Vol.2010-ARC-189 No.9
Vol.2010-OS-114 No.9
2010/4/22

省電力MIPSプロセッサ Geyser の FPGA 版評価ボードへの Linux の移植

茂木　勇[†1]　木村一樹[†2]
砂田徹也[†3]　並木美太郎[†4]

システム LSI の省電力への要求により，細粒度パワーゲーティング (PG) を施した CPU コア「Geyser-0」の研究開発を行っている．この CPU コアの，評価環境として FPGA 上に Geyser-0 及び各種 IO，電力評価機構の実装を行ってきた．本研究では，その FPGA 環境へ，Linux システムの移植を行い評価環境を構築した．移植を行った Linux に対し，Geyser-0 の細粒度 PG 制御機能及び FPGA 環境の電力評価機構へアプリケーションプログラムからアクセスする機能を追加した．本研究で作成したシステムを利用し，システム起動中の ALU に対し電力が供給されていないスリープサイクルの割合を計測したところ，全実行時間の約 20%〜26%であるとの結果が得られた．

Porting Linux OS to an Evaluation Board for Power-saving MIPS Processor "Geyser"

Isamu Mogi,[†1] Kazuki Kimura,[†2] Tetsuya Sunata[†3] and Mitaro Namiki [†4]

This paper describes porting Linux OS to an evaluation board of Geyser-0 with FPGA. By the demand for power saving, CPU core named Geyser-0 with fine grain power gating was developed. Also, I/O and electric power evaluation mechanism of Geyser-0 was implimented on FPGA for test bed. The environment enables Linux kernel and applications to use functions to control fine grain power gating and electric power evaluation mechanisms. By using our system, the ratio of sleep cycle of ALU in total cycle measured 20% - 26%.

1. 背　　景

近年, システム LSI はその急激な高性能化に伴う消費電力の増大という問題に直面している. その主な要因として, 回路の高速化により増大するダイナミック電力と, 回路の微細化により増大するリーク電力があげられる．特に, 半導体の製造プロセスの微細化により回路と回路の距離が近くなることで, システム LSI の消費電力全体に対するリーク電力の割合が増大している．戦略的創造研究推進事業「CREST」における研究領域「情報システムの超低消費電力化を目指した技術革新と統合化技術」のプロジェクトである「革新的電源制御による次世代超低電力高性能システム」の研究チームでは, 回路技術, アーキテクチャ, システムソフトウェアの各階層が連携, 協力し, 革新的な電力制御を実現することで高性能システム LSI の消費電力を大きく低下させることを目指している．その成果として, 細粒度パワーゲーティングという技術を適用した MIPS R3000 ベースの CPU,Geyser-0 の開発が行われた．先行研究により Geyser-0 が搭載されたシステムへ OS の実装を行い, 複数のプログラムをマルチタスクで実行し評価を行ったところ, 演算器全体での消費電力の削減量は約 50 %となっている[2)]．

また, 先行研究により Geyser-0 on FPGA 環境が開発された[3)]．Geyser-0 の評価, 開発環境として,Geyser-0 を FPGA 上に実装した環境である．FPGA ボードに搭載された SRAM, タイマ, シリアルポート, プログラムデバック用機構などが利用できる．この環境により, 従来から利用されている RTL のシュミレーションに対し, 約 400 倍の速度でプログラムが実行でき, また FPGA ボードに搭載された各種出入力装置や, ステップ実行等のデバッグ機構が利用できるようになった.

2. Geyser-0 について

Geyser-0 は MIPS R3000 をベースとした CPU である[1)]．また細粒度パワーゲーティン

†1 東京農工大学工学部情報コミュニケーション工学科
Department of Computer and Information Science, Tokyo University of Agriculture and Technology
†2 東京農工大学大学院工学府
Graduate School of Technology, Tokyo University of Agriculture and Technology
†3 東京農工大学大学院工学府
Graduate School of Technology, Tokyo University of Agriculture and Technology
†4 東京農工大学大学院共生科学技術研究院
Institute of Symbiotic Science and Technology,Tokyo University of Agriculture and Technology

