情報処理入門

8章「エミュレーション」

中島康彦

§8. 1 ハードウェア化が良いとは限らない

命令セット・アーキテクチャや実行モデルが世代交替

- メーカは，新製品に集中投資したい．旧型のハードウェアに縛られたくない．
- ユーザは，長年使っているソフトウェアを使い続けたい．できれば新しい機能も同時に使いたい．

他のアーキテクチャ上で動く優れたソフトウェアが羨ましい

- メーカは，自社ハードで動くソフトを増やしたい
- ユーザは，ソフトごとにハードを使い分けるのは面倒

要するに，開発時期や対応機種が異なる様々なプログラムを 最新のコンピュータ上で動かしたい

§8. 2 資産の形態に応じて対応

再コンパイル(プログラムの移植)
- 高級言語によるソースプログラムが存在する場合
- 最新コンピュータ本来の性能を引き出せる
- ただし，ソースが存在しても再コンパイルできないこともある
- そもそも，一般のユーザには無理！

インタプリタ(命令の逐次解釈実行)
- ロードモジュールしか存在しない場合
- 全体として所要メモリ量を小さくできる
- 実行速度は最も低速(100分の1以下)

§8. 2 資産の形態に応じて対応

静的命令変換(予め旧命令を新命令に変換)
- アセンブリ言語によるソースプログラムが存在する場合
- 命令とデータが区別できる場合
- アプリケーション・プログラムは変換できることが多い
- ハードウェア依存度によっては，人手の介入が必要

動的命令変換(実行しながら新命令に変換)
- ロードモジュールしか存在しない場合
- 実行してみないと，どこが命令かわからない場合
- 人手の介入が不要
- 比較的高速(10分の1程度)
- 最も現実的な方法

§8. 3 事例

銀行等基幹業務用汎用コンピュータのプログラムを最新パソコン上で動かす

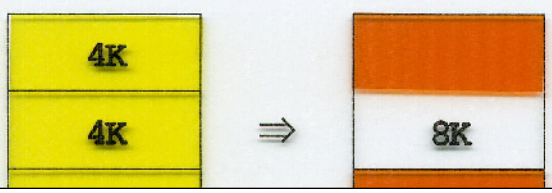
- 旧命令は，Mアーキテクチャ
- 新命令は，SPARCアーキテクチャ
- アプリケーションだけでなく，OSをまるごと動作させる.

United States Patent 6,031,988
特願H9-198283

課題１

ハードウェアの選択

- 低コスト
- I/Oを含む詳細仕様が入手可
- Big-endian byte order
- Mの汎用レジスタ（32bit×16）を常駐可
- Mの浮動小数点レジスタ（64bit×8）を常駐可
- 演算方式（条件コード，例外，精度）が類似
- ４Ｋバイト単位のアドレス変換／保護機構



