TOP－1オペレーティング・システム
（５）デバッグ環境

3P-8

白鳥敏幸、森山孝男、河内谷清久仁、山崎秘砂、穂積元一
日本アイ・ビー・エム株式会社　東京基礎研究所

1　はじめに

ＴＯＰ－１ ＯＳ は、小規模並列処理の研究を目的とした高機能マルチ・プロセッサ・ワークステーションＴＯＰ－１のオペレーティング・システムである。[1,2,3,4] マルチ・プロセッサ対応のＯＳを開発するにあたり、シングル・プロセッサ対応の既存のデバッガでは十分なデバッグが行えない事は容易に想定される。

本稿では、ＴＯＰ－１ ＯＳ のデバッグ環境について説明する。

2　TOP－1

2．1　TOP－1 ハードウェア

ＴＯＰ－１は、１０枚のプロセッサ・カード、共有メモリ、ハード・ディスク制御カード、ハード・ディスク、Ｉ／Ｏインターフェース・カードから成り、Ｉ／Ｏ装置として、ＩＢＭ(R) ＰＳ／５５(TM) モデル５５７０がＩ／Ｏインターフェース・カードに接続される（図１）。

2．2　TOP－1 OS

ＴＯＰ－１ ＯＳ は、ＩＢＭ ＡＩＸ ＰＳ／２(TM) をもとに並列処理機能を追加した、ＴＯＰ－１専用のオペレーティング・システムである。現在のＴＯＰ－１ ＯＳは、ハード・ディスク制御カードが接続されているプロセッサ・カード（カーネル・プロセッサ）上でカーネル・モードのプログラムが走り、それ以外のプロセッサ・カード（ユーザ・プロセッサ）上ではユーザ・モードのプログラムが走るマスタ・スレーブ型の並列オペレーティング・システムである。

５５７０上では、ＴＯＰ－１ ＯＳの入出力をサポートするために機能を拡張したＡＩＸ ＰＳ／２あるいはＤＯＳが走る。

3　デバッガ

ＵＮＩＸ(R) におけるデバッガは、

ユーザ・プログラム用デバッガ

カーネル用デバッガ（ネイティブ・デバッガ）

の２種類に大別できる。

今回は、マルチ・プロセッサ対応のカーネルの開発援助用であるので、カーネル専用のデバッガで必要最低限の機能のみを持たせることにした。

カーネルをデバッグする際には、

（１）ブレーク・ポイントを設定し、ブレーク時にネイティブ・デバッガを用いて、メモリ、各種構造体などの内容を表示させ、バグの原因を探すとともに、誤っているデータを更新し、実行を継続させる。

（２）デバッグ用のプリント文をソース・コード内に埋込み、実行過程をトレースし、さらに、アサート文を用いて、誤った実行を停止させる。

の二つの方法が、頻繁に行なわれると思われる。今回は、開発期間が非常に短かった事もあり、トレースの機能を実現することを主目的とした。

3．1　トレース

現在のＴＯＰ－１ ＯＳ においては、マスタ・スレーブ型ではあるが、ユーザ・プロセッサ上でも、ディスパッチャ、トラップ・ハンドラ等のカーネル・モードのプログラムの一部が走っている。このため、カーネル内のプリント文の出力をトレースする場合に、どのプロセッサ上で実行された結果であるかを知る必要がある。一案として、出力内容にプロセッサ番号を付加する方法が考えられる。この場合、特定の１台のプロセッサから出力させることにすると、そのプロセッサの振舞いが他のプロセッサのものと異なってしまう。この状況を避けるためにトレース結果の表示は各プロセッサでローカルに処理できるようにしたい。ＴＯＰ－１では他のプロセッサに悪影響を与えないで利用できるデバイス、つまり各プロセッサにローカルなデバイスとして、ＬＥＤと非同期シリアル・ポートが用意されている。しかし、ＬＥＤでは情報量が極端に少なく、また、シリアル・ポートでは端末の台数、設置場所、表示速度等の問題があり、トレースの目的に使用するには不適当である。