Vol.2010-ARC-189 No.9
Vol.2010-OS-114 No.9
2010/4/22

グ技術を用い,CPU の省電力化を行う. このプロセッサは,ALU, シフタ, 乗算器, 除算器, システム制御コプロセッサ CP0 それぞれの回路に対する電源の供給を選択的に遮断することにより CPU の省電力化を行う技術である. プロセッサがメモリからの命令をフェッチ, 及び命令のデコードを行っている間に, 利用する回路に電源を供給を開始, 回路をアクティブ状態に遷移させ, 命令実行終了後に回路の電源を遮断, スリープ状態にする. 回路がスリープ状態になればその回路が消費する電力は非常に低く抑えることができるが, 回路のアクティブ/スリープ切り替えの瞬間, アクティブ時よりも多く電力を消費してしまう. そのため, アクティブ/スリープ切り替えが頻繁に行われる場合, 逆に消費電力が増加してしまうことが考えられる. この問題をソフトウェアを用いて回避するため,Geyser-0 にはパワーゲーティング制御命令とパワーゲーティング制御レジスタが追加された.

2.1 パワーゲーティング制御命令

MIPS R3000 の R 形式命令の上位 6 ビットを $(10111)_2$ とすることで, その命令実行後に利用した演算器は, 演算終了後の自動スリープをさせないよう設定できる[2]. 事前に, 現在実行中の命令で利用中のパワーゲーティング対象の演算器が近い未来に再使用される可能性が高い場合, 自動スリープをせずに再使用することによりスリープ状態遷移及びウェイクアップ状態遷移のオーバーヘッドを削減することができる.

2.2 パワーゲーティング制御レジスタ

CP0 に PGStatus というパワーゲーティング制御レジスタを追加した. PGStatus レジスタを利用して, 各パワーゲーティング対象の回路がどのような場合にスリープを行うかを制御することができる. 具体的には,ALU, シフタ, 除算器, 乗算器において

- 常にスリープ
- 常にアクティブ, ただしキャッシュミスが発生した場合はスリープする
- 常にアクティブ

の三種類の動作を選択することができる.

また,CP0 においては

- 常にアクティブ
- カーネルモードでは常にアクティブ, ユーザーモード時は常にスリープ

の動作を選択させることができる. この機能を用いれば, 実行中のプログラムのパワーゲーティング対象回路を使う頻度等から, パワーゲーティング制御レジスタを最適な値に設定することにより, 状態遷移のオーバーヘッドを削減することができる.

2.3 Geyser-0 on FPGA 環境

本研究では Geyser-0 の動作環境として, 東京農工大学並木研究室の木村により実装された FPGA ボード上に構築された Geyser-0 の評価環境, Geyser-0 on FPGA 環境[3] を用いる. 本環境では FPGA 上に Geyser-0 コアを再現することにより, RTL のシュミレーションよりも高速かつ実機に近い動作を実現でき, また FPGA ボードに搭載された SRAM, タイマ, シリアルポート, 電力評価機構, およびアドレスへのブレークポイントの設置やクロック単位のステップ実行等のデバック機構を利用できる.

2.4 シリアルポート

FPGA ボードに搭載されているシリアルポート経由で外部と通信を行うことができる. 本環境では, 出入力用の, メモリアドレス 0xA0000000 - 0xA0000008 に対応するメモリマップドレジスタにデータを書き込むことでシリアル出力が行われ, シリアル入力されたデータはメモリマップドレジスタを読み込むことで取得できる.

2.5 電力評価機構

FPGA では Geyser-0 の実際の消費電力を測定することができないため, スリープ制御信号の動作を記録し, これを換算することで省電力効果を算定する. 具体的には ALU, シフタ, 乗算器, 除算器各々において,

- スリープサイクル数
- スリープ状態遷移回数
- ウェイクアップ状態遷移回数

を集計することができる. ソフトウェア側でそれらの情報にアクセスするためには, Geyser-0 on FPGA 環境上に搭載されている, メモリアドレス 0xA3000000 - 0xA30007F0 に対応するメモリマップドレジスタを利用する.

3. 目　　標

本研究では, Geyser-0 on FPGA 環境へ MIPS R3000 版 Linux の移植を行い, また Linux に, アプリケーションプログラムから Geyser-0 on FPGA の細粒度パワーゲーティング制御機能及び評価機構にアクセスが可能になるよう, システムコールを経由したへのインターフェースを追加する. これにより, 既存の Linux アプリケーションプログラムに対し性能・スリープサイクル数の評価環境を整えることを目標とする.

移植を行う Linux においては, Geyser-0 on FPGA 上に搭載されているハードウェアも十分動作可能な機能について, 現実的な範囲で, マルチプロセスや仮想記憶機能を含めたで

Vol.2010-ARC-189 No.9
Vol.2010-OS-114 No.9
2010/4/22

きるだけ多くの機能の移植を行い, 評価を行う. 現時点において,Geyser-0 on FPGA 環境には CPU コア, 評価環境以外に利用可能なハードウェア資源は現在のところ外部と通信を行うシリアルポートと,16MB の RAM のみとなっている.

