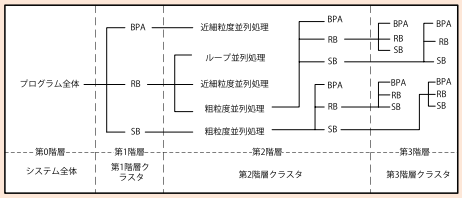
OSCAR自動並列化コンパイラとOSCAR API
マルチコアの自動高速化と電力削減

早稲田大学　笠原博徳・木村啓二研究室

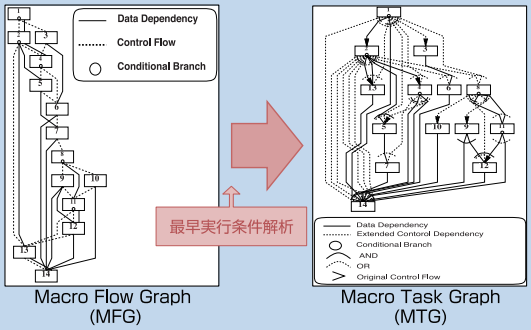
研究室ホームページ
http://www.kasahara.cs.waseda.ac.jp

マルチグレイン並列処理
異なる粒度の並列性を階層的に利用



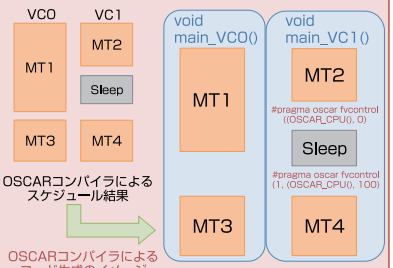
- 粗粒度タスク並列処理
  - サブルーチン
  - ループ
  - 基本ブロック
- 中粒度並列処理
  - ループイタレーション
- 近細粒度並列処理
  - ステートメント
  - インストラクション

粗粒度タスク間並列性解析
プログラム中のループ・関数間粗粒度タスク間の並列性を解析

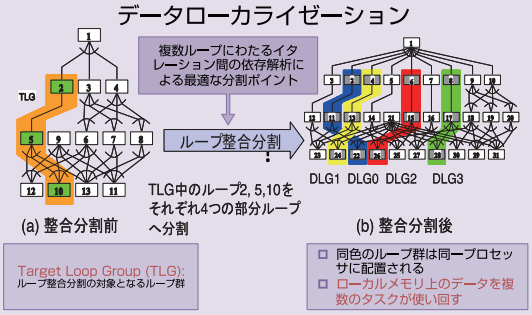


連絡先メールアドレス
ET2013@kasahara.cs.waseda.ac.jp

低電力並列APIプログラム生成



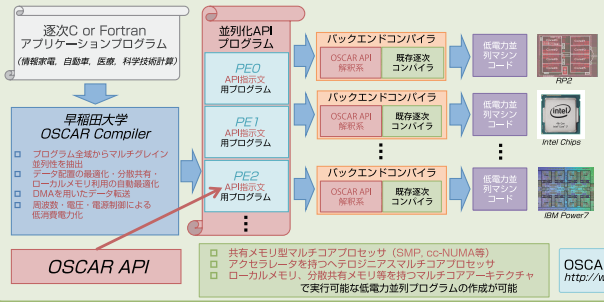
ローカルメモリ・キャッシュ最適化のための データローカライゼーション



Target Loop Group (TLG):
ループ整合分割の対象となるループ群

- 同色のループ群は同一プロセッサに配置される
- ローカルメモリ上のデータを複数のタスクが使い回す

マルチプラットフォームソフトウェアインタフェース：OSCAR API
各種OS・マルチコアプロセッサ上の自動並列化が可能



OSCAR APIの指示文
- Parallel Execution API
- Memory Mapping API
- Synchronization API
- Data Transfer API
- Power Control API
- Timer API
- Accelerator API
- Cache Control API

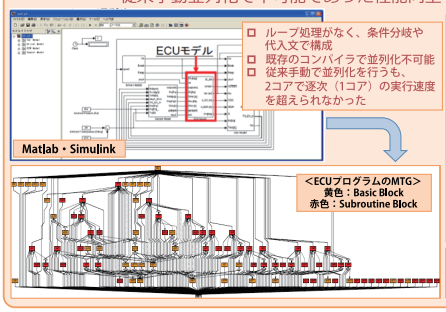
- 共有メモリ型マルチコアプロセッサ（SMP, cc-NUMA等）
- アクセラレータを持つヘテロジニアスマルチコアプロセッサ
- ローカルメモリ、分散共有メモリ等を持つマルチコアアーキテクチャ

