プロローグ

なぜ，T-Engine が必要とされるのか？

リアルタイム OS の現状

猪飼 國夫

OS といえば，もともとはメイン・フレームという大型機の世界の話でした．最近は Windows や MacOS，Linux，種々の BSD など PC ベースの OS がすぐに思い浮かびますが，アプリケーションの種類や装備されている機器の数からいうと，じつはリアルタイム系の OS がいちばん数が多いと考えられます．

筆者も，1979 年にゲーム機を制御するためにリアルタイム OS のようなものを開発して組み込みました．たかがゲームの機械にと思うかもしれませんが，なぜリアルタイム OS を組み込んだかに言及することで，リアルタイム OS を採用する意味を考えてみたいと思います．

リアルタイム OS 以前の組み込み機器

● 昔のゲーム機や家電製品の制御プログラムはアセンブラで書いた

1970 年代後半に市場に出周り始めたゲーム機は 8 ビットの CPU を使うものでした．当時，ゲームを OS の制御下で動かすという考え方はあまりなく，動作速度を上げるためにプログラムをアセンブラでゴリゴリ書いて，それでも間に合わない部分はハードウェアでまかなうという手法が用いられていました．

一方，このころは家電機器や自動車などの多くのコンピュータ以外の機器に，プログラムによる制御が採用されるようになった時代です．プログラムで制御することにより，より使いやすく細かい制御が可能となり，日本の家電製品はその信頼性とともに利便性で世界中で大人気になりました．

これらの家電機器の多くは 4 ビットのマイコンで制御されていました．ここでの「マイコン」はマイクロコントローラの略で，ROM に書き込まれたプログラムで制御される I/O ポート付きの CPU です．Intel が発売した 4 ビットの CPU 4004 を見て，数多くの 4 ビットの CPU が開発されたのです．

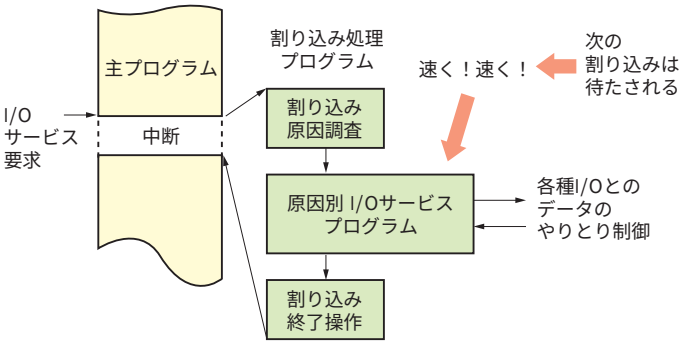

図 1　制御を取られっぱなしの割り込み処理での全 I/O サービス方式

● 急ぐサービスは割り込み処理プログラム内で対処

これらの制御プログラムは I/O からのサービス要求に対して一定時間以内に応えるために，おもに割り込み処理プログラムの中で必要な処理を行うように作られていました（**図 1**）．ものによっては，PLC（Programmable Logic Controller：日本ではシーケンサと呼んでいる）のように，高速にエンドレスで回る構造のプログラムを採用した制御機器もありました．これは昔ながらのリレーによるシーケンス制御をそのままマイコンなどに移植したものです．

家電機器やゲーム機になぜ OS が使われなかったかというと，当時 OS は（UNIX を別にすれば）IBM などの大型計算機のスケジュール管理的な動作を行わせるものだという考えがあったからのようです．もちろん，IBM 機の OS にはリアルタイム処理の機能が盛り込まれていて，チャネルと呼ばれる I/O（通信回線などリモート機器も含む）では一定時間以内の応答の保証をするプログラムがありました．

リアルタイム OS はなぜ必要か

● メカが主体のゲーム機の制御は割り込み処理での対処はやめた

ゲーム機は 4 ビット CPU ではさすがに能力不足だったので，Z80 で制御することにしました．プログラムは当初割り込み処理で対応する予定でしたが，以下のような考えで簡単なリアルタイム処理プログラムを作成することにしました．