また, アプリケーションから Geyser-0 の評価機構にアクセスができるように, 評価機構へのアクセスを行うシステムコールを Linux への追加を行う. 評価機構を用いることにより, 各種アプリケーション実行中に, そのアプリケーションが利用している各演算器のスリープサイクル及び全実行サイクル数を取得することがき, そのことにより, またそれらを用いてどのスリープポリシーが最適であるかの検討を行うことができるようにする.

4. Linux システムの設計

4.1 全体の設計

Geyser-0 on FPGA 環境上に Linux システムを構築する. 本研究では Xilinx 社の FPGA ボード, ML501 上に構成された開発・評価用 Geyser-0 on FPGA 環境へ, OS として Linux, 出入力装置としてシリアルポート, ファイルシステムとして tmpfs, C ランタイムライブラリとして uClibc の移植を行う.

表 1 本システムの概要

FPGA ボード	Xilinx ML501
動作クロック	16.5MHz
RAM	16MB
カーネル	Linux 2.6.30.9
出入力装置	シリアルポート
ファイルシステム	tmpfs
C ランタイムライブラリ	uClibc 0.9.30.1

本システムの構成は図 1 のようになる. 本研究で追加・修正を行った個所を灰色で示す.

Linux

Linux の基本的な機能である,TLB を用いた仮想記憶機能, メモリの管理機能, プロセスやスレッドの管理機能の移植を行った. また,Geyser-0 on FPGA 環境における電力評価機構, パワーゲーティング制御レジスタを利用するためのシステムコールを追加した.

デバイスドライバ

本システムには, 外部と通信手段としてシリアルポートのみが利用できる. それらに対応

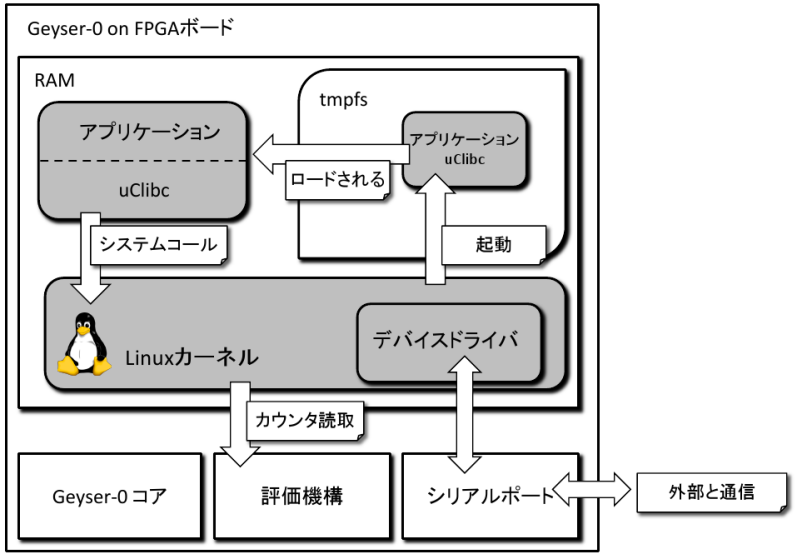

図 1 本システムの構成

するデバイスドライバを作成し, デバイスファイルを経由してアクセスできるようにした.

tmpfs

本システムには二次記憶装置が存在しない. しかし,Linux においては, ファイルシステムは通常ファイルを保存する以外にもアプリケーションからハードウェアにアクセスする機能等, いくつかの重要な機能を担当している. そのため, RAM 上にファイルシステムを展開, 操作を行う tmpfs を利用する.

uClibc

本システムでは利用可能な RAM が 16MB と少なく,Geyser-0 での一部命令の変更を受けて,C ランタイムライブラリのソースコードの修正も必要となるため,C ランタイムライブラリとして, 規模の小さい uClibc を用いた.

アプリケーション

評価用アプリケーション等,uClibc をリンクしたアプリケーションを tmpfs 上に保存して

Vol.2010-ARC-189 No.9
Vol.2010-OS-114 No.9
2010/4/22

おき, 実行することができるようにした.

