Vol.2011-ARC-195 No.3
Vol.2011-OS-117 No.3
2011/4/13

AnT オペレーティングシステムの SH-4 への移植

鶴谷昌弘[†1]　山内利宏[†2]　谷口秀夫[†2]

AnT オペレーティングシステムは，マイクロカーネル構造を有し，高い適応性と堅牢性を実現している．従来の ***AnT*** は，IA-32 アーキテクチャを採用した Pentium4 プロセッサ上のみで動作し，他のプロセッサには対応していない．そこで，***AnT*** の適応域を拡大するため，***AnT*** を代表的な組み込みプロセッサの 1 つである SH-4 プロセッサへ移植した．ここでは，起動処理，例外処理と割り込み処理，およびメモリ管理について，移植のために改造した内容を述べる．また，SH-4 プロセッサへ移植した ***AnT*** の性能評価の結果について報告する．

Porting *AnT* operating system to SH-4

Masahiro Tsuruya,[†1] Toshihiro Yamauchi[†2]
and Hideo Taniguchi [†2]

AnT operating system is based on microkernel architecture, and realizes high adaptability and toughness. ***AnT*** operating system works on Pentium4 processor based on IA-32 architecture, and did not support other processor. We have ported ***AnT*** operating system to SH-4 processor that is the one of the representative embedded processors. This paper describes the modification for porting of start processing, exception processing and interruption processing, and memory management. In addition, this paper reports the performance of ***AnT*** operating system on SH-4 processor.

†1 岡山大学工学部
Faculty of Engineering, Okayama University
†2 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University

1. はじめに

計算機が提供するサービスの高度化とともに，それを支える基盤ソフトウェア（以降，OS と略す）は複雑化している．既存 OS のプログラム構造の 1 つであるモノリシックカーネル構造は，OS 機能をカーネルとして一体化して実装するため，新しい機能の追加や削除が容易でない．これに対し，マイクロカーネル構造は，カーネルに OS 機能の必要最小限な機能のみを持たせ，その他の OS 機能をプロセスとして動作させるため，OS 機能の追加や削除が容易である．このため，マイクロカーネル構造を持つ OS の研究が行われている[1)2)3)]．

我々は，マイクロカーネル構造を有する ***AnT*** オペレーティングシステム[4)5)6)]（An operating system with adaptability and toughness）（以降，***AnT*** と略す）を開発している．しかし，従来の ***AnT*** は，IA-32 アーキテクチャ[7)] を採用した Pentium4 プロセッサ上で動作し，他のプロセッサには対応していない．そこで，***AnT*** を代表的な組み込みプロセッサの 1 つである SH-4 プロセッサ[8)] へ移植し，***AnT*** の適応域拡大を目指した．

ここでは，***AnT*** が利用しているハードウェア機能を明確化し，起動処理，例外処理と割り込み処理，およびメモリ管理について，移植のために改造した内容を述べる．また，SH-4 プロセッサへ移植した ***AnT*** の性能評価の結果について報告する．

2. *AnT* オペレーティングシステム

2.1 適応性と堅牢性

AnT は，計算機システムの開発者と利用者の両者に「使いやすい」かつ「壊れにくい」という特徴を有する基盤ソフトウェアであることを目指している．

開発者にとって「使いやすい」ためには，新たなサービス（機能）の追加，つまり新たなプログラムの追加を容易にできる必要がある．特に，組込みシステムの場合，新たなドライバプログラムの組込みが頻繁に発生する．このため，ドライバプログラムの新規作成の容易化だけではなく既存プログラムの流用ができれば好ましい．利用者にとって「使いやすい」ためには，高性能であるとともに，利用したいサービスを簡単に利用できる必要がある．したがって，両者の要求を満足するには，基盤ソフトウェアは高い適応性を有することが求められる．

そこで，高い適応性を有する基盤ソフトウェアの構成法が必要になる．具体的には，サービス（機能）の組込みや取外しが簡単なプログラム構成法，および利用者の利用履歴を考慮したプログラム実行制御法が必要である．また，計算機システムの性能低下の大きな要因で

Vol.2011-ARC-195 No.3
Vol.2011-OS-117 No.3
2011/4/13

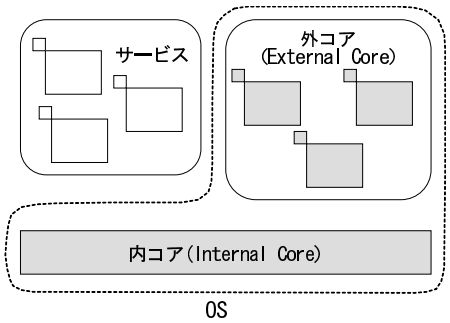


図 1 ***AnT*** の基本構造

あるメモリ間データ複写をゼロにする高速なプログラム間データ通信機構も必要である．

開発者にとって「壊れにくい」ためには，開発中のプログラムの暴走によりシステム全体が壊れることを防ぐ必要がある．利用者にとって「壊れにくい」ためには，いろいろなサービス（機能）を使っても相互干渉によって不都合な動作にならないような高い信頼性とともに，高いセキュリティが必要である．したがって，両者の要求を満足するには，基盤ソフトウェアは高い堅牢性を有することが求められる．