1）メカが主体のゲーム機には多くの I/O があり，同時に受け付けなければならないサービス要求がある
2）割り込み処理プログラム内で各 I/O からの要求に対するサービスを行っていたのでは，その処理中にほかの要求に対処するようにプログラムを作ることがたいへんめんどうである
3）ゲーム機が動いている途中で，動きのデータや動作への介入が必要になったとき，別プロセスで動くタスクがあれば，外部から簡単に動作状態のモニタができて機械のデバッグ効率が上がる

ごくごく単純な理由ですが，じつは最大の理由は筆者がリア

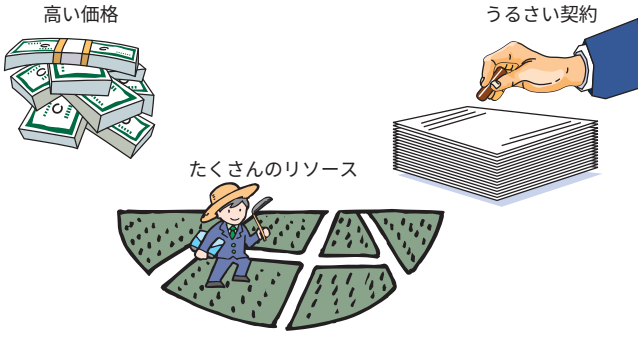

図2　1980年当時の市販リアルタイムOS

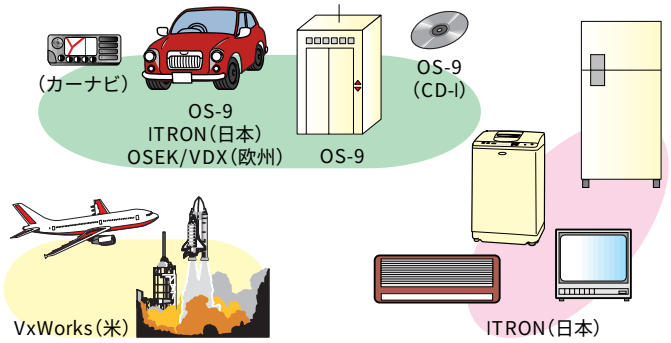

図3　リアルタイムOSの対象別・国別の住み分け

ルタイムOSを作ってみたかったということです．まるでOSオタクのようないいぶんですが，それをもっとまともにして世のためになる構想にしたのが，坂村 健氏のTRON計画だと思います．

● プロセスが独立して動く便利さを実感した

当時，価格と大きさの点で，使うことができたハードウェアには限界があったため，実際に作成したのはリアルタイムOSの純粋なコア部分だけでした．

8Kバイト程度の小さなEPROMにOSと機械制御用プログラム，モニタ用の通信プログラム，当たり管理プログラムを入れなければなりませんでした．

それでもリアルタイムOSの威力は発揮されました．いちばん役に立ったのは，今どきのOSなら常識ですが，ほかのプロセスをモニタ・プログラムから制御(起動・停止・与える条件の変更・内部状況の監視)できたことです．じつは，それまではZ80や8080Aを使った制御装置をいろいろ作ってきましたが，リアルタイムOSは使わずに，すべて割り込み処理プログラム内で対処してきました．

● 手が出なかったリアルタイムOS

それまでリアルタイムOSを使わなかった理由は，安価に使えるOSとしては1974年にDigital Research社から出されたCP/Mしかなく，市販のリアルタイムOSはライセンス料が高くてとても量産品に組み込むことができなかったうえ，大量のメモリを必要とする高機能仕様だったからです(**図2**)．

そのあとに出た16ビット版のMicrosoft社のMS-DOSも形だけはUNIX類似のコマンドが使えましたが，当然ながらシングル・プロセスのOSでした．

当時開発を手伝っていた，リアルタイム処理を要求する工作機械の制御プログラムはMS-DOS上で動かしたため，デバッグにはたいへん苦労しました．この制御にMTOSというリアルタイムOSが使えないか検討したことがありましたが，性能要求を満たすとコストが合わなくなり断念しました．