システムの起動

Geyser-0 on FPGA 環境を起動する際は, ホストマシンと通信し,Linux カーネル, ファイルシステムをホストマシンから FPGA へ転送する. 転送が終了したら, 指定したロードアドレスからカーネルの実行が開始される. カーネルが起動したら, メモリ管理機能, プロセス管理機能等初期化後にカーネルは RAM 上に tmpfs ファイルシステムを展開し, そこに含まれるプログラムを起動したのちにアイドル状態となり, 外部からの割り込みを待つ. 起動されるプログラムは tmpfs 上に存在する評価用プログラムである.

4.2 評価機構へのインターフェース設計

表 2 新規追加したシステムコール

システムコールの機能	システムコール名
演算器のスリープポリシー設定	geyser_set_policy()
スリープサイクルカウンタ有効化	geyser_enable_counter()
スリープサイクルカウンタ無効化	geyser_disable_counter()
スリープサイクル数取得	geyser_get_count()
スリープサイクルカウンタ初期化	geyser_clear_counter()

Linux の各機能を移植するとともに、Geyser-0 on FPGA のもつパワーゲーティング制御機能, 電力評価機構へのインターフェースを提供する. FPGA 環境と Geyser-0 の実機では, クロック周波数等が異なるため実機と FPGA 環境ではプログラムの性能評価時の実行時間が異なる. そのため, FPGA 環境の評価機構を利用してプログラムの実行サイクル数を記録することにより, 評価機構で実行したプログラムの実機での実行時間を算出できるようにし,Linux のシステムコールとしてアプリケーションから利用できるようにする. また, 各演算器に電力を供給していない状態であるスリープサイクル数も取得できるようにする. また, プログラム上で演算器のアクティブ/スリープ切り替え時には通常時よりも消費電力が上昇してしまうためできるだけ長い時間スリープ状態を保つことが重要である. そのためスリープサイクル数の取得の際は, 「N 回連続でスリープ発生」ごとに, 発生した回数を記録することとする.

5. Linux システムの移植

Linux の実装について述べる. 実装作業は, Geyser-0 アーキテクチャへの移植作業, Geyser-0 on FPGA 環境への移植作業にわかれる. 移植によるソースコードの修正行数は, Geyser-0 アーキテクチャ移植のための MIPS 依存コードの変更が 250 行程度, Geyser-0 on FPGA 環境への対応のために追加したコードが 300 行程度, その他若干行となった.

表 3 修正を行ったソースコードののべ修正・追加行数

モジュール	修正追加行数	対象全行数
MIPS 依存, TLB 関連	50	7,000
MIPS 依存, C ランタイム	20	4,500
MIPS 依存, その他	180	12,5000
シリアルポート用ドライバ	60	-
システムコール追加	90	-
その他 FPGA ボード専用	140	-
その他	30	-

5.1 MIPS R3000 と Geyser-0 の相違点への対応

Linux ソースコードツリーには,Geyser-0 アーキテクチャのベースとなっている MIPS R3000 用のソースコードが存在しているため, 今回の移植作業ではそのコードを利用することになる. そこで, R3000 アーキテクチャとの互換性の無い変更へ対応する修正を行う. これら変更に対応するため次に述べるような仕様の差異の確認とソースコードの修正を行った.

CP0 レジスタ番号の差異への対応

Geyser-0 アーキテクチャは, 各コプロセッサレジスタについて MIPS R3000 アーキテクチャからコプロセッサレジスタ番号が変更されているため, コプロセッサレジスタの番号を MIPS R3000 のものから Geyser-0 のものへ変更を行った.

プロセッサ ID レジスタの省略への対応

MIPS R3000 アーキテクチャのプロセッサ ID レジスタには CPU の種類, そのリビジョンが保存されているが, Geyser-0 アーキテクチャには存在しない.Linux ソースコードではプロセッサ ID レジスタを利用しているコードは 2 個所あり, 片方はデバッグ用途、もう片方はプロセッサ ID レジスタの値により浮動小数点レジスタの有無などの情報を実行時に確認を行う個所であった. そのため, 該当箇所を,Geyser-0 on FPGA 環境に適合するように変更行った.

コンテキストレジスタの省略への対応

Geyser-0 アーキテクチャには, MIPS R3000 アーキテクチャのコンテキストレジスタが存在しない. コンテキストレジスタは TLB のリフィル処理の高速化を目的としたレジスタ

Vol.2010-ARC-189 No.9
Vol.2010-OS-114 No.9
2010/4/22

である. Linux ソースコードでは, コンテキストレジスタを利用しているコードは 2 個所あり, 2 か所とも TLB 関連の例外発生時に, 例外を発生させる原因となった仮想アドレスの情報をコンテキストレジスタより読み出す処理が行われていた. TLB 関連の例外発生時には, 誤り仮想アドレスレジスタにコンテキストレジスタと同じ情報が格納されるため, 誤り仮想アドレスレジスタを利用するコードへ修正した.