そこで，高い堅牢性を有する基盤ソフトウェアの構成法が必要になる．具体的には，組み込んだプログラムに不具合が発生しても他のプログラムやシステム全体に悪影響を与えない仕組み，およびネットワークを介したセキュリティ低下を防ぐ仕組みが必要である．

2.2 プログラム構造

AnT は，マイクロカーネル構造を有するオペレーティングシステムである．***AnT*** のプログラムは，OS とサービスからなる．この様子を**図 1** に示す．OS は，内コアとプロセスとして動作する外コアからなる．内コアは，最小のシステムの動作を保証するプログラム部分である．外コアは，システムの利用形態に適応するために必須なプログラム部分であり，動的に再構成可能な構造を有する．サービスは，サービスを提供するプログラム部分である．

3. 移植方式

3.1 移植方針

AnT の SH-4 プロセッサへの移植方針は，移植工数の削減である．

AnT は，必要最小限の OS 機能を内コアして実現しており，アーキテクチャの相違の影響を受ける OS 機能は，内コアの OS 機能であり，外コアの OS 機能は影響を受けない．また，内コアの OS 機能は，アーキテクチャに依存する部分と依存しない部分の大きく 2 つに分けることができる．このため，アーキテクチャに依存しない OS 機能は基本的に変更を加えず，アーキテクチャに依存する OS 機能を書き換えた際に影響を受ける処理は，必要最小限の変更を加えることとした．

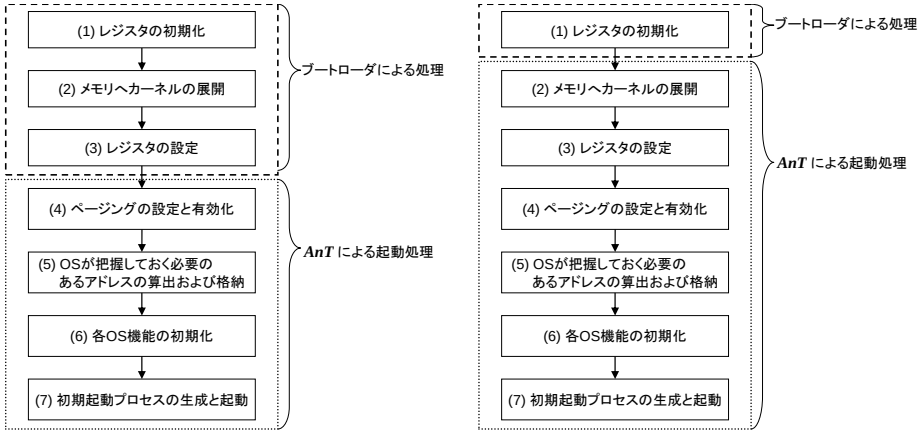


図 2 起動処理の流れ

3.2 基本的課題と対処

3.2.1 起動処理

Pentium4 プロセッサのハードウェア環境（以降，Pentium4 環境と略す）で動作する ***AnT***（以降，Pentium4_***AnT*** と略す）の起動処理を**図 2**（A）に示す．Pentium4 環境では，ブートローダは，レジスタの初期化，メモリへのカーネルの展開，およびレジスタの設定を行う．このため，Pentium4_***AnT*** の起動処理は，ページングの設定と有効化，OS が把握しておく必要のあるアドレスの算出および格納，各 OS 機能の初期化，および初期起動プロセスの生成と起動を行う．しかし，SH-4 プロセッサのハードウェア環境（以降，SH-4 環境と略す）では，図 2（B）に示すようにブートローダはレジスタの初期化しか行わない．このため，SH-4 環境で動作する ***AnT***（以降，SH-4_***AnT*** と略す）の起動処理は，メモリへのカーネルの展開とレジスタの設定も行う必要がある．

