事象は、raise() SVCにより親へ波及させることが可能である。これにより、階層の低いタスクが引き起こした例外事象を高位の(祖先)タスクが監視し、処理することが可能となる。

(8) その他

R²のタスク管理には、以上に述べたほか、メモリブロック管理機能やリアルタイマ機能が含まれる。

4. おわりに

本稿では、R²におけるタスク管理の特徴ならびに方式について述べた。R²のタスク管理の各機能は、タスクスケジューラを除いてほとんどC言語により記述されている。性能に関しては、現在チューンナップを進めているところである。ユーザ親和性や信頼性に関しては、今後、いくつかのアプリケーションを作成して各方面から総合的に評価する必要がある。