EntryHI レジスタの一部仕様変更への対応

EntryHI レジスタは,64 ビットの TLB エントリの上位 32 ビットの内容に対応し, 下位 32 ビットに対応する EntryLO レジスタと併用して TLB のエントリの読み書きに利用される. MIPS R3000 アーキテクチャにおいては, EntryHI レジスタの仮想ページ番号ビットには, TLB 関連での例外発生時に原因となった仮想ページ番号が格納されることとなっているが, Geyser-0 アーキテクチャではそのような処理は行われない. そのため, コンテキストレジスタの代用と同じように, 誤り仮想アドレスレジスタの内容を加工し EntryHI レジスタへ書き込むことで対応した.

ステータスレジスタの一部仕様変更への対応

ステータスレジスタ CPU の様々な状態を表すビットが含まれている.Geyser-0 では, その中の各コプロセッサが利用可能かどうかを設定する Cu0, Cu1, Cu2, Cu3 ビットが省略されていて値を書き込むことができない. Linux においては, Cu0 ビットのみ使われている. そのため, Cu0 ビット読み書き時に, 読み書きされるデータの場所を特定のメモリアドレスへ変更することで対応した.

syscall, break 命令の一部仕様変更への対応

Geyser-0 の内部処理の関係により, syscall 命令, break 命令の前後で nop 命令を実行しなければならない. そのため, 次のような修正が必要となる.

- syscall 命令の直前, 直後に nop 命令を挿入する
- ジャンプ, 分岐命令の飛び先に syscall 命令がある場合, 飛び先を直前の nop 命令に変更する

前者はその通りに修正を行った. 後者は通常のジャンプ, 分岐命令の飛び先の修正の他に, 例外から復帰した先に syscall,break 命令が存在する場合の対応を行わなければならない. ハードウェアによる例外や TLB 関連の例外は割り込みマスクが行われていない状態ではどこでも起こりうるため, 例外ハンドラ内で, 例外発生時の復帰アドレスの内容を調査し, 内容が syscall,break 命令であった場合は例外の復帰アドレスを, その直前に存在する nop 命令を指すように変更を行った.

tlbwr 命令への対応

Geyser-0 の内部処理の関係により, MIPS R3000 の tlbwr 命令が利用できない. 移植の際はその他の TLB 書き込み命令とランダムレジスタを組み合わせて tlbwr 命令と同様の動作を実現し, 実装を行った.

TLB 数変更への対応

MIPS R3000 アーキテクチャには 64 個の TLB が存在する.Geyser-0 の TLB の個数は R3000 アーキテクチャと違い 16 個であるが, MIPS 版 Linux には利用する TLB 数を起動時に決定することが可能であるため, R3000 用の TLB 操作のコードをそのまま利用することができた.

5.2 Geyser on FPGA 環境への移植

Geyser on FPGA 環境には, 出入力装置としてシリアルポート, 主記憶として 16MB の SRAM, タイマ, 電力評価のためのパフォーマンスカウンタが存在する. これらの機能を有するボード依存コードを Linux ソースコードに追加した.

主記憶

MIPS R3000 版 Linux カーネルは, 有効な RAM が存在するアドレスを通知することで,TLB によるマップや,MIPS R3000 の固定マップ領域で利用する対象の物理アドレスを指定することができる. 本環境では物理アドレス 0x00000000 - 0x01000000 までに有効な RAM が存在すため, そのことをカーネルに通知するコードを追加した. 移植を行った Linux は, 通知した物理アドレス対して TLB を用い, 仮想記憶機能を実現する.

tmpfs

カーネルのコンフィグレーションにより, 二次記憶装置の代わりとして利用するため, RAM 上のファイルシステムである tmpfs をデフォルトで有効な状態にした. これにより, カーネルは初期化時にルートファイルシステムとして tmpfs を自動的にマウントするようになる. 評価用のプログラムの他に/dev, /tmp ディレクトリを作成, 便宜利用するようにした.

シリアルポート

シリアルポートを利用するために Linux のコンソールデバイスドライバを作成, 組み込みを行った. アプリケーションプログラムから read() や write() システムコールが発生すると, コンソールデバイスドライバは,Geyser-0 on FPGA に存在する, シリアルポートにアクセスするためのメモリマップドレジスタ経由でデータの送受信を行い結果をアプリケーションプログラムへ返す.