Vol.2011-ARC-195 No.3
Vol.2011-OS-117 No.3
2011/4/13

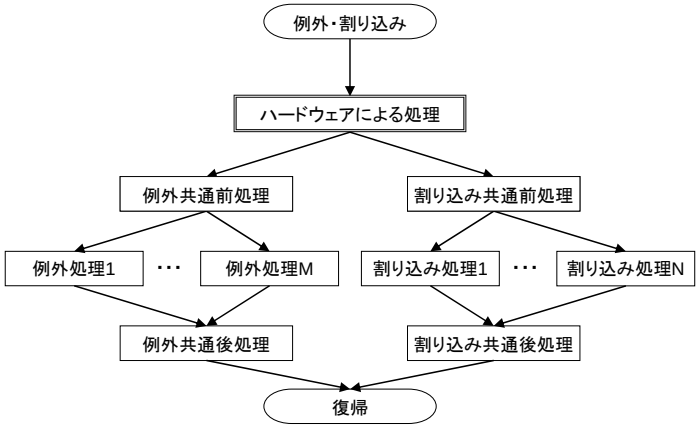


図 3 例外処理と割り込み処理の流れ (Pentium4_***AnT***)

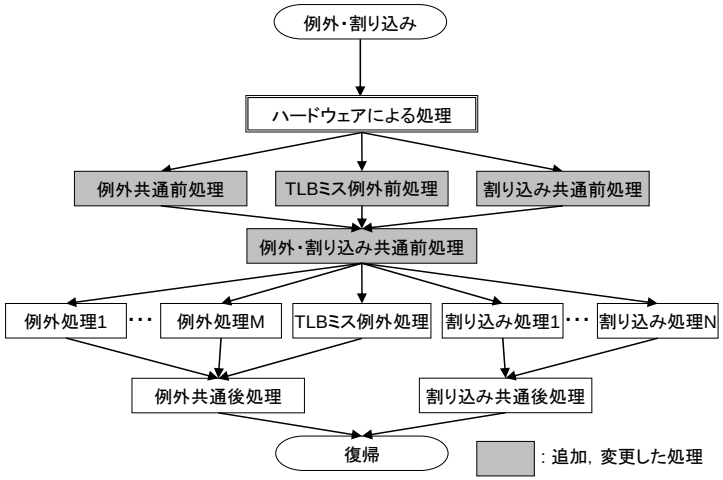


図 4 例外処理と割り込み処理の流れ (SH-4_***AnT***)

3.2.2 例外処理と割り込み処理

Pentium4_***AnT*** の例外処理と割り込み処理の流れを**図 3** に示す．Pentium4 プロセッサは，各例外と各割り込みを識別することができるため，各例外と各割り込みに対して，実行する処理のアドレスを例外・割り込みテーブルへ個別に設定することができる．Petium4_***AnT*** では，全ての例外に対して例外共通前処理，全ての割り込みに対して割り込み共通前処理へ移行するように例外・割り込みテーブルへ設定を行っている．処理の内容を以下に説明する．

(1)　例外共通前処理と割り込み共通前処理では，各前処理の必要に応じて，レジスタの退避，スタックの切り替え，戻りアドレスの設定，および割り込みマスクの設定を行い，各例外処理と割り込み処理へ移行する．例外処理と割り込み処理に対して前処理が異なっているのは，例外処理と割り込み処理に対してハードウェア処理が異なってるためである．

(2)　例外・割り込みの個別処理は，各例外や各割り込みに対する個別の処理を行う．

(3)　例外共通後処理では，スタックの切り替え，レジスタの復元，および割り込みマスクの設定を行い，例外の呼び出し元へ戻る．また，割り込み共通後処理では，スタックの切り替え，レジスタの復元，および割り込みマスクの設定を行い，再スケジュールを行う．

一方，SH-4 プロセッサは，各例外と各割り込みを，例外，割り込み，および TLB ミス例外と識別する．つまり，各例外と各割り込みを識別できない．また，Pentium4_***AnT*** とは異なり，TLB ミス例外が存在する．このため，SH-4_***AnT*** の例外処理と割り込み処理の流れを**図 4** のようにした．SH-4_***AnT*** では，例外，TLB ミス例外，および割り込みに対して，それぞれ例外共通前処理，TLB ミス例外前処理，および割り込み共通前処理へ移行するように設定する．処理の内容を以下に説明する．

(1)　例外共通前処理，TLB ミス例外前処理，および割り込み共通前処理では，各処理を行う上で必要なレジスタの設定を行う．Pentium4_***AnT*** では，例外処理と割り込み処理に対してハードウェア処理が異なっていたため，前処理が別となっていた．しかし，SH-4_***AnT*** では，例外処理と割り込み処理に対してハードウェア処理が共通であるため，例外・割り込み共通前処理とした．例外・割り込み共通前処理の内容は，Pentium4_***AnT*** の例外共通前処理と割り込み共通前処理と同様である．

