Embedded C:

CoSyコンパイラ開発システムによる DSP用高性能コードの生成

Marcel Beemster/Yoichi Sugiyama
ACE Associated Compiler Experts/Japan Novel Corporation
contact: yo_sugi@jnovel.co.jp

ACE社とは？

- CoSyコンパイラ開発システムを製品として提供
  - コンパイラ生成システム
  - モジュール構造
  - 再構築可能
  - リターゲット可能
  - 豊富な機能
  - 拡張可能
  - 高品質
  - 最適化されたシステム

Japan Novel and CoSy

- 日本ノーベルはACE社の日本での販売代理店
- 日本ノーベルはソフトウェア製品の品質向上を実現する製品を提供
  - コンパイラ評価サービスの提供
  - 組込み機器の自動テストシステム「QualityCommander」の販売
  - PlumHall社製品販売代理店
- コンパイラ評価サービスはC／C++、Embedded－C、DSP-Cテストを提供
- CoSyコンパイラ開発システムはその高い信頼性から日本の組込みシステム開発に大きく貢献できるものと考える

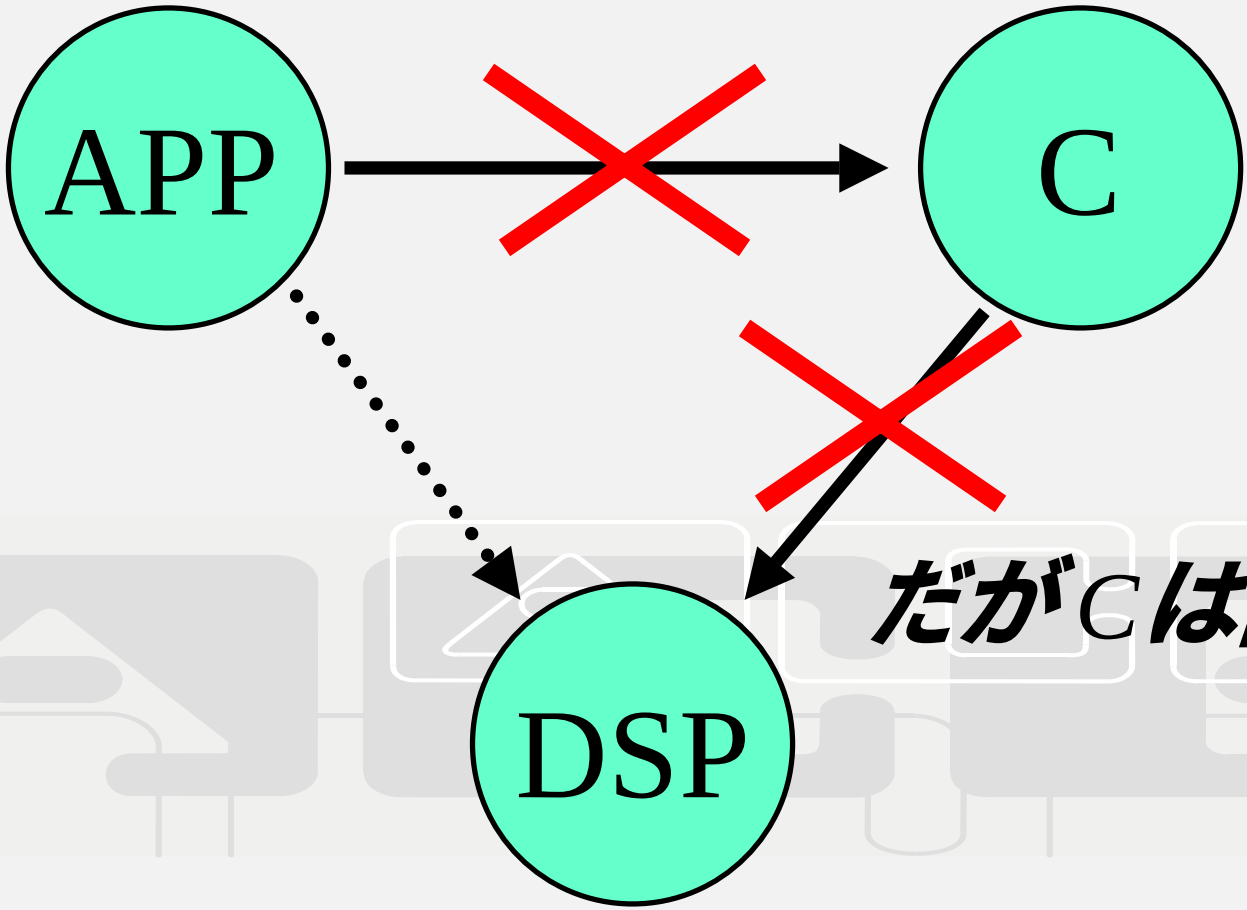
DSPプロセッサをCで記述する利点

- 他のDSPへの移植が容易
  >>> *CPUを特定しない*
- ソフトウェア工学上アセンブラ言語より高品質
  >>> より少ない*不具合*
- アセンブラ･エキスパートを必要としない
  >>> *低開発コスト/早期開発*