Vol.2010-ARC-189 No.9
Vol.2010-OS-114 No.9
2010/4/22

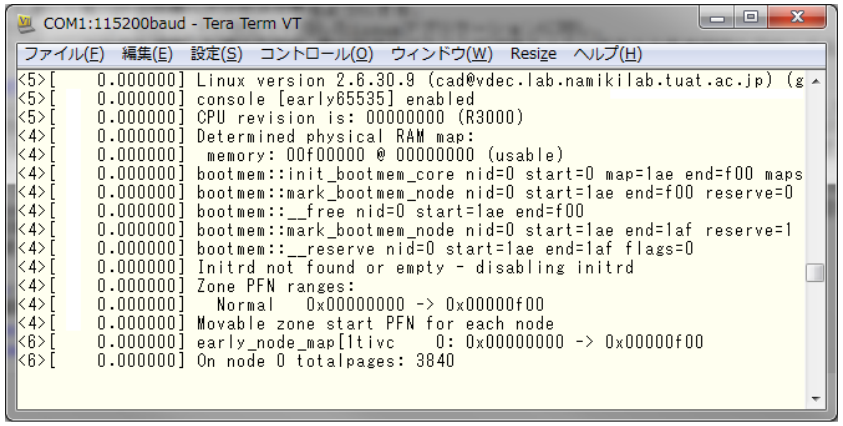

図 2 Geyser-0 on FPGA 環境上で動作する Linux

評価機構へのプログラミングインターフェース

Geyser on FPGA に搭載されている評価機構をユーザープログラムから利用するためのシステムコールを新たに追加した.

6. 評　　価

評価機構へのプログラミングインターフェースを利用し各 Linux システムコールやその他機能, アプリケーションについて ALU のスリープサイクル及び実行時間の評価を行う. 各計測結果の表の sleep cycle の行はそれぞれ, 1 サイクル連続スリープサイクルが起きた回数, 2 サイクル連続のスリープサイクルが起きた回数, ... ,40 サイクル連続のスリープサイクルが起きた回数を表す. また, 大きい数値の行は, 小さい数値の行のスリープサイクル数を含まない. つまり, 16 サイクル連続のスリープサイクルが起きた回数は, それに含まれる 12 サイクル連続でのスリープサイクルは, 12 サイクル連続での sleep cycle の行には含まない. また, 実行時間は total cycle 数と FPGA の動作クロック数 16.5MHz から算出している.FPGA のクロックサイクル数は FPGA に搭載されている液晶を利用して見ることができる. 目測においても,FPGA の動作クロック数が概ね 16.5MHz であることも確認できた.

(1) Linux のブートプロセス

Geyser-0 Linux カーネルのブートプロセスを計測する. ソースコードの FPGA にロードされてから初めて実行される個所から計測を開始し, ユーザープログラムを実行する直前で計測を終了した. 結果を表 4 に示す. 今回評価を行ったプログラムの中で,sleep cycle rate が一番高いという結果となった. これは, シリアルポートの出力待ちの時間が影響していると考えられる.

表 4 Linux のブートプロセスの計測結果

sleep cycle	times	sleep cycle	times
1	7,471	12	2,293,159
2	394,755	16	2,315,963
6	2,043,368	40	3,290
11	48		
total sleep cycle	77,759,385	total cycle	296,690,259
sleep cycle rate	0.262	exec time	17.981sec

(2) プロセス生成

fork() システムコールを用いたプロセス生成処理の計測を行った. 結果を表 5 へ示す.

表 5 プロセス生成の計測結果

sleep cycle	times	sleep cycle	times
1	174	12	35,108
2	7,118	16	108,085
6	62,119	40	7
11	13		
total sleep cycle	2,538,203	total cycle	11,121,709
sleep cycle rate	0.228	exec time	0.674sec

(3) コンテキストスイッチ

sched_yield() システムコールを実行しコンテキストスイッチを行い, それによって親プロセスに切り替わった場合を特定し計測を終了, 集計処理を行った. 結果を表 6 へ示す. MIPS R3000 用 Linux では,fork() システムコールにより作成された子プロセスの実体は fork() を呼び出したプロセスとなっている. そのため,fork() で作成された子プロセスは親プロセス用のプロセス空間をそのまま利用する. そのかわり, 親プロセスにコンテキストスイッチした際, 親プロセスは自身で利用するプロセス空間を新たに設定し直さなければならない.fork() システムコールよりも実行時間が若干長いのはそのためであると考えられる.