(2)　例外・割り込みの共通処理では，各例外や各割り込みに対する個別の処理を行う．

(3)　例外共通後処理と割り込み共通後処理の内容は Pentium4_***AnT*** と同様である．

3.3 メモリ管理

3.3.1 メモリ空間の構成

Pentium4_***AnT*** のメモリ空間の構成[5] を**図 5** に示す．0 から 3GB をプロセス空間，3GB 以降をカーネル空間としている．プロセス空間の 0 から 2GB は，外コアやサービスのプログラムが利用する空間である．また，プロセス空間の 2GB から 3GB は，内コアと外コアとサービスのプログラムが相互のデータ通信に利用する空間である．この領域をコア間通信

Vol.2011-ARC-195 No.3
Vol.2011-OS-117 No.3
2011/4/13

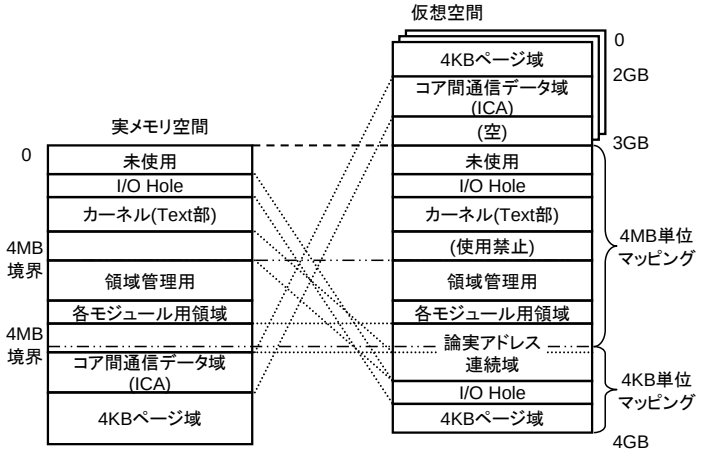


図 5 メモリ空間の構成 (Pentium4_***AnT***)

データ域（ICA : Inter-core Communication Area）[6] と呼ぶ．なお，カーネル空間にメモリマップド I/O 用の領域である I/O Hole を確保している．

一方，SH-4 プロセッサのメモリ空間は，外部メモリ空間と仮想アドレス空間からなり，Pentium4 プロセッサの実メモリ空間と仮想空間に対応する．また，仮想アドレス空間は，5 つの領域に分かれており，各領域で使用方法が限定される．このため，SH-4_***AnT*** のメモリ空間の構成を**図 6** のようにした．P0 領域（0 から 2GB の空間）をプロセス空間，P1 領域（2 から 2.5GB の空間）をカーネル空間とする．プロセス空間の 0 から 1.5GB は，外コアやサービスのプログラムが利用する空間である．また，プロセス空間の 1.5G から 2GB は，ICA とする．SH-4 のユーザモードでは，2GB 以降の空間にアクセスすることができない．これにより，カーネル空間を 2GB 以降（P1 領域）に配置することで，ユーザプロセスから保護する．また，SH-4 プロセッサは，メモリマップド I/O 用の領域として P4 領域を使用する．このため，メモリマップド I/O 用の領域である I/O Hole を確保する必要はない．外部メモリ空間はカーネル空間（P1 領域）へストレートマッピングされているため，外部メモリ空間のメモリが接続されているエリアのデータ構成がカーネル空間の構成となる．

Pentium4_***AnT*** と SH-4_***AnT*** のメモリ空間の違いを**表 1** に示す．プロセス空間，カーネル空間，および ICA については，SH-4_***AnT*** は Pentium4_***AnT*** よりもサイズが小さい．また，SH-4_***AnT*** では，利用するシステムのメモリ容量やメモリが接続されているエリアにより，カーネル空間のサイズや位置が変化する．

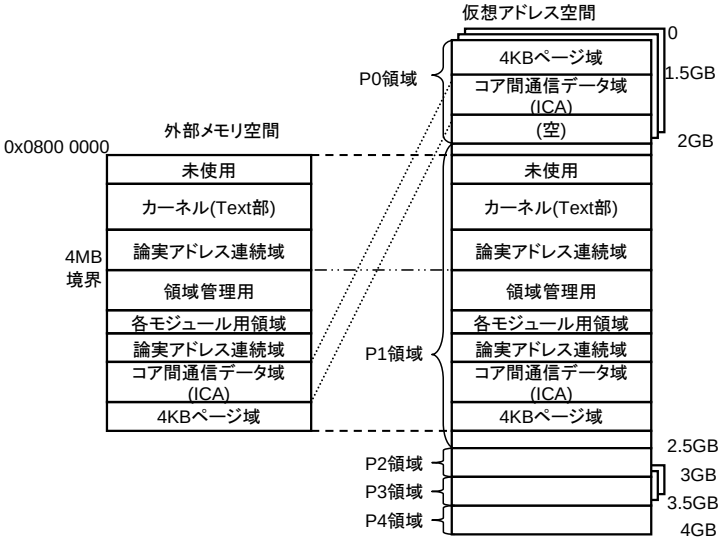


図 6 メモリ空間の構成 (SH-4_***AnT***)