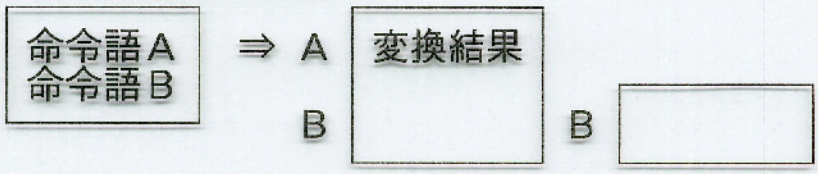
課題２

変換結果の構造

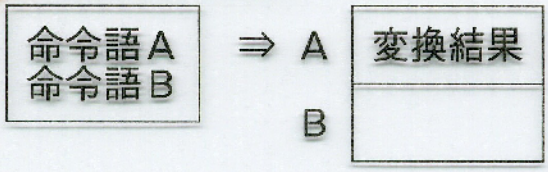
変換コストが無視できるほど再利用されないと予想

自己変更プログラムへの対応（変換結果の無効化）

・ブロック単位の変換



・命令語単位の変換



課題３

例外の検出

明示的な検査を極力減らす

C　　2，4（6，8）

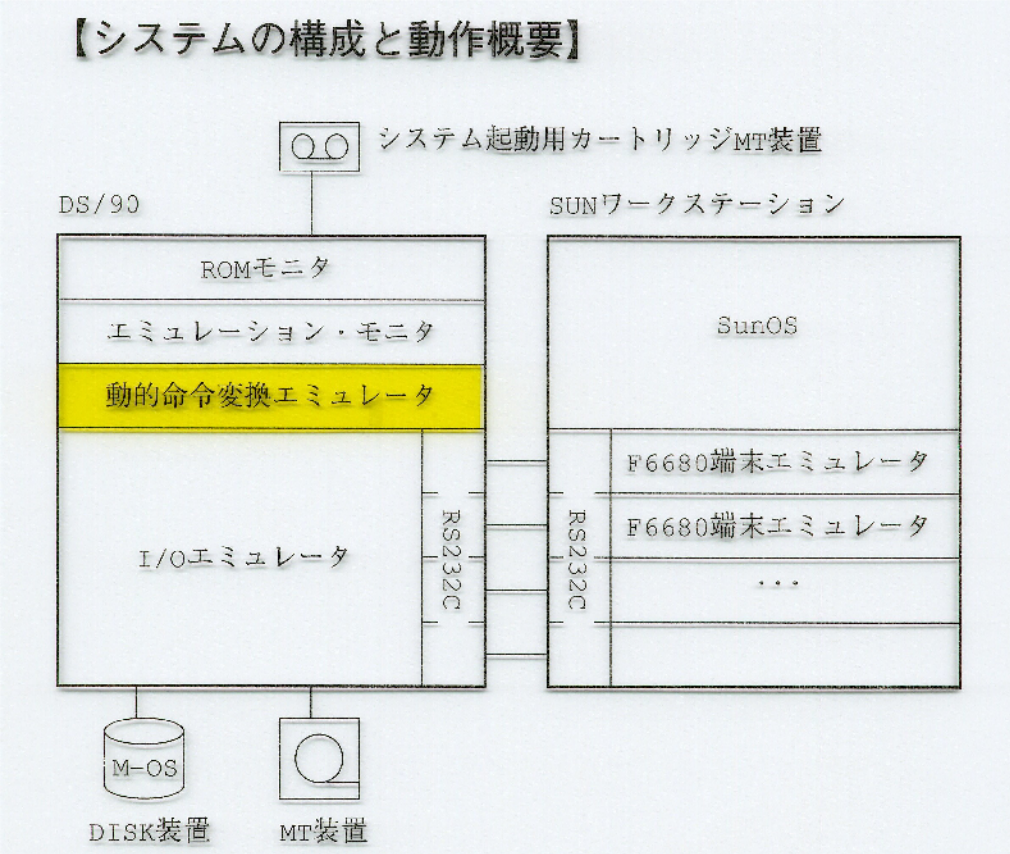
↓

加算	4＋R6		
加算	＋R8		
マスク	0xFFFFFF	⇒	X
ロード	BASE+X	⇒	D
IF	R2＝D	⇒	結果0
IF	R2＜D	⇒	結果1
IF	R2＞D	⇒	結果2
加算	PC＋4	⇒	新PC