Vol.2010-ARC-189 No.9
Vol.2010-OS-114 No.9
2010/4/22

表 6　コンテキストスイッチの計測結果

sleep cycle	times	sleep cycle	times
1	177	12	35,891
2	8056	16	108,566
6	62,714	40	7
11	13		
total sleep cycle	2,560,744	total cycle	11,42,394
sleep cycle rate	0.224	exec time	0.692sec

(4) ページ確保

Linux カーネルのコード内にてわざとページフォールトを起こし, その実行結果について計測を行った. 今回は実行バイナリファイルをユーザー空間に書き込む処理内に計測, 集計コードを追加した. また, 計測結果を表 7 へ示す. ページフォールト処理において, スリープサイクルの割合が非常に高いのは, セキュリティのため取得した新しいページの値をクリアする処理による影響だと考えられる.

表 7　ページ確保の計測結果

sleep cycle	times	sleep cycle	times
1	4	12	1,248
2	1,944	16	913
6	1,866		
11	13		
total sleep cycle	44,815	total cycle	176,713
sleep cycle rate	0.254	exec time	0.011 sec

(5) ファイルへの書き込み

RAM 上に展開されたファイルシステムに対しデータを書き込む処理を行った. 結果を表 8 へ示す. データの書き込みが主な処理となるため, 他のシステムコールに比べてスリープサイクル数の割合が若干高い結果となった.

(6) ユーザープロセスとして実行する行列演算

行列の足し算, 掛け算を行うプログラムを利用して計測を行った. 結果を表 9 へ示す. ほぼすべての時間を計算に費やしているため, スリープサイクル数の割合が他の項目に対して低いことがわかる.

表 8　ファイルへの書き込みの計測結果

sleep cycle	times	sleep cycle	times
1	37	12	452,810
2	152,310	16	554,491
6	406,580	40	55
11	13		
total sleep cycle	17,052,056	total cycle	73,740,264
sleep cycle rate	0.231	exec time	4.469 sec

表 9　ユーザープロセスとして実行する行列演算の計測結果

sleep cycle	times	sleep cycle	times
1	24,096	12	113,563
2	9,404	16	173,452
6	238,269	40	114
11	12		
total sleep cycle	5,615,198	total cycle	26,349,947
sleep cycle rate	0.213	exec time	1.597 sec

(7) カーネルモードで実行する行列演算

カーネルモードで実行する行列演算プログラムでは, ユーザーモードで実行する行列演算プログラムと同じものを Linux のブートが完了し, 割り込み待ちの無限ループを実行する直前から計測を開始した. プログラムの終了とともに計測を終了する. 意図した結果であるが, カーネルモードで実行する行列演算は, アドレス変換やページフォールト等のオーバーヘッドが発生しないためユーザープロセスとして実行する行列演算と比べてもスリープサイクル数の割合がさらに低く, 今回計測を行った項目の中で一番低いものとなった.

7. 考　　察

本研究にて,Linux の主要な機能を移植することができた. 評価の節で示したように, アプリケーションプログラムからはプロセス管理機能, ファイルシステム機能などのシステムコールを利用可能になった. それにより, 研究評価環境の利用は GCC, GNU Build System などを利用した一般的な Linux 向けの開発作業とほぼ同じ手順となり, RTL シミュレーションによる既存の環境と比較すると, プログラム実行での評価に対する障壁が下がったと考えられる. また, 既存の環境では評価のシミュレーション時間も長く, ターンアラウンドタイム

Vol.2010-ARC-189 No.9
Vol.2010-OS-114 No.9
2010/4/22

表 10 カーネルモードで実行する行列演算の計測結果

sleep cycle	times	sleep cycle	times
1	24,009	12	69,877
2	2,710	16	112,479
6	206,914	40	49
11	12		
total sleep cycle	3911193	total cycle	19,902,414
sleep cycle rate	0.197	exec time	1.206 sec

が長かったが本環境により研究, 評価の結果が早く得られるようになった.

評価結果については, 実行時間の多くにシリアルポートの I/O 待ち時間が占めるカーネルの起動処理や, ページのクリア処理を行うメモリ管理機能におけるページの確保処理など,ALU の利用頻度が低いと考えられる処理はスリープサイクル数の割合が大きくなる. また, 行列演算処理など ALU の利用頻度が高いと考えられる処理はスリープサイクル数の割合も低くなることがわかった. それぞれのスリープサイクルの割合は, 約 26% - 約 20%となることがわかった.