で実行可能な低電力並列プログラムの作成が可能

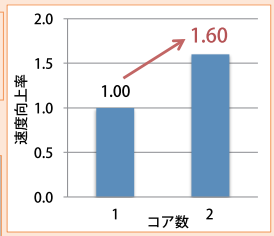
OSCAR API v2.0
http://www.kasahara.cs.waseda.ac.jp/api2/regist.html

OSCAR自動並列化コンパイラによる 自動車・医療・マルチメディアアプリケーションの速度向上と電力削減

MATLAB・Simulinkを用いてモデルベース設計した ECU（エンジン制御ユニット）プログラムのRP-X上での並列処理
従来手動並列化で不可能であった性能向上を自動並列化で実現



- ループ処理がなく、条件分岐や代入文で構成
- 既存のコンパイラで並列化不可能
- 従来手動で並列化を行うも、2コアで逐次（1コア）の実行速度を超えられなかった



2コアで1.6倍の高速化

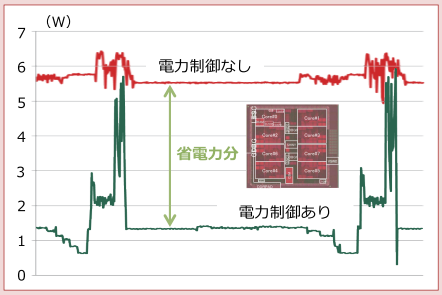
OSCARコンパイラ協調型アーキテクチャ ホモジニアスマルチコアRP2上でのMPEG2デコーダの電力削減
リアルタイムMPEG2デコードを8コアRP2上で消費電力1/4に削減

<RP2>
CPU：SH4A

<OSCARコンパイラの電力制御>
- 消費電力をHWとSW協調で低減
- 周波数/電圧・電源遮断制御

8コア電力制御なし 平均電力
5.73[W]

8コア電力制御あり 平均電力
1.52[W]

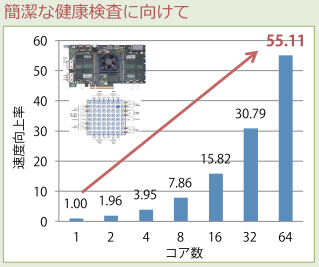


次世代カメラ・カプセル内視鏡のための静止画圧縮 JPEG XRエンコーダのメニーコアチップ上での並列処理
短時間・簡潔な健康検査に向けて

<未来のカプセル内視鏡>
人間ドックで朝飲むだけで，カプセル内視鏡が体内の画像を複数のカメラで連続的に撮影，病変認識，画像送付，記憶する．誰でも検査が簡単に．



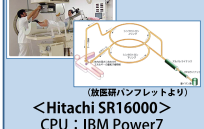
<TilePro64>
CPU：Tile Processor



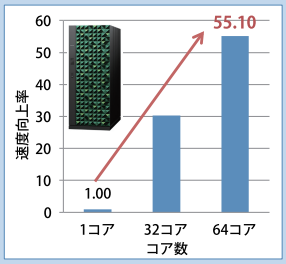
64コアで55倍の高速化：10[s] → 0.18[s]