実際にリアルタイムOSの利点は，モニタ・プログラムのタスクを独立に持てるというような低水準のものではなく，数多くのI/Oからのms以下の水準の高速な処理応答要求に対応できる点に意味があります．そして，ゲーム機に搭載したような簡単なOSですらたいへんな効果を発揮するところがミソです．

いろいろなリアルタイムOS

●リアルタイムOSは競合の世界

リアルタイムOSは本誌2004年5月号の特集でも取り上げられているように，いろいろな種類が作られています．その中でもOS-9，VxWorks，OSEK/VDX，ITRONなどが比較的よく知られています．そのほかPC上で動くものを含めるとLynx OS，RT-Linux，QNX，Windows CEなどがあります．

良くも悪くも競争が成り立っています．特定の種だけが栄えて，多様化されていた遺伝子の種類が減る状況は，自然界では異常です．予想もしなかった原因での大絶滅の危機が襲ってくる可能性が高くなります．その意味ではリアルタイムOSの世界は健全であるといえます(**図3**)．

● OS-9

OS-9(Operating System for 6809)はMacOS 9とは違います．もっと歴史が古く，Motorola社から発売されたCPU 6809を産業用の組み込み制御に使う際のリアルタイムOSとして旧Microware Systems社から1981年に発売されました．その後，16ビットCPU 68000が出たため移植され，CD-Iの制御に使われた以外は，自動車や空調システム，エレベータ制御など，おもに家電機器以外の世界で使われ続けました．

OS-9はアセンブラで書かれていたのですが，Cで書き直されたOS-9000はx86(80386以降)やMIPS，PowerPC，SH-3などのCPUにも移植されており，リアルタイムOSの原則を押さえて作られているため，その使いやすさと動作の安定性には定評があります．

ただ，基本構想の時期から四半世紀ほど経っているので，仕様上，いろいろと不満なところもないわけでもありません．仮想記憶が使えない，プログラムをモジュール単位で容易に不正コピーされてしまう，などの点で不満が出ることがありました．

以下の特徴はOS-9/9000ファンの方々が利点として挙げているものから抜粋しました．

1) すべてにモジュール構造を採用

2）ROM 化可能
3）カーネル以外のモジュールをダイナミックに追加・削除・更新可能
4）UNIX のような階層的なファイル構造を持ち I/O をファイルとして処理
5）マルチプロセスによるプリエンプティブ・マルチタスク

OS-9 が出た当時は CPU の性能に比べて仕様が重く，爆発的に広がるという状況ではありませんでしたが，日本では FM-7 や X68000 などの 68 系パソコンの OS としても採用されました．現在でも，OS-9 系の組み込みシステムの開発には，これらのパソコンやエミュレータが使われています．

● VxWorks

VxWorks は Wind River 社が発売しているリアルタイム OS です．1987 年に発売された OS で，開発当初は映像関係の処理を行うリアルタイム制御がターゲットでした．

航空機や火星探索機など軍事・宇宙航空関係の制御に多く使われていて，その高速性と安定度は定評があります．Boeing 社の次世代航空機 7E7 にも採用が決定しています．OS-9 や Windows，Linux のように独立したプロセスでのマルチタスクではなく，ITRON と同じようにスレッドで動いています．

一般にスレッドで処理するほうがプロセスをタスクごとに立てるよりは動作を速くできます．スレッド間のデータのやりとりも簡単ですが，それだけタスクごとの分離が甘くなります．

● OSEK/VDX

この OS は欧州の電機業界と自動車業界が主体となって仕様を策定したものです．車載 LAN の規格である CAN との併用を図って，米国企業の支配から脱しようという意図もあります．

車載に力を入れているという意味では汎用の OS とはいえませんが，基本的に下位層のハードウェアから独立した OS で，8 ビットから 32 ビットまでの車載に使われるどの CPU にも搭載可能です．