表 1 メモリ空間の相違点

	Pentium4	SH-4
プロセス空間	0〜2GB の領域	0〜1.5GB の領域
カーネル空間	3〜4GB の領域	P1 領域の一部
コア間通信データ域（ICA）	2〜3GB の領域	1.5〜2GB の領域
メモリマップド I/O 用の領域	I/O Hole	P4 領域

3.3.2 アドレス変換

Pentium4_***AnT*** のアドレス変換の流れを**図 7** に示す．Pentium4 プロセッサは，TLB に該当エントリが存在しない場合，ハードウェアにより TLB へのエントリ登録を行う．また，現在走行している仮想空間のマッピング表の先頭アドレスは CR3 レジスタに格納されている．仮想空間へのアクセスがあり，TLB に該当エントリが存在しない場合，MMU が CR3 レジスタに格納されているアドレスをもとに該当するエントリを探索し，TLB への登録を行う．

Vol.2011-ARC-195 No.3
Vol.2011-OS-117 No.3
2011/4/13

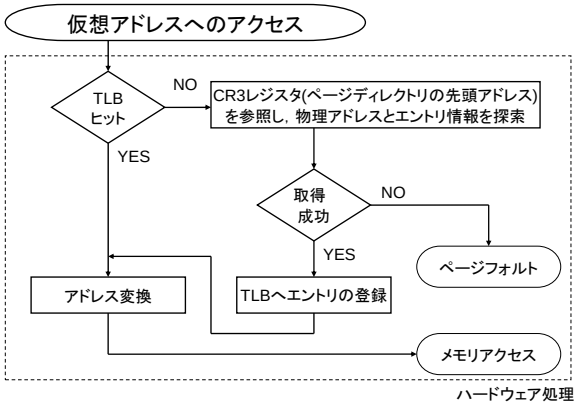


図 7 アドレス変換の流れ (Pentium4_***AnT***)

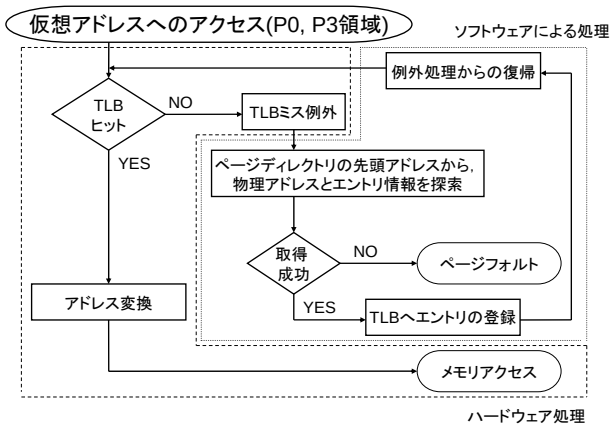


図 8 アドレス変換の流れ (SH-4_***AnT***)

一方，SH-4 プロセッサは，TLB へのエントリ登録を行わない．このため，TLB へのエントリ登録処理を OS が行う必要がある．SH-4_***AnT*** のアドレス変換の流れを**図 8** に示し，TLB ミス例外発生時に行う処理を**図 9** に示す．SH-4 プロセッサは，TLB に該当エントリが存在しない場合，TLB ミス例外を発生させる．この例外に対する処理として，SH-4_***AnT*** では TLB へのエントリ登録処理を行う．具体的には，TLB ミス例外の発生したアドレス

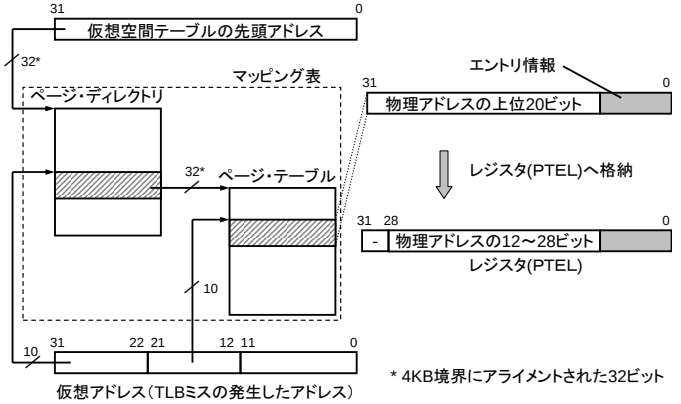
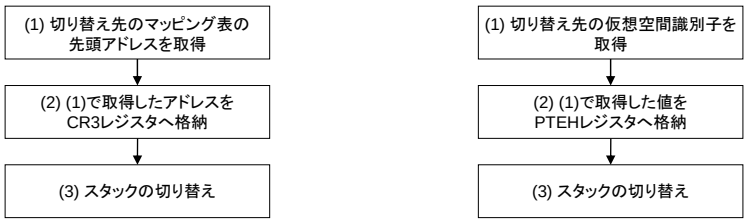