重粒子線ガン治療のメニーコアサーバ上での並列処理
ガン治療の低治療費化・保険適応へ

<重粒子線ガン治療>
重粒子線（炭素イオン）を極めて正確に制御・照射し，ガン細胞のみを消滅させる治療法．開腹手術不要・痛みなく治療が可能．

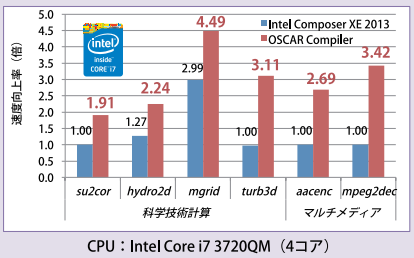


<Hitachi SR16000>
CPU：IBM Power7



64コアで55倍の高速化：20分 → 22秒

Intel Corei7 4コアマルチコアプロセッサ 搭載サーバ上での並列処理



CPU：Intel Core i7 3720QM（4コア）

インテル製コンパイラを全体で平均2.2倍高速化

OSCARコンパイラによるAndroidマルチコア上での自動低消費電力化 - 消費電力を1/7に削減 -

- OSCAR Compiler
- スマートフォン
- 低消費電力化

デモンストレーションビデオ:http://youtu.be/M63W2RAjXfc

Android上での低消費電力化の実現

端末充電の１回以下/週 を目指して

電力制御の手法

① OSCARコンパイラによる自動並列化及び電力制御

電力制御環境の作成

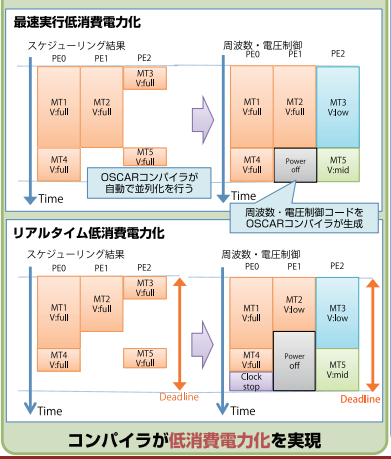
② WFIによる高速クロックゲーティング

電力測定環境の構築

③ ODROID-X2上での電力測定環境の構築

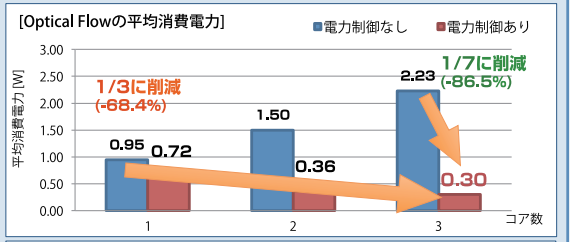
並列化とDVFS・高速クロックゲーティングによる低消費電力化の実現

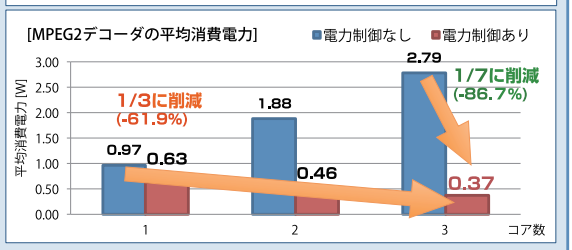
① OSCARコンパイラによるアプリケーション自動並列化および電力制御



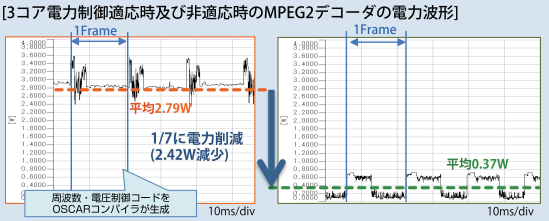
コンパイラが低消費電力化を実現

ARM Cortex-A9 4コアによるOSCARコンパイラによる電力削減効果



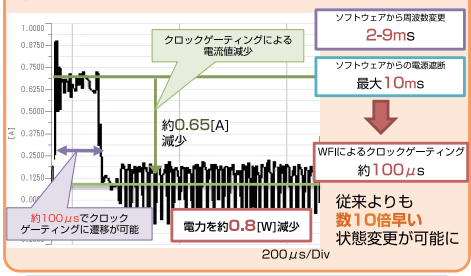


[3コア電力制御適応時及び非適応時のMPEG2デコーダの電力波形]

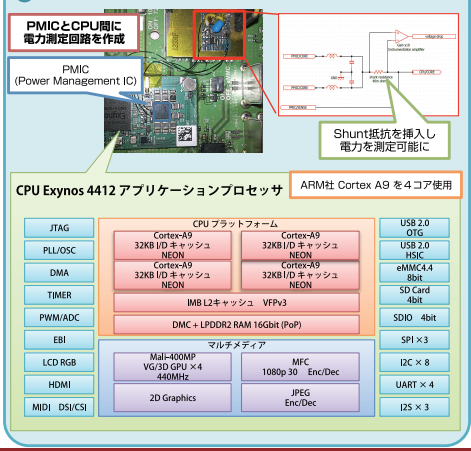


(a)電力制御なし 平均2.79W　(b)電力制御あり 平均0.37W

② WFIによる100μsの高速クロックゲーティング手法



③ ODROID-X2を用いた電力測定環境の構築



Embedded Technology 2013

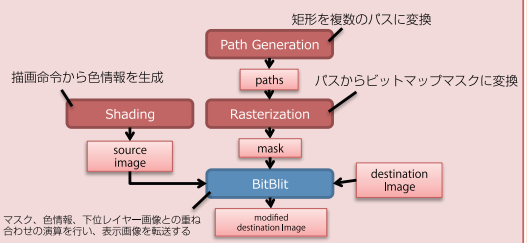
OSCARコンパイラを用いた Android 2Dレンダリングエンジン Skiaの Google Nexus7上での自動並列化 - Skiaを2倍高速化 -