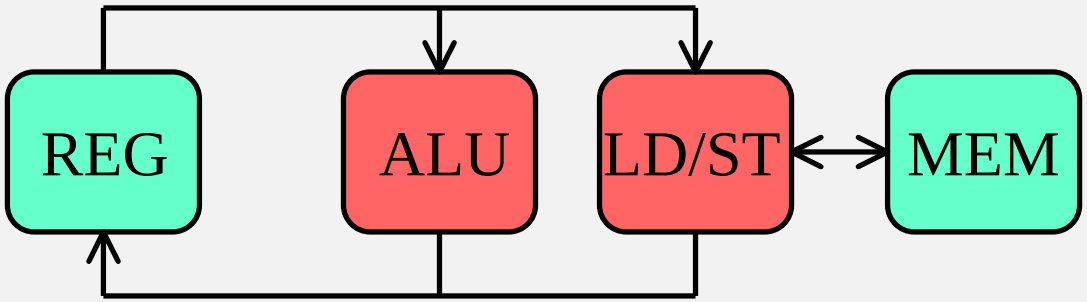
…しかし、それでもアセンブラでのDSPプログラム開発は行われる！　それは、、、

要求性能を満たすため

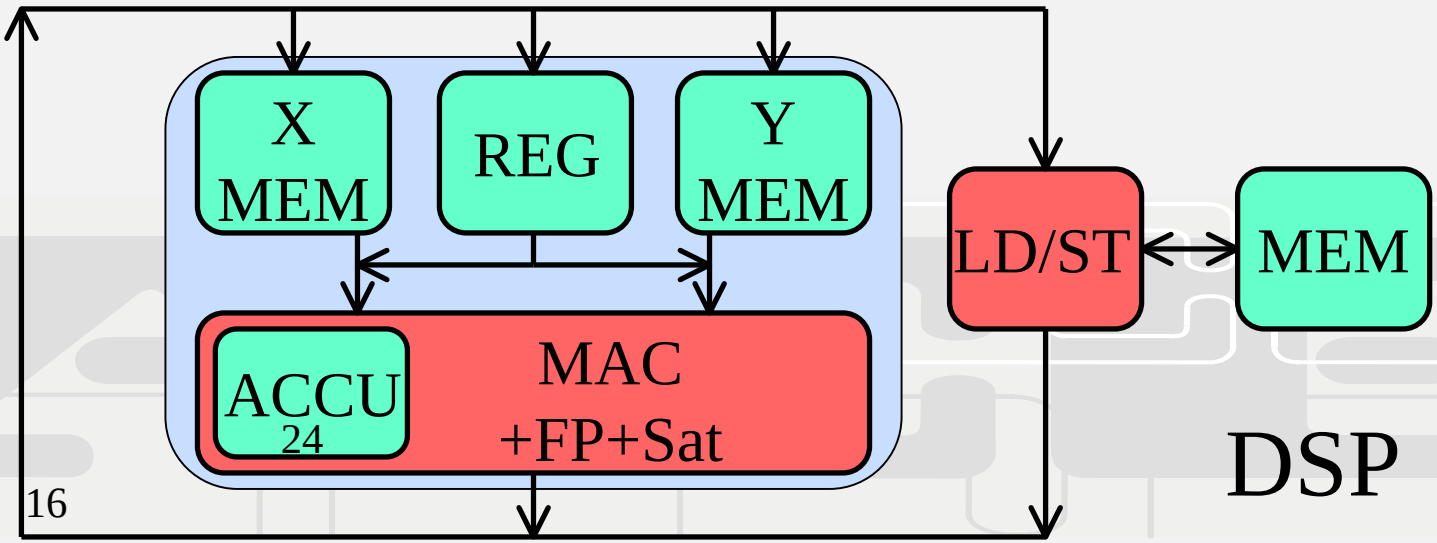
DSPはアプリケーションに特化可能



RISC とDSP アーキテクチャ比較