図 9 TLB ミス例外処理の流れ



（A） Pentium4_***AnT***　　（B） SH-4_***AnT***

図 10 仮想空間切り替えの流れ

に対応する物理アドレス（12 から 28 ビット）とエントリ情報をマッピング表から検索し，レジスタ（PTEL）へ格納する．次に TLB エントリへの登録を行う命令（LDTLB）を発行し，TLB へのエントリ登録を行う．なお，マッピング表の構成が関連する OS 機能への影響を最小限にするため，SH-4_***AnT*** で使用するマッピング表を Pentium4_***AnT*** と同様の構成とした．

3.3.3 仮想空間切り替え

AnT は，各プロセスが独立した仮想空間を保有する多重仮想記憶方式を用いている．このため，プロセス切り替えの際に仮想空間切り替えを行う．Pentium4_***AnT*** の仮想空間切り替えの流れを**図 10**（A）に示す．Pentium4 プロセッサは，現在利用している仮想空間の

Vol.2011-ARC-195 No.3
Vol.2011-OS-117 No.3
2011/4/13

マッピング表の先頭アドレスを CR3 レジスタへ格納している．このため，仮想空間切り替え処理では，切り替え先仮想空間のマッピング表の先頭アドレスを取得し，CR3 レジスタへ格納する．この際，Pentium4 プロセッサは，CR3 レジスタへ書き込みを行うと TLB をフラッシュするため，切り替え前の仮想空間のエントリが TLB に残ることはない．

一方，SH-4 プロセッサは，現在走行中の仮想空間識別子を PTEH レジスタへ格納しており，仮想空間を識別することができる．このため，SH-4_***AnT*** の仮想空間切り替え処理の流れを図 10（B）のようにした．SH-4_***AnT*** の仮想空間切り替え処理では，切り替え先の仮想空間識別を取得し，PTEH レジスタへ格納する．なお，SH-4 プロセッサは TLB の各エントリに仮想空間の識別子を保持させることができる．このため，SH-4 プロセッサは TLB に格納されているエントリがどの仮想空間のものであるかを識別することができるため，仮想空間切り替えの際に TLB フラッシュを行う必要がない．したがって，SH-4_***AnT*** は，Pentium4_***AnT*** に比べ，TLB を有効に利用できる場合があると考えられる．

4. 評　　価

4.1 評 価 項 目

移植工数と性能を評価する．具体的には，***AnT*** を SH-4 へ移植する際の工数を明らかにする．また，アーキテクチャの相違が ***AnT*** に与える影響を明確化するため，***AnT*** の特徴的な機能の 1 つである走行モード変更機構[9] に関する性能を評価する．走行モード変更機構とは，プロセスが走行中に，自身や他のプロセスの走行モードを任意のタイミング（システムコールを用いて）で変更可能な機構である．この機構により，プロセスの走行モードをスーパバイザモードにすることで，システムコールを関数呼び出しの形で行うことができる．このため，システムコールをソフトウェア割り込みで行う際に発生する走行モード変更のオーバヘッドを削減することが可能である．

4.2 移 植 工 数

移植により，***AnT*** ソースプログラム等の格納ディレクトリ構造をマルチアーキテクチャを考慮した構造に変更した．移植前のディレクトリ構造を**図 11** に示し，マルチアーキテクチャを考慮したディレクトリ構造を**図 12** に示す．移植前の構造は，OS 機能別に関連するファイルをディレクトリにまとめて格納する構造である．例えば，起動処理に関連するファイルを startup ディレクトへ格納している．これに対し，マルチアーキテクチャを考慮した構造にするために，以下のように変更した．

(1)　アーキテクチャに依存する OS 機能のディレクトリに common，i386，および sh4 のサブディレクトリを作成し，アーキテクチャに依存しないファイルを common ディレクトリへ格納し，アーキテクチャに依存するファイルをそれぞれ i386 と sh4 ディレクトリへ格納する．

(2)　アーキテクチャに依存しない OS 機能のディレクトリについては変更しない．

上記により，***AnT*** を複数のアーキテクチャへ対応させることを容易にした．なお，基本的に移植前の構造を保っているため，ディレクトリ構造の変更によるソースコードへの影響は小さい．

次に，コード量について考察する．移植前のファイル数とコード量を**表 2** に示し，移植後のファイル数とコード量を**表 3** に示す．また，改造した既存ファイルの数とコード量を**表 4** に示す．これらにより，以下のことがわかる．