1995 年に OSEK と VDK（もともとは独仏で別の仕様だった）を結合した最初の仕様が公開されました．実装は各 OS ベンダに任されており，仕様だけという形態は ITRON と同じです．その特徴は以下のようであると公表されています．

1）明確なコストの削減と開発時間の短縮
2）いろいろな会社の制御装置のソフトウェアの質の向上
3）異なった設計思想の制御装置間に標準化されたインターフェースを提供
4）車の各所に配置された処理機能（機器）を順次利用することで，追加のハードウェアなしでシステム全体のパフォーマンスが向上
5）OS の個々の実装からは完全に独立した仕様を提供し，実装面の仕様は何も規定されていない

自動車各社の車内部のデータ伝送規約やそこで使われる OS を統一することで分散処理も可能とする目的があるようです．これにより，各社の重複投資を避けて自動車各社の標準機器だけではなく，市場から広く安価に器材を調達できるようにという，経済的な側面も伺われます．

表 1　組み込みシステムにおけるリアルタイム OS の利用動向に関するアンケート調査報告書より（トロン協会・日本システムハウス協会）

OS の種類	割合（%）	OS の種類	割合（%）
市販 ITRON	24.9	Windows（CE 以外）	3.4
自社製 ITRON	10.9	Windows CE	2.8
フリー ITRON	2.8	MS-DOS	0.6
ITRON 合計	38.6	Microsoft 社合計	6.8
RTOS 不使用	16.2	OS-9	2.8
Linux	10.9	ほかの市販 OS	9
自社独自仕様	7.2	ほかのフリーの OS	1.2
VxWorks	7.2		

ITRON

● ITRON は仕様であって実装は各社まかせ

トロン協会では去る 6 月 2 日に 20 周年記念行事を催しました．

ITRON は 1984 年に始まった TRON 計画の一環として仕様が策定されました．しかしそれは仕様を提示しただけで，具体的な実装については何も規定していませんでした．それでも特定の会社が開発した OS や，特に米国政府の息がかかった OS を使うと，営業上何かと不利益を被るところが多かった日本の製造業各社は積極的に ITRON の利用を進めました．

● ITRON への期待はまだまだ増大

2002 年度にトロン協会がまとめた日本における組み込み用リアルタイム OS の利用状況の調査では，ITRON が**表 1** のように市販・自社製・フリーを含めて約 4 割のシェアを占めています．この調査には筆者も回答を送ったことがあります．

ITRON の利用率は調査のたびに少しずつ伸びてきており，新規の採用希望の調査では 5 割を越えています．実装の面でもフリーの ITRON としての TOPPERS プロジェクトや T-Engine 計画の T-Kernel に期待が寄せられているようです．

● ITRON の採用と不満の意見

ITRON は以下の観点から日本の多くの開発者の支持を得てきました．

1）OS の仕様や使用条件が特定の会社の利益や特定の国の政策に縛られない
2）仕様が公開されているので，既成のものでも自分で手を加えることができる
3）多くのベンダから種々の実装が提供されるので，取捨選択の余地がある
4）仕様段階ではライセンス料の支払いが不要である
5）坂村氏が提唱するユビキタス社会へのフィッティングが期待できる

ただ，米国や欧州では，Linux や GNU などを生み出したコ

ミュニティの土壌が本来あるにもかかわらず，なぜかITRONはあまり支持されていません．インターフェースの仕様だけで実装については何も語らないというのは，仕様で権利を主張する欧米人の目からみると中途半端と映るのでしょうか．また，アジア人が提唱したことへの偏見もあるのかもしれません．

さらに，コミュニティが生み出した財産もすぐに個人の幸せに結びつけるという狩猟民族的な行動様式が，GNUのStallman氏のようにあくまでも仕様の開放をめざすTRONプロジェクトと合わないのかもしれません(**図4**)．

同じ意味では，中国をTRONやT-Engineプロジェクトの仲間に引き入れようとする努力も，中国人の一族主義や商業主義の文化との関係を考慮に入れなければならないかもしれません．

