TMS320C6713搭載
DSPスタータ・キットで学ぶ

リアルタイムOS「DSP/BIOS」を利用するDSPアプリケーションの開発

三上 直樹

第1回　DSP/BIOSを使うための準備とハードウェア割り込みの設定方法

本連載ではTexas Instruments社製のDSPスタータ・キットを使用し，DSP用のRTOS「DSP/BIOS」を使用したアプリケーションの開発方法について説明していく．
第1回目は，DSP/BIOSの基本的な使い方を解説する．連載の最終回では，リアルタイム・マルチタスク・システムの例としてFFTを利用するディジタル・フィルタをDSP/BIOS上で実現する予定である．（著者）

一般のプロセッサを使ったリアルタイム・マルチタスク・システムを開発する場合，組み込みシステムの世界でも，μITRONなどのリアルタイムOS（以下RTOS）を利用するのが当たり前になってきました．DSPも組み込みシステムで使われますが，一般のプロセッサに比べて，非常に速い処理速度や応答速度が要求されます．そのため，従来はRTOSとは無縁といってもよい状態でした[注1]．

しかし，DSPの性能向上により，近年はリアルタイム・マルチタスク・システムでDSPの使われる場面が多くなってきました．その代表的な例が携帯電話です．このような組み込みシステムを実現するには，リアルタイムOSを利用しなければ，プログラム開発が不可能と言っても過言ではないでしょう（**コラム1**参照）．

この連載では，Texas Instruments（以下TI）社が提供するDSP/BIOS[注2]という名称のDSP用RTOSを例として取り上げ，DSPアプリケーションを開発する方法について解説します．プログラムの開発やその実行は，TMS320C6713の搭載されたDSK[(1),(2)]を利用します[注3]．使用するCode Composer Studio（以下CCS）のバージョンは3.1で，DSP/BIOSのバージョンは4.9です．CCSの基本的な使い方については，本誌の2009年1月号[(3)]を参照してください．また，使用しているDSPについては，参考文献(4)を挙げておきます．

この連載で使う用語や略語などを，**コラム2**に示します．

1　DSP/BIOS[(4)]を使うための準備

DSP/BIOSを使うためには，コンフィグレーションを行う必要があります．コンフィグレーションの設定はGUI（グラフィカル・ユーザ・インターフェース）の画面から行います[注4]．

まず**図1**のように，CCSのメニューから［File｜New｜DSP/BIOS Configuration...］を選択します．次に，使用するDSPまたはDSKなどの開発システムを選びます．ここ

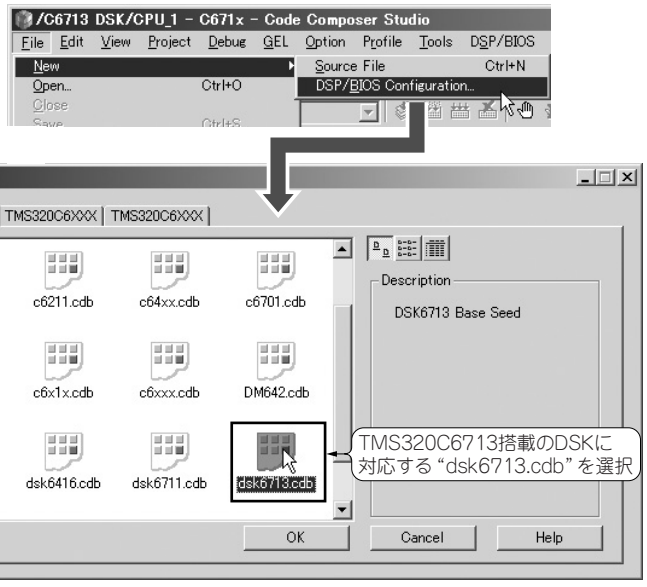

図1　DSP/BIOSコンフィグレーション・ツールの設定

注1：DSP用RTOSとして，1990年にTI社のTMS320C30対応の“SPOX”というRTOSがサード・パーティより提供されていたようである．

注2：「DSP/BIOS」とはDSP用のBIOS（Basic Input/Output System）のことではない．TI社では，DSP/BIOSをRTOSとは呼ばずに，リアルタイム・カーネルと呼んでいるようである．しかし，文献(4)の第14章にも書いてあるが，RTOSと呼んでも差し支えないと筆者も思っている．

注3：DSP/BIOSは，DSPアプリケーション開発用の統合開発環境であるCCSに付属する．もちろんDSPスタータ・キット（以下DSK）にもCCSが付属しているので，DSP/BIOSを使うための新たな出費はない．