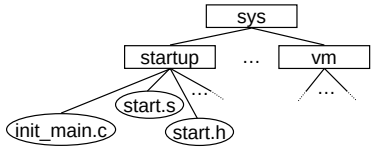


図 11　移植前のディレクトリ構造

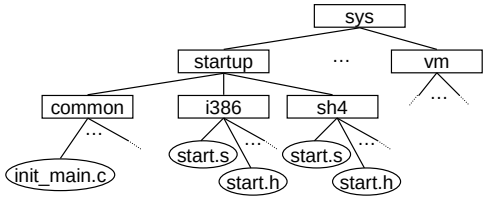


図 12　マルチアーキテクチャを考慮したディレクトリ構造

表 2　移植前のファイル数とコード量

記述言語	ファイル数	コード量（行数）
C 言語	126	15487
アセンブリ言語	11	2125

表 3　移植後のファイル数とコード量

ディレクトリ	共通（common）		i386		sh4（新規作成）	
記述言語	ファイル数	コード量	ファイル数	コード量	ファイル数	コード量
C 言語	68 （42）	8874 （6659）	58	7101	33	3328
アセンブリ言語	0	0	11	2125	12	1509

表 4　改造した既存ファイルの数とコード量

記述言語	ファイル数	コード量
C 言語	11	301
アセンブリ言語	0	0

Vol.2011-ARC-195 No.3
Vol.2011-OS-117 No.3
2011/4/13

表 5 システムコール発行に要する処理時間

プロセッサ	システムコール発行方式	クロックサイクル数	処理時間 (μs)	プロセスの走行モード
Pentium4	ソフトウェア割り込み	1242.0	0.52	ユーザモード
(2.4GHz)	関数呼び出し	438.4	0.18	スーパバイザモード
	sysenter 命令	456.4	0.19	ユーザモード
SH-4	ソフトウェア割り込み	290.1	1.21	ユーザモード
(240MHz)	関数呼び出し	180.6	0.75	スーパバイザモード

(1) i386 ディレクトリに格納されたファイルのコード量と sh4 ディレクトリに格納されたファイルのコード量を比較すると，sh4 ディレクトリの方がコード量が少ない．これは，Pentium4 環境が備える機能のうち，SH-4 環境が備えていないものがあるためである．例えば，SH-4 環境はディスク装置を備えていない．

(2) C 言語記述の部分では，新規作成したコード量は 3328 行であり，改造した既存ファイルのコード量は 301 行である．これらは，移植前のコード量である 15487 行の約 23%である．***AnT*** はマイクロカーネル構造の OS であるため，新規作成や改造の割合は大きくなる．なお，アセンブリ言語記述の部分は，新規作成である．

4.3 走行モード変更機構に関連する性能

Pentium4_***AnT*** の測定は，動作周波数 2.4GHz の Pentium4（Pentium4 2.40C）プロセッサを搭載した計算機で行った．また，SH-4_***AnT*** の測定は，動作周波数 240MHz の SH-4（SH7751R）プロセッサを搭載した SH-4 ボードで行った．

走行モード変更機構に関する評価として，システムコール発行に要する処理時間を比較する．具体的には，ソフトウェア割り込み，関数呼び出し，および sysenter 命令の 3 方式を比較する．なお，sysenter 命令方式の測定は，Pentium4_***AnT*** のみである．また，測定には，getpid（）を利用した．システムコール発行に要する処理時間を**表 5** と**図 13** に示す．また，システムコール発行のオーバヘッド削減率を**表 6** に示す．これらにより，以下のことがわかる．

(1) Pentium4_***AnT*** の場合，関数呼び出し方式は，ソフトウェア割り込み方式に比べ，オーバヘッドを約 0.34μs（約 64.7%）削減できる．しかし，sysenter 命令方式と比べると，オーバヘッドを約 0.01μs（約 3.9%）しか削減できない．したがって，走行モード変更機構を用いてプロセスをスーパバイザモードで走行させた場合，ソフトウェア割り込み方式ではオーバヘッドの削減効果は大きいものの，sysenter 命令方式では小さいといえる．

(2) SH-4_***AnT*** の場合，関数呼び出し方式は，ソフトウェア割り込み方式に比べ，オーバヘッドを約 0.46μs （約 37.7%）削減できる．しかし，Pentium4_***AnT*** の約 64.7%より