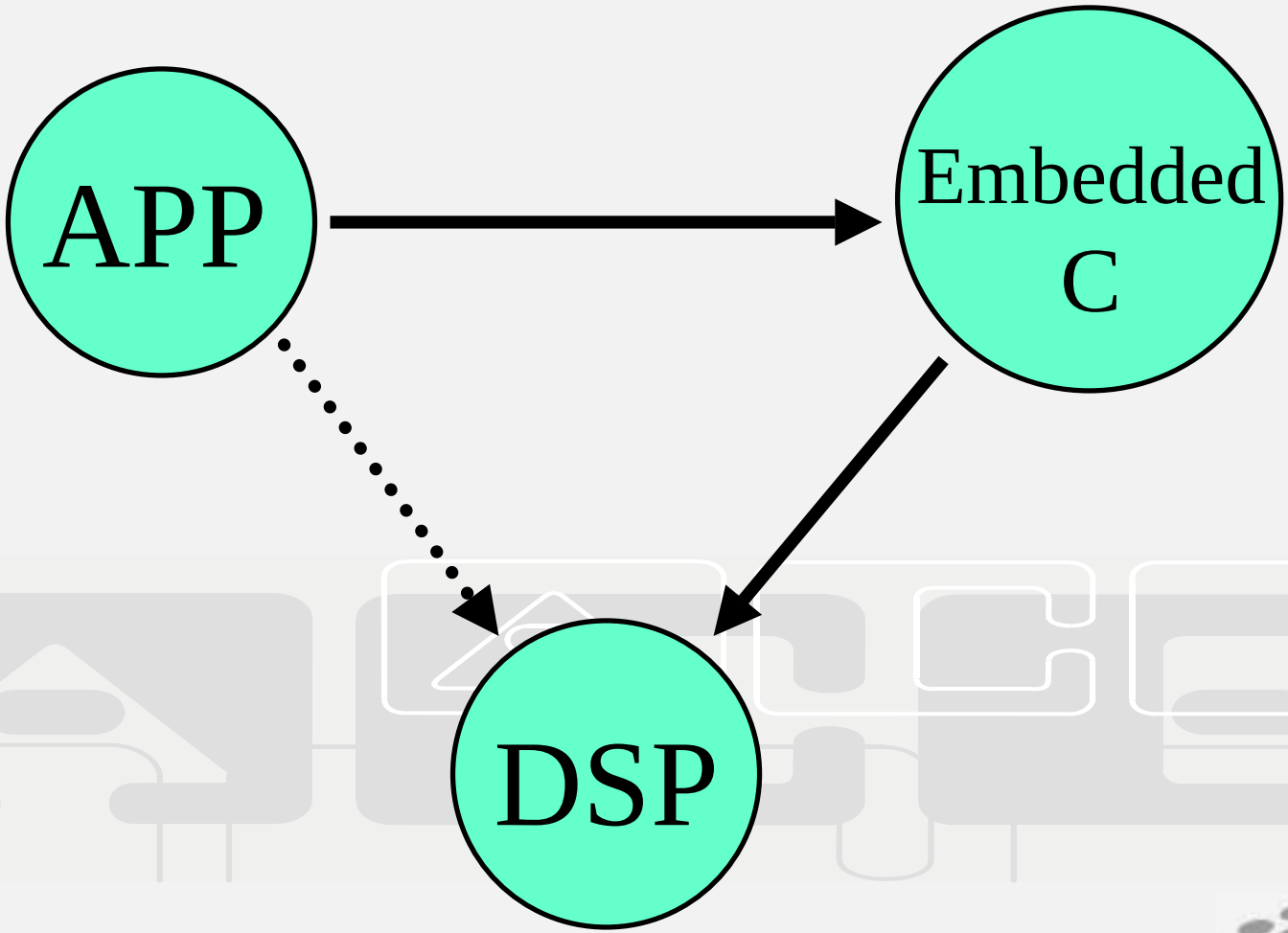


RISC



DSP

Embedded Cの登場



Embedded Cとは？

- DSP-Cの拡張仕様 でACEの働きにより業界標準となる
- 組み込みプロセッサ向けに記述した高級言語コードの性能限界を高めることが可能
- 移植性を高める
- C言語仕様に固定小数点演算、飽和演算、名前付きアドレス空間、およびI/Oアクセスを追加

