ＡＩＰ－ｄｂｘの実現法

4K-6

今井　徹　　斎藤　光男
（株）東芝　総合研究所

１．はじめに

ＡＩＰ（ＡＩプロセッサ）[1],[2] は，Ｐｒｏｌｏｇ，ＬＩＳＰを高速に実行するために開発されたプロセッサであるが，汎用言語であるＣをサポートすることにより，統合的な言語環境を提供することを目指している．

ＡＩＰは，ＡＳ３０００とＶＭＥバスにより結合された，アタッチドプロセッサとして実現されており，ＡＳ３０００と同等の環境でのソフトウェアの開発，実行が可能なため，デバッガもＡＳ３０００同様にｄｂｘをサポートすることとした．

このＡＩＰ－ｄｂｘはアタッチドプロセッサにおけるＯＳ開発用デバッガという側面を持っている．

本稿では，ＡＩＰにおけるＣのデバッガであるＡＩＰ－ｄｂｘの実現法について述べる．

２．実現形態

ＡＩＰはアタッチドプロセッサとして実現されており，Ｉ／ＯはＡＳ３０００上のサーバプロセスに対して要求を発生するが，それ以外はＡＩＰ内部で処理するという形態で，プログラムを実行する．

また，専用のメモリを持ったプロセッサであり，ＡＳ３０００の管理を受けることなくメモリ等のリソースを使用することができる．

そこで，ＡＩＰ上にＯＳを実現することにより，ＡＩＰでのマルチプロセスやリソースの管理を，自立的に行なうことができる．

ＡＩＰ－ｄｂｘは，アプリケーションの開発のみならず，アタッチドプロセッサにおけるＯＳ開発用のデバッガという位置づけをすることができ，この側面を考慮して実現形態を決定した．

まず，ＡＩＰにデバッガをロードし，セルフで動作させる方式ではなく，ＡＳ３０００上で動作し，ＡＩＰ上のプログラムをデバッグする，クロスデバッガとした．（図１）

これは，

- ＡＩＰ上のＯＳのメモリ管理に対して制約を与えない
- デバッグ対象のプログラムの暴走によりデバッガが破壊されることがない
- デバッガ開発に，ＡＳ３０００の開発環境を使用できる
- ＡＳ３０００からインタラプトをかけることにより，デバッグプロセスだけでなく，ＡＩＰ自体の実行を中断することができる

からである．

また，デバッガからＡＩＰの持つ任意のリソースに対するアクセスを可能とした．

特にメモリに関しては，物理アドレスでの指定を可能とした．

これらの機能により，ＭＭＵ等のモニタリングを可能とし，開発の自由度を高めた．

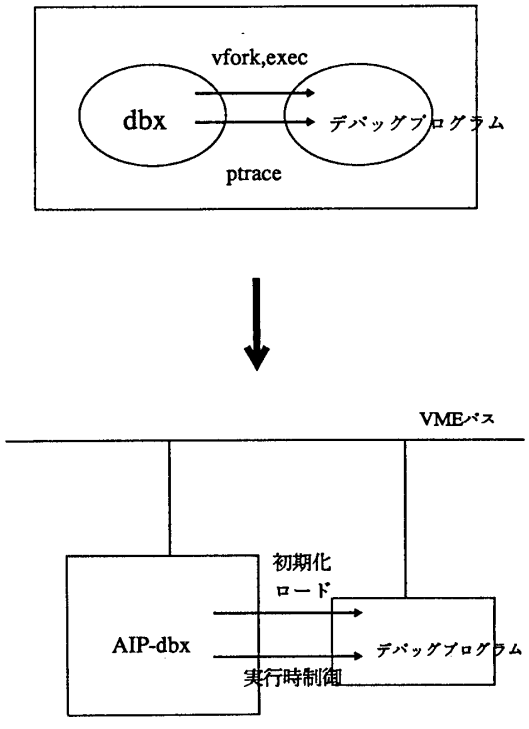

図１　ＡＩＰ－ｄｂｘの実現形態

Implementation of AIP-dbx
Toru IMAI, Mitsuo SAITO
TOSHIBA Corporation

3．主な改造点

ｄｂｘは，同じＯＳの下で子プロセスを生成してデバッグ対象のプログラムを実行し，プロセス間通信により制御を行なうが，ＡＩＰ－ｄｂｘは，前項に述べるように，異なるプロセッサにデバッガとデバッグ対象のプログラムを割り当てるため，システムコールを発生することよるＯＳのサポートを受けることができない．