トロン協会の調査ではITRONへの希望として以下のような項目が挙げられています．

1) ソフトウェア・モジュールの標準化
2) 開発環境とのインターフェースの標準化
3) C++/Javaの標準バインド
4) デバイス・ドライバの標準化

ユーザはITRONがゆるい仕様として規定していないところを求めているようです．これらを解決したITRONがよいかどうかは不明ですが，その対応がITRONの将来を決めるともいえるでしょう．

ただ，リアルタイムOSはWindowsなどのエンド・ユーザが直接触るOSとは異なり，専門家が採用を決定するOSなので，大衆的な皮膚感覚だけでは将来を占うことはできません．あえて占うと，組み込み用のリアルタイムOSの世界はRT-Linux(Embedded Linux)とITRON/T-Kernelの併用になっていくかもしれません．

期待されるITRONの標準実装とT-Engine計画

● ITRONの実装

ITRONの実装は多くの商用実装や自社専用実装があります．その中で半導体メーカの富士通，日電，ルネサス，東芝などが自社のCPU向けに専用のITRONを出しているほかに，いくつかの専業ベンダもあります．

フリーな実装としては高田 広章氏が進めて来たTOPPERSが有名です．もともとは高田氏が豊橋技術科学大学の教授だったときに始めたものですが，今はTOPPERSプロジェクトとしてNPO法人で運営されています．

実装に当たってカーネルをターゲットCPUに依存する部分と独立な部分を分けて，移植性を高めています．また，開発環境はフリーなものだけで完結するようになっており，LinuxやWindows上で動作シミュレーションが行えます．

● TRONチップとT-Engine計画

TRON計画は当初から専用のCPUの開発もめざしてきました．

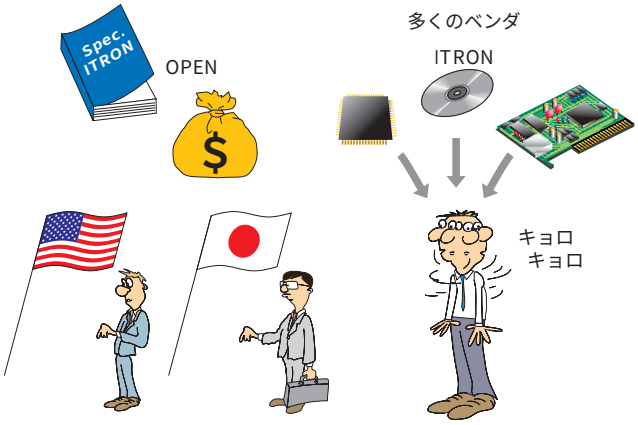

図4　ITRONが歓迎された理由

すなわち現在のIntelチップとWindowsの関係のように，ソフトウェア開発との一体化が目的でした．その方針に乗って1980年代後半から1990年代初めにかけてG_{MICRO}やM16などのTRONチップが作られました．筆者もリアルタイム性が強く要求されるFA用途に使おうと考えましたが，チップの供給と開発環境の点であきらめた経緯があります．

しかし，TRONチップの仕様が当時流行したRISCではなくCISCだったこともあり，M16などのローエンドの組み込み用チップ以外はあまり採用されませんでした．ただ思想自体はSHなど多くのRISCチップにも使われています．

T-Engine計画では，ハードウェアの内部仕様は厳密に規定していませんが，TRON計画よりはインターフェースの仕様がかなりはっきりと決めてあります．いくつかのベンダからARMやSHなどの既存のチップをコアにしたT-Engineが発表されるなど，G_{MICRO}とは違う動向があり，今後大いなる利用が期待されます．

● T-Kernel

T-KernelはT-Engine計画の中で従来のITRONが担ってきた部分をより強力な標準化で移植性を確保するものです．そのため機種依存部分をT-Monitorなどの別の基盤として分離してあります．

詳しい解説は本誌の記事をご覧いただくとして，今後のリアルタイムOSの世界に新しい展開が期待されます．

いかい・くにお　工学博士，(株)エム・アイ・ベンチャー