これらの問題を解決するために、実メモリ上に仮想端末を実現し、各プロセッサからローカルにコントロールするシステムとした。この仮想端末の内容は５５７０上で走るブラウザによって実際のディスプレイに表示される（図２の上側四分の３，上から順にカーネル・プロセッサ、ユーザ・プロセッサ１番，２番に対応している）。

仮想端末は、ＴＯＰ－１から５５７０への出力用として８０文字×数百行のリング・バッファ、カーソルのポインタ、現在行のポインタ，５５７０からＴＯＰ－１への入力用として数十文字のリング・バッファ、書き込み位置、読み出し位置のポインタ、文字数カウンタ、スピンロック用

TOP-1 Operating System (5) Debugging Environment
Toshiyuki SHIRATORI, Takao MORIYAMA, Kiyokuni KAWACHIYA, Hisa YAMASAKI, and Motokazu HOZUMI
IBM Research, Tokyo Research Laboratory, IBM Japan, Ltd.

フラグから成る。ブラウザは、出力用のリング・バッファ内の任意の位置を覗く事ができる。このため、リング・バッファ内にある限りは、実行結果を過去にさかのぼって確認することができる。

このトレース用仮想端末からの表示、および以下で述べるメモリ・ダンプ等の機能はいずれもＴＯＰ－１　ＯＳの動作とは無関係に動作し、５５７０上に無限ループで実現されている。このため、非常に大量のデータが出力された場合、プロセッサの性能差等から、文字落ち等が起こる場合も考えられる。しかし、実用上の問題は発生しておらず、逆に、メモリ・ダンプ時においては、メモリの内容が変更される様子が動的に監視できるので有効な方法と思われる。

３．２　メモリ・ダンプ

５５７０は、他のプロセッサ・カードと同様に共有メモリ内の任意のアドレスを読み書きできる。これを利用して、５５７０上のブラウザで任意のアドレスのメモリの内容を１６進数およびＡＳＣＩＩ文字で表示し、さらに、任意のアドレスのメモリの内容を複数箇所同時に、つまり不可分に書き換える機能を実現した。メモリの内容を書き換える際には、ＴＯＰ－１のハードウェアの持つバス・ロックの機能を用いて、５５７０以外のプロセッサ・カードからのバス・アクセスを禁止している。

３．３　動作状況モニタ

カーネルのデバッグがある程度まで進むと、トレース、メモリ・ダンプ等ではシステムの動作状況が十分に把握できなくなってくる。このため、必要に応じて１台のプロセッサを専用に割当て、仮想端末に各プロセッサ、プロセス等の状況を表示させている（図３の下側）。図３では、システム中の各プロセス・キュー（スリープ・プロセス・キュー、カーネル・プロセッサ用のレディ・キュー、ユーザ・プロセッサ用のレディ・キュー）内、またはプロセッサ（左から順にカーネル・プロセッサ，ユーザ・プロセッサ１番、２番、…）上のプロセス番号、プライオリティ、プロセッサの状態などが表示されている。

４　おわりに

ＴＯＰ－１　ＯＳのデバッグ環境の現状について説明した。現時点では、機能的に不十分な点も多いが、ＴＯＰ－１　ＯＳの今後の改良と併せて、発展させていく予定である。

ＩＢＭは、ＩＢＭ Ｃｏｒｐ．(米)の登録商標。
ＡＩＸ ＰＳ／２，ＰＳ／５５は、ＩＢＭ Ｃｏｒｐ．(米)の商標。
ＵＮＩＸは、ＡＴ＆Ｔ(米)の登録商標。

【参考文献】

［１］大庭他：「高速並列処理ワークステーション（ＴＯＰ－１）－アーキテクチャー」、情報処理学会第37回全国大会論文集(1988)、7N-2、pp. 172

［２］中田他：「同上　－入出力システム－」、同上、7N-7、pp. 182

［３］穂積他：「ＴＯＰ－１オペレーティングシステム（１）基本方針」、情報処理学会第39回全国大会論文集(1989)