8. ま と め

本研究により,Geyser-0 上での Linux システムの動作を確認できた. また, 本システム上で, マルチプロセス機能, 仮想記憶機能, メモリ確保機能など, 研究用の簡易 OS では実現できなかった, 多くのシステム LSI 上のシステムで実際に利用されるプログラムコードに対する, 細粒度パワーゲーティング技術の有効性についての評価を行うためのデータが得られるようになった. 今後の課題としては, 本システムでは Geyser-0 の省電力化機能を実際に利用し, システムの省電力化を行ってはいないため, Geyser-0 のパワーゲーティング制御機能や評価結果等を用いて省電力化を行う機能を追加することが課題となる. また,Geyser-0 の最新バージョンが利用可能な, Geyser-2 on FPGA 環境への移植作業がある.Geyser-2 では syscall,break 命令が通常の R3000 アーキテクチャと同じように利用できる他, 各種の改善が行われている. また, 現在移植した Linux カーネルは, 安定性のためにカーネルイメージの作成時にほとんどのカーネルの機能を OFF にしている. そのため,FIFO や仮想端末, ループバックデバイスなど, 多くの機能が利用不可能である. また, アプリケーション用のライブラリとして用意したのは C ランタイムライブラリのみのため, 多くのアプリケーションの移植に支障が生じてしまう. そのため, 今後は現在 OFF にしているカーネルの機能をとりこみ, またよく利用さているライブラリの移植を行い, より多くの既存のアプリケーションに対し評価が行えるような環境を整えることが課題となる.

参 考 文 献

1) 中村宏, 天野英晴, 宇佐美公良, 並木美太郎, 今井雅, 近藤正章 革新的電源制御による超低電力高性能システム LSI の構想. 情処研報 ARC/信学報 ICD, pp.79-84 (2007)
2) 砂田徹也, 関直臣, 香嶋俊裕, 中田光貴, 近藤正章, 天野英晴, 並木美太郎 省電力 MIPS プロセッサにおける OS の試作とシミュレーションによる電力評価. 情報処理学会「システムソフトウェアとオペレーティング・システム」第 108 回研究報告, Vol.2008-OS-108 pp.163-170 (2008)
3) 木村一樹, 砂田徹也, 長井智英, 関直臣, 近藤正章, 天野英晴, 宇佐美公良, 中村宏, 並木美太郎 省電力 MIPS プロセッサコア評価のための計算機システムの FPGA による試作. 情報処理学会「システムソフトウェアとオペレーティング・システム」第 111 回研究報告, Vol.2009-OS-111, No.34, pp.1-8 (2009)
4) 光澤敦, 林和則, 田中浩一. Mach マイクロカーネルシステムの MIPS アーキテクチャへの移植. 情報処理学会「システムソフトウェアとオペレーティング・システム」第 75 回研究報告,OS-75, No.56, pp.19-24 (1997)
5) 森田知宏, 八木孝介, 湯川真紀. 組込み Linux 用 C ライブラリ : MIPS ボードによる uClibc と Glibc の比較. 電子情報通信学会「磁気記録研究会」技術研究報告. pp.43-48 (2007)
6) Dominic Sweetman 著. See MIPS Run Linux Second Edition. Morgan Kaufmann Publishers Inc (2006)
7) Gerry Kane 著, 前川 守監訳. mips RISC アーキテクチャ-R2000/R3000-, 共立出版,(1992 Oct).
8) Linux/MIPS, http://www.linux-mips.com/wiki/Main_Page
9) Ralf Bachle, Linux/MIPS HOWTO, http://oss.sgi.com/mips/mips-howto.html
10) Omicron Linux/MIPS, http://tiki.is.os-omicron.org/tiki.cgi?p=Linux%2FMIPS
11) 高橋浩和, 小田逸郎, 山幡為佐久 著. Linux カーネル 2.6 解読室. ソフトバンク クリエイティブ株式会社 (2006)
12) Daniel p. Bovet, Marco Cesati 著. 詳解 LINUX カーネル 第 3 版. 株式会社オライリー・ジャパン (2007)
13) Jonathan Corbet, Alessandro Rubini, Greg Kroah-Hartman 著. Linux デバイスドライバ 第 3 版. 株式会社オライリー・ジャパン (2005)
14) Michael Barr, Anthony Massa 著. C と GNU 開発ツールによる組み込みシステムプログラミング 第 2 版. 株式会社オライリー・ジャパン (2007)