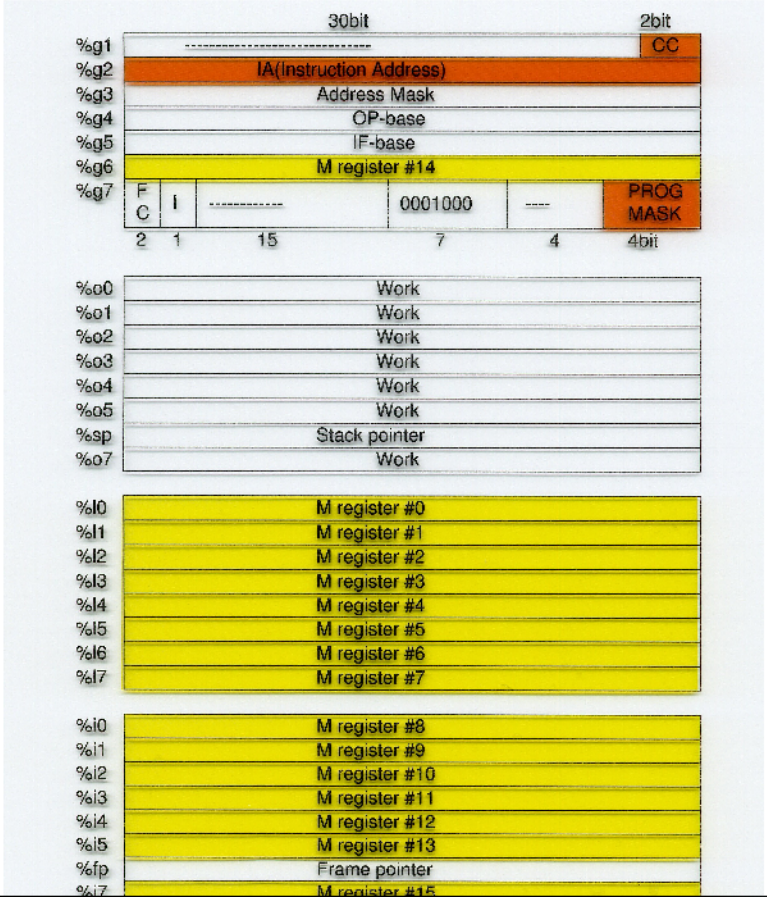
ページフォルト／保護例外

精度完全一致のため
演算例外は明示的に検出

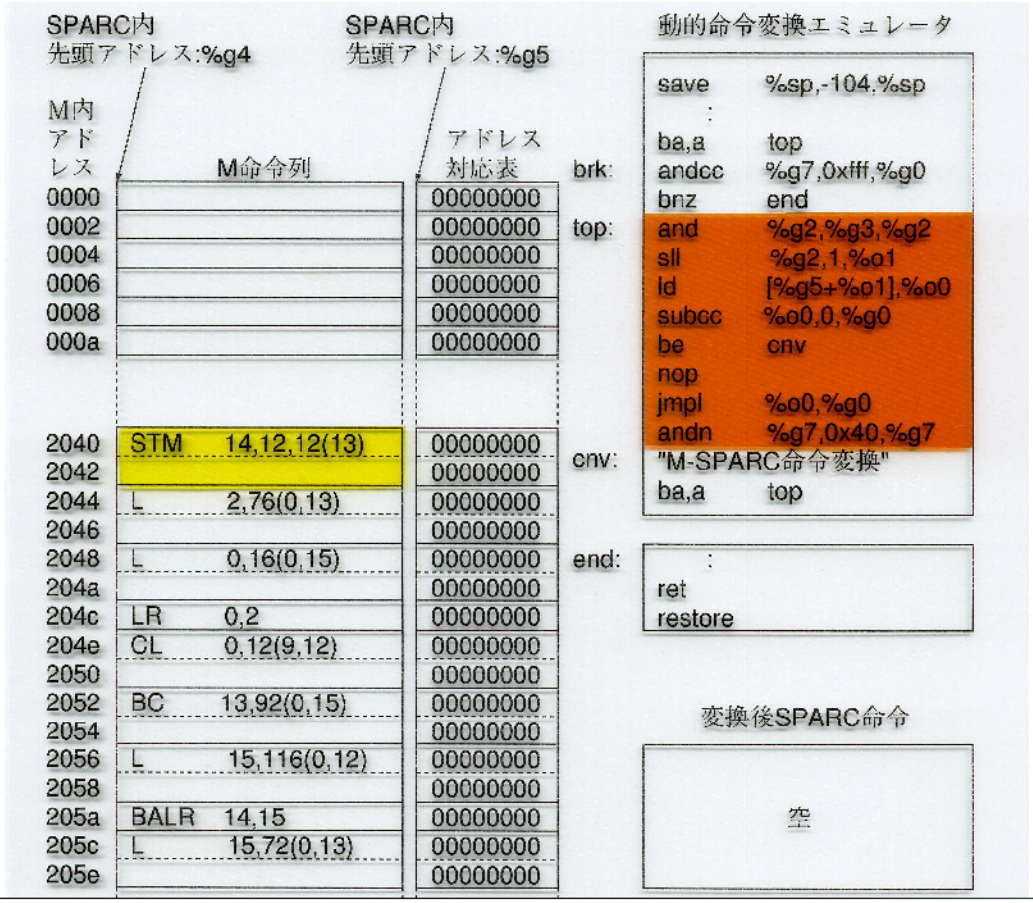
§8. 4 ホストコンピュータの構成



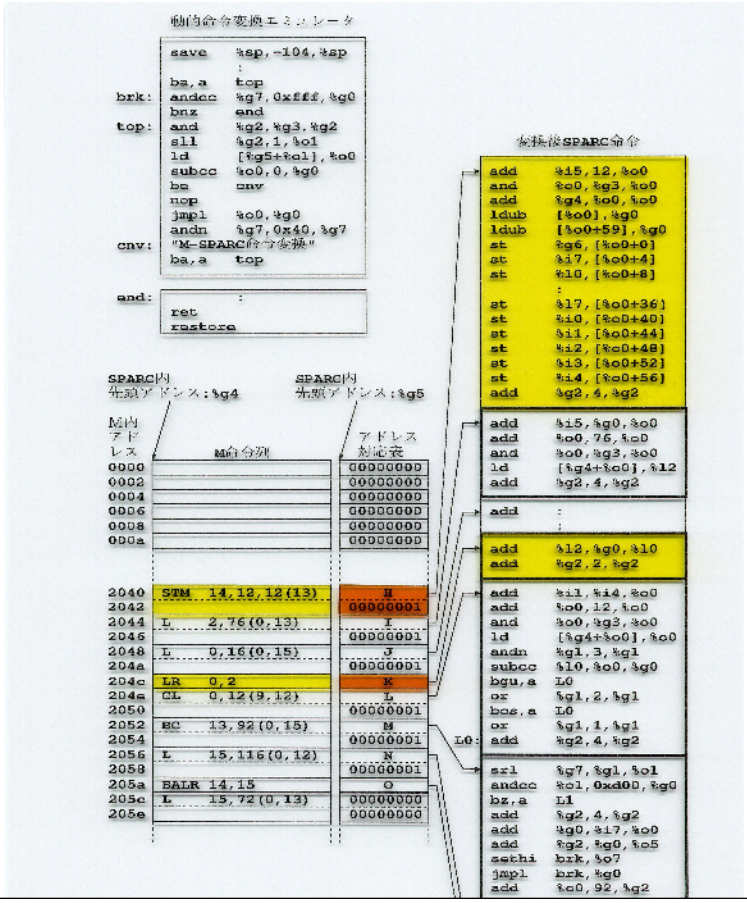
§8. 5 レジスタの対応付け