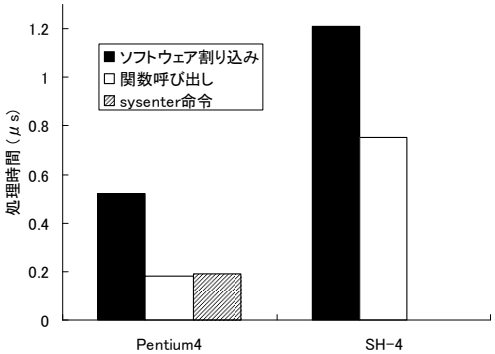


図 13 システムコール発行に要する処理時間

表 6 システムコール発行のオーバヘッド削減率

プロセッサ	比較項目	処理時間の差 (μs)	削減率（%）
Pentium4	（ソフトウェア割り込み） - （関数呼び出し）	0.34	64.7
	（ソフトウェア割り込み） - （sysenter 命令）	0.33	63.5
	（sysenter 命令） - （関数呼び出し）	0.01	3.9
SH-4	（ソフトウェア割り込み） - （関数呼び出し）	0.46	37.7

オーバヘッドの削減効果は小さい．したがって，走行モード変更機構を用いてプロセスをスーパバイザモードで走行させた場合，Pentium4_***AnT*** に比べ，SH-4_***AnT*** のオーバヘッド削減効果は小さい．

(3) Pentium4 プロセッサの動作周波数は，SH-4 プロセッサの 10 倍である．しかし，ソフトウェア割り込み方式の処理時間は約 1.7 倍，関数呼び出し方式の処理時間は約 4.1 倍の差である．したがって，Pentium4_***AnT*** と SH-4_***AnT*** のシステムコール発行の処理時間の差は，プロセッサの動作周波数の差と比べ小さい．これは，プロセッサのパイプライン段数や，キャッシュの影響であると推察できる．

5. おわりに

AnT が利用しているハードウェア機能を明確化し，起動処理，例外処理と割り込み処理，およびメモリ管理について，移植のために改造した内容を述べた．また，SH-4 プロセッサへ移植した ***AnT*** について，移植工数と走行モード変更機構に関連する性能を評価した．

C 言語記述の部分では，新規作成したコード量は 3328 行，改造した既存ファイルのコード

Vol.2011-ARC-195 No.3
Vol.2011-OS-117 No.3
2011/4/13

量は 301 行であり，移植前のコード量である 15487 行の約 23%を占める．また，Pentium4 プロセッサの動作周波数は，SH-4 プロセッサの 10 倍であるにもかかわらず，システムコール発行において，ソフトウェア割り込み方式の処理時間は約 1.7 倍，関数呼び出し方式の処理時間は約 4.1 倍の差しかないことを明らかにした．

残された課題として，SH-4 プロセッサへ移植した ***AnT*** におけるサーバプログラム間通信の性能評価がある．

参 考 文 献

1) J. Liedtke,“Toward Real Microkernels,”Commnuications of The ACM，Vol.39，Issue 9，pp.70-77，1996.
2) Andrew S. Tanenbaum, Jorrit N. Herder, Herbert Bos,“Can we make operating systems reliable and secure?,”IEEE Computer Magazine，Vol.39，No.5，pp.44-51，2006.
3) Black, D.L., Golub, D.B., Julin, D.P., Rashid, R.F., Draves, R.P., Dean, R.W., Forin, A., Barrera, J., Tokuda, H., Malan, G., and Bohman, D. ,“Microkernel Operating System Architecture and Mach,”Journal of Information Processing, Vol.14, No.4，pp.442-453，1992.
4) 谷口秀夫，乃村能成，田端利宏，安達俊光，野村裕佑，梅本昌典，仁科匡人,“適応性と堅牢性をあわせ持つ ***AnT*** オペレーティングシステム,”情報処理学会研究報告，2006-OS-103，Vol.2006，No.86，pp.71-78，2006.
5) 梅本昌典，田端利宏，乃村能成，谷口秀夫,“***AnT*** オペレーティングシステムにおけるメモリ領域管理の設計と実現,”情報処理学会研究報告，2007-OS-104，Vol.2007，No.40，pp.33-40，2007.
6) 岡本幸大，谷口秀夫,“***AnT*** オペレーティングシステムにおける高速なサーバプログラム間通信機構の実現と評価,”電子情報通信学会論文誌 (D)，Vol.J93-D，No.10，pp.1977-1989，2010.
7) IA-32 インテル R. アーキテクチャソフトウェア・デベロッパーズ・マニュアル，下巻 : システムプログラミング・ガイド，http://www.intel.co.jp/，Last accessed 25 Jan., 2007.
8) 日立 SuperH RISC engine SH-4 ハードウェアマニュアル SH7750.
9) 横山和俊，乃村能成，谷口秀夫，丸山勝巳,“応用プログラムの走行モード変更を可能にするプロセス制御機構,”電子情報通信学会論文誌 (D)，Vol.J91-D，No.3，pp.696-708，2008.