- OSCARコンパイラ
- Skia
- マルチコア

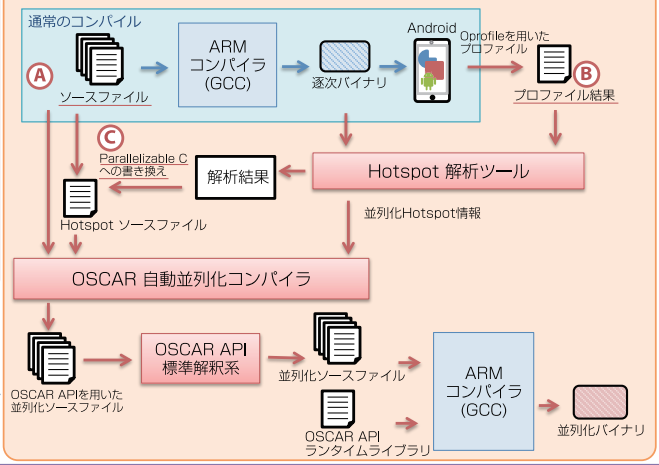
デモンストレーション ビデオ：http://youtu.be/ihj7YYGa5tc

Ⓐ Android 2Dレンダリングエンジン Skia

Android上で、2D描画を行う標準的なライブラリ

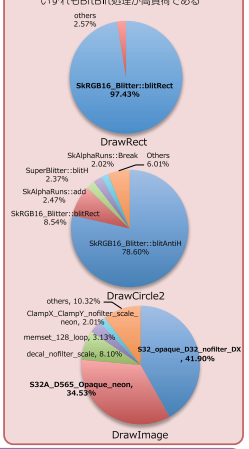


プロファイル情報を用いた自動並列化



Ⓑ Skia プロファイル

実行パス(関数)は各テストによって異なるが、いずれもBitBlt処理が高負荷である



Google Nexus 7

NVIDIA Tegra3 Chip
Processor : NVIDIA Tegra3
ARM Cortex A9 - 4Core
Clock Frequency : 1.2[GHz]



Ⓒ Skia コードチューニング[DrawRect]

```
while (--height >= 0) {
    blend32_16_row(src32, device, width);
    device = (uint16_t*)((char*)device + deviceRB);
}
```

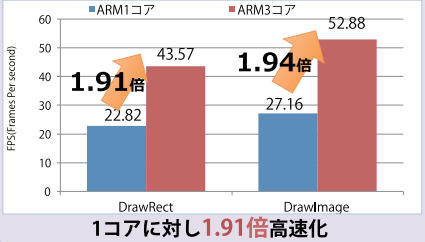
オリジナルソースコード

チューニング後

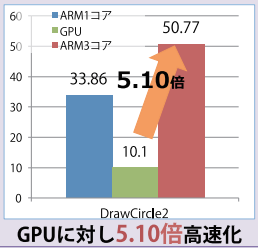
```
for (i = height; i > 0; i--){
    deviceTMP = (uint16_t*)((char*)device + (deviceRB * (height - i)));
    blend32_16_row(src32, deviceTMP, width);
}
```

device変数の依存解消

□NEXUS7で約2倍高速化 2D Canvas ベンチマーク

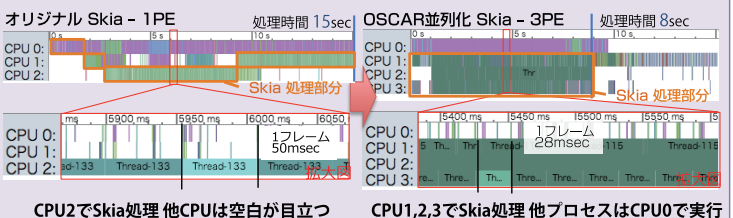


1コアに対し1.91倍高速化



GPUに対し5.10倍高速化

□ 3CPUの実行トレース



CPU2でSkia処理 他CPUは空白が目立つ

CPU1,2,3でSkia処理 他プロセスはCPU0で実行

Embedded Technology 2013