［４］山崎他：「同上（２）プロセス・スケジューリング」、同上

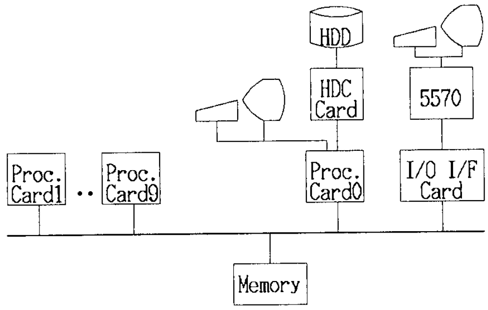

図１　ＴＯＰ－１ハードウェア

```
pnattach:
grinit: called
grinit: set callout
vraminitl: KEY=0x19630217, BASE=0x0, SIZE=0x40000
vraminitl: KEY=0x19630218, BASE=0x40000, SIZE=0x100000
!!!! tscall by KPU !!!!!!
trap: invalid address ? 004046e0
vcprintmap: procp=f10ce05c
        pvsp=f0196014 vsp=f018b320 : 00000000 - 0000d7b4
        pvsp=f0195fe4 vsp=f018b1f4 : 00400000 - 00404000
        pvsp=f0195fd4 vsp=f018b190 : 1ffff000 - 20000000

slave_sched: entered. thispu = 2
slave_sched: waiting to take off !
slave_sched: resume to proc#22531

0021E050  C4 0C 85 C0 0F 84 CE FF FF FF 8A 47 5C 84 C0 0F
0021E060  84 07 00 00 00 C7 45 F4 FF FF FF FF C6 47 5D 19
0021E070  66 8B 45 FC 66 89 47 54 C6 47 5C 00 8B 45 F8 89
0021E080  47 68 FF 75 F8 E8 82 5E 02 00 59 89 47 18 8B 45
0021E090  F4 89 47 64 8D 47 70 8D 56 0C 8B 0A 89 08 8B 4A
0021E0A0  04 89 48 04 0F BE 46 5D 40 88 87 80 00 00 00 6A
 [ 0]  [ 1]  [ 2]  [11]
```

図２　ブラウザの出力例（トレース、メモリ・ダンプ）

```
00210300  00 00 00 8B 45 FC FF 30 68 FB B6 1B F0 E8 62 10
00210310  04 00 83 C4 08 E8 D6 E1 03 00 85 C0 0F 84 02 01
00210320  00 00 8B 45 FC 8B 00 3D 01 00 00 00 0F 84 F7 00
00210330  00 00 3D 02 00 00 00 0F 84 47 01 00 00 3D 03 00
00210340  00 00 0F 84 10 02 00 00 3D 10 00 00 00 0F 84 05
00210350  01 00 00 3D 11 00 00 00 0F 85 C2 02 00 00 A1 DC
00210360  D2 7F F0 89 45 D0 C7 45 EC 12 B7 1B F0 B8 12 B7
00210370  1B F0 A3 04 D3 7F F0 C7 45 F0 00 00 00 00 C7 45
pninit:
pnattach:
grinit: called
grinit: set callout
vraminitl: KEY=0x19630217, BASE=0x0, SIZE=0x40000
vraminitl: KEY=0x19630218, BASE=0x40000, SIZE=0x100000
!!!! tscall by KPU !!!!!!

sleep queue:22590  22589  22580  22565  22561     1  22567     2  22596  22598
  22594  22592      0  22563  22591  22573  22599  22595  22597  22593
runq   22629(54,54)  22633(54,54)  22634(54,54)  22632(54,54)
urunq  22632(54,54)
22629÷ 22637÷ 22638÷ 22635÷ 22630÷ 22639÷ 22632÷ 22641÷ 22640÷ 2)
 54 54  54 54  54 54  54 54  54 54  54 54  54 54  50 50  50 50
 [11]  [ 0]  [10]
```

図３　ブラウザの出力例（モニタ）