そのため，以下のような改造が必要となった．

デバッグプログラムの実行の開始

ｄｂｘは，fork,exec によりデバッグプログラムの実行を開始するが，ＡＩＰ－ｄｂｘでは，MMU のセット，レジスタ，メモリのクリア，マイクロプログラムおよびユーザプログラムのロード，スタックの設定を．デバッガから直接行った．

デバッグプログラムの制御

ｄｂｘは，デバッグプログラムのレジスタ，メモリのread,write，ブレークポイントの設定，プログラムの続行，ステップ実行等の制御をptraceを呼ぶことにより行なうが，ＡＩＰはコピーバック方式のキャッシュを採用しており，またパイプライン処理をしているため，ブレークポイント設定によるデバッグプログラムの中断，再開時にキャッシュのフラッシュや，パイプクリアが必要である．

（例）ブレークポイントを設定し，そこで止まった後，再開する場合

- ブレークアドレスを計算し，該当命令を退避した後，ブレーク命令と置き換える．
- ＡＩＰに起動をかけると，bkp で止まる．
- キャッシュをフラッシュする．
- 命令を復帰し，パイプをクリアし，pcを戻す．
- ＡＩＰに再起動をかける．

デバッグプログラムからのＩ／Ｏ要求の処理

ＡＩＰは直接Ｉ／Ｏの手段を持たないため，ＡＩＰ－ｄｂｘは，Ｉ／Ｏサーバプロセスとしての機能も包含することにした．（図２）

ＡＩＰメモリの特定領域をＩ／Ｏバッファとし，Ｉ／Ｏ要求がおこると，その種類と引数をＡＩＰがセットし，ＡＩＰ－ｄｂｘに対し割り込みを掛ける．

これをＡＳ３０００で処理した後，リターン値をＩ／Ｏバッファにセットして，ＡＩＰに対し制御を渡す．

メモリ書き込み検出機能の高速化

ｄｂｘには，ある変数の値が変わった場合や，ある条件が成立した場合に実行を中断する機能があり，実行時のデータの破壊の検出に有効である．

ところが，１命令ごとに調べるため，実行速度が著しく遅く，実用的でない．

ＡＩＰ－ｄｂｘでは，ハードウェアのサポートにより，実行速度を落とすことなく検出することができる．

4．おわりに

本稿では，ＯＳ開発を意識したデバッガの実現法を示した．

ＯＳのような制御構造の複雑なソフトウェアを開発する際には，開発環境の整備が効率の向上につながる．

ＡＩＰ－ｄｂｘを用いることにより，ＡＳ３０００の環境を生かしつつ，シンボリックデバッガによるＯＳのブートストラップのモニタリングが可能となった．

参考文献

[1] 斎藤他　：　ＡＩワークステーション（ＷＩＮＥ）の開発Ｉ－Ⅶ　情報処理学会第３５回全国大会（１９８７）

[2] 斎藤他　：　ＡＩワークステーションの開発思想　人工知能学会第１回全国大会（１９８７）

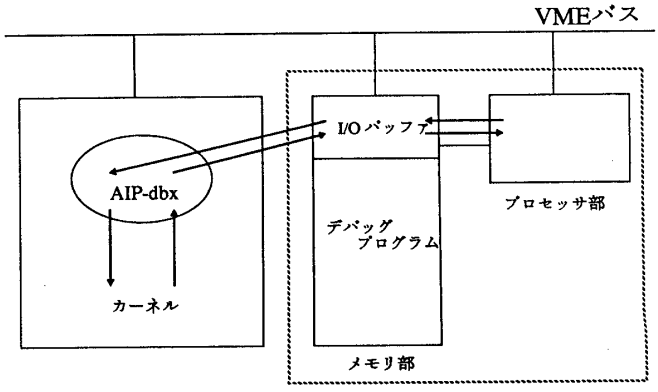

図２．　Ｉ／Ｏ要求の処理