Embedded C の状況

- ISO テクニカルレポート(TR18037)として標準化の作業が開始され　2004年2月に承認
- 詳細は以下のURLで：

  `www.open-std.org/jtc1/sc22/wg14/`

固定小数点型: _Fract

- 数値レンジは [-1.0, 1.0>
- 整数部はなく小数部のみ使用
- 2の補数演算機構に効率的に実装
- short および long 型のみサポート
- 精度はアーキテクチャ及び実装方法による

_Accum 型

- 整数の使用もサポートしている
  例：[-256.0, 256.0>
- 固定小数点演算の中間値として使用される
- 保存データとしての型ではない
- _Fract 型と同精度である short および long 型

飽和演算: _Sat 修飾子

- 飽和演算の実施：

```
-0.75r + -0.75r == -1.0r
```

- 保存データとしての型ではない
- SN比向上のため信号処理ではレンジの境界値近辺での数値操作が必要の場合が多い

名前付きアドレス空間

- キーワードは特にない
- 例:

```
X int a[25] ;
X int * Y p ;
```

- 特定のアドレス空間に対するアクセスの制限を可能とする
- 制限の無い (一般的な) ポインタ型はすべてのアドレス空間へのアクセスが可能

記述例

```
X fract coeff[N] = { 0.7r, ... } ;

fract fir( Y fract *inp ) {
   int i ;
   accum sum = 0.0k ;
   for( i = 0 ; i < N ; i++ ) {
      sum += coeff[i] * (accum)*inp++ ;
   }
   return (sat fract)sum ;
}
```

DSP-C における性能比較

- Ericsson 内製通信プロセッサ(16-bit, VLIW, dual MAC, dual 32-bit ld/st)向けCoSy DSP-C コンパイラを使用
- NEC μPD77016 プロセッサ(16-bit 標準 DSP)向けCoSy DSP-C コンパイラを使用
- 比較内容:
  - 基本操作
  - NECアプリケーション
  - MiBench ループ

Ericsson, NEC 及び University of Michigan様に感謝致します

基本操作

- Ericssonコンパイラを使用して最大性能が出る様に設定

	ISO C Cycles/Size	*CoSy DSP-C* Cycles/Size
飽和演算	10/46	0*/0*
FIR**フィルタ**	2/74	1*/26
配列コピー	2/12	1*/12

サイクル数: 内部ループでのサイクル数、飽和演算を含まず
***マーク付きの結果が最適解です**

NEC アプリケーション

NEC µPD77016 CoSyコンパイラを使用
アプリケーション別総クロックサイクル比較

	ISO C	*CoSy DSP C*	*Ratio*
制御部	3946	3890	1.0
DSP 1	5144	550	9.4
DSP 2	168546	48064	3.5
DSP 3	2822	349	8.1

10倍近くの改善あり!　さてどうやって?

MiBench オリジナル記述

- GSM通信よりプロシージャを流用:
  Short_term_analysis_filtering
- 内部ループ:

```
for (i = 0; i < 8; i++) {     /* YYY */
      ui    = u[i];
      rpi   = rp[i];
      u[i]  = sav;
      zzz   = GSM_MULT_R( rpi, di  );
      sav   = GSM_ADD   ( ui,  zzz );
      zzz   = GSM_MULT_R( rpi, ui  );
      di    = GSM_ADD   ( di,  zzz );
}
```

MiBench DSP-C 記述 + Accum

- DSP-Cを用いて Accumとともに記述

```
for (i = 0; i < 8; i++) {     /* YYY */
    ui   = u[i];
    rpi  = rp[i];
    u[i] = sav;
    sav  = ((long accum)ui) + rpi * di ;
    di   = ((long accum)di) + rpi * ui ;
}
```

MiBench ループ

- Ericsson コンパイラを使用、最高性能が出る様に設定して内部ループのサイクル数をレポート

	ISO C Cyc./Size	*CoSy DSP-C* Cyc./Size	*DSP-C+ accum* Cyc./Size
`S_t_a_f`	32/228	4/112	3*/112

* マーク付きの結果が最適解です

MiBenchループ結果の説明

- オリジナルでは保守性向上のため *macros*を用いて記述しているが内部ループにおいて不必要な飽和演算が発生する
- 標準ISO C を用いた固定小数点演算のエミュレーションでは多数のレジスタを必要とするがレジスタ数に制限がある場合 *spill-code* となる
- 累積効果による内部ループでのオーバーヘッドは１０倍にもなる

CoSyにおける Embedded C

- Embedded C記述の完全なフロントエンドのサポート
- データ型のサイズを自由に設定可
- IR内部表現として統合化
- CoSy最適化エンジンによる最適化の実施
- エミュレーションとサポートライブラリの用意
- 例題コンパイラの用意
- Windows, Linux, Solarisプラットフォームのサポート
- 1998年より CoSyの DSP-Cオプションとして提供
- Embedded Cオプションとして 2005年より提供