注4：μITRON 4.0のように，コンフィグレーションの内容をテキスト・ファイルの形で記述するようになっているRTOSもある．

KEYWORD —— DSP，DSK，RTOS，リアルタイム・マルチタスク・システム

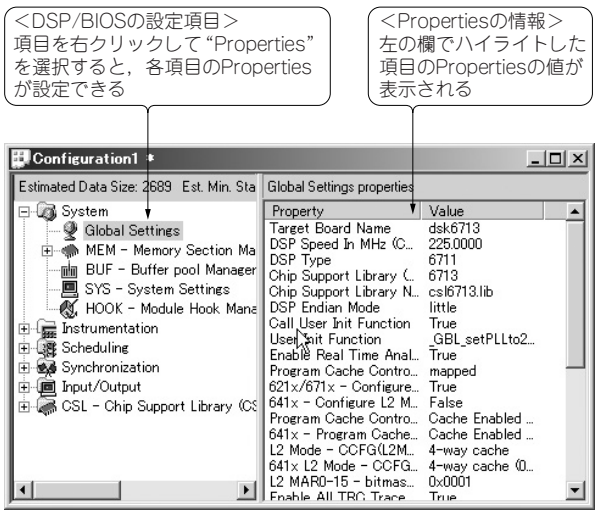

図2 “System”の項目の“Grobal Settings”のプロパティを表示してるようす

では TMS320C6713 搭載の DSK を使うので，“dsk6713.cdb”を選択し“OK”ボタンをクリックします．すると，設定用の GUI ウィンドウが開きます（**図2**）．今回作るプログラムでは，“System”の下の各項目は変更せず，デフォルトのまま使います．

2 DSP/BIOS のスケジューラ

リアルタイム処理をマルチスレッド・システムとして実現するためのスケジューラは，DSP/BIOS の重要な機能です．DSP/BIOS のスケジューラは，プリエンプティブ[注5]なスレッド管理を提供します．

● プリエンプティブなマルチスレッド・システム

スレッドが実行されるようすを**図3**に示します[注6]．スレッドを切り替えながら，システムの仕事全体を進めていきます．

column 1 なぜ DSP で RTOS を使うの？

● 昔は，RTOS は必要なかった

従来の DSP は処理能力があまり高くないということもあり，通常はディジタル・フィルタなどの一つの機能を実現するために使われていました．そのような場合は，特に RTOS を使う必要はありませんでした．

● DSP で多彩な処理を平行して実行したい

DSP の処理能力が高くなると，DSP に何でもやらせたいという要求が生まれてきました．たとえば携帯電話で，音声信号の変調と復調といったディジタル信号処理から，ディジタル・フィルタなどの各種ディジタル信号処理などを平行して実行する必要があります．しかも各信号処理で標本化周波数が異なる場合もあります．さらに，ディジタル信号処理以外の処理，たとえばネットワークに関する処理や GUI 処理から電源の管理まで，多種多様な処理を平行して実行します．したがって，DSP で携帯電話を実現するためにはマルチスレッド・システムを構築する必要があります．

● マルチスレッド・システムには RTOS が必要

マルチスレッド・システムは，もちろん RTOS なしでも構築することはできます．しかし，その場合はスレッドのスケジューリングをすべてプログラマが管理しなければならないので，非常に複雑で，デバッグや保守管理の非常にやりにくいプログラムができあがってしまいます．そのためプログラムの再利用も難しくなります．

一方，RTOS は一般に優先順位に基づくプリエンプティブなスケジューリング機能を備えています．そのため，RTOS を使えば，プログラマは個別の処理に対応するプログラム開発に専念することが可能になり，一つのシステムを多人数で開発する際にも有利になります．また，再利用も比較的楽になります．

● RTOS を使用すればデバッグも簡単になる

リアルタイム処理を行っているディジタル信号処理用プログラムのデバッグでは，プログラムを止めてデバッグするわけにはいかないケースもあります．この連載で扱う RTOS である DSP/BIOS では，リアルタイムでシステムを解析し，デバッグするしくみを提供しています．

たとえば，マルチスレッド・システムにおいて，各スレッドがどのようなタイミングで処理を行っているのかということを，実行しているプログラムを止めることなくリアルタイムで表示することができます．この機能により，優先順位の低いスレッドが優先順位の高いスレッドにプリエンプト（preempt）されるようすを確認し，必要に応じてスレッドの優先順位を変更するということも簡単に行えます．そのほかにも，ある部分の実行時間や，DSP の使用率の推移などを，プログラムを止めることなく取得するための手段も提供しています．

このようなことから，DSP でアプリケーションを開発する際にも，RTOS を利用するケースが増えています．