Embedded Cの世界

- **Cコンパイラの DSP拡張としてこれまでの開発例**: **Philips REAL, Adelante Saturn, NEC µPD77111/210, TI TMS320C54x, Analog Devices SHARC, MMDSP+, META RISC/DSP 等々…**
- **ユーザー/サポーター: ACE, Ericsson, AbsInt, NullStone, Mentor Graphics XRAY, Japan Novel, 等々…**

Embedded Cの利点

- 高級言語を用いて高効率の組み込みプロセッサ用プログラムの開発が可能
- DSPアーキテクチャにおける主要な性能関連項目の標準化
- I/Oハードウェアアクセス方法の標準化
- 結果としてソフトウェアの開発メソドロジおよび組み込みアプリケーションの移植性を飛躍的に改善
- 経済的利点：開発期間短縮、柔軟性

CoSyを Embedded Cとともに使用しよう!

marcel@ace.nl/yo_sugi@jnovel.co.jp
www.embedded-c.org

暗黙的拡張

```
_Fract f = (_Fract)(0.1r * 2.5) ;
```

- **順序:** `int`, `_Fract`, `_Accum`, `float`
- **Cにおける新しい考え方, 混合型演算:**

```
f = 3 * 0.1r ;
```

整数と固定小数点の混合のみ

unsigned 固定小数点型

- レンジ: `[0.0, 1.0>`
- 画像処理に有効
- すべてのDSPプロセッサで使用可能ではない (飽和演算)

実現されなかった機能

- 循環バッファ
- モジュロ固定小数点演算のための `_Modwrap` 修飾子
- 複素固定小数点型
- BCDデータ型

Embedded Cの性能と移植性 (1)

- 通常 DSPアルゴリズムは一定の精度を必要としているが Embedded Cの固定小数点型の精度は保証されていない
- メモリ修飾子キーワードは Embedded Cでは定義されていない
- Cコードでの扱いに良く似ている

Embedded C で記述されたプログラムが
１００％の移植性を持っている訳ではない

Embedded Cの移植性 (2)

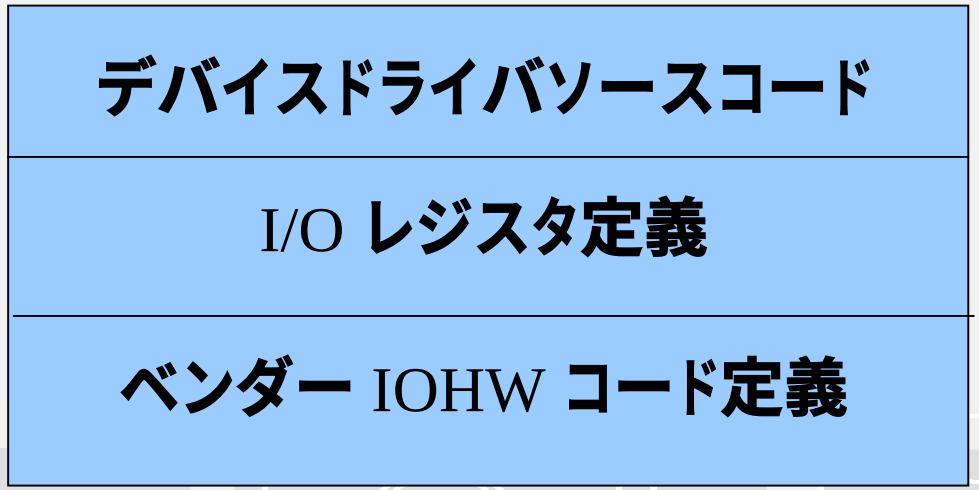
- Embedded Cは目的とするプロセッサのアーキテクチャに合わせて記述する
- 例：24ビット精度のDSPアプリケーション は、そのままでは16ビットDSPプロセッサで動作しない
- Embedded Cの記述は特別なアーキテクチャへの対応も可能

NEC コードサイズ比較 (バイト)

	ISO C	DSP-C	Ratio
Control	442	414	1.1
DSP 1	350	90	3.9
DSP 2	2152	1155	1.9
DSP 3	3807	2781	1.4

I/Oハードウェアアドレシング

- デバイスドライバでの使用を考慮
- 3階層モデルを基本とする



- オーバヘッドの最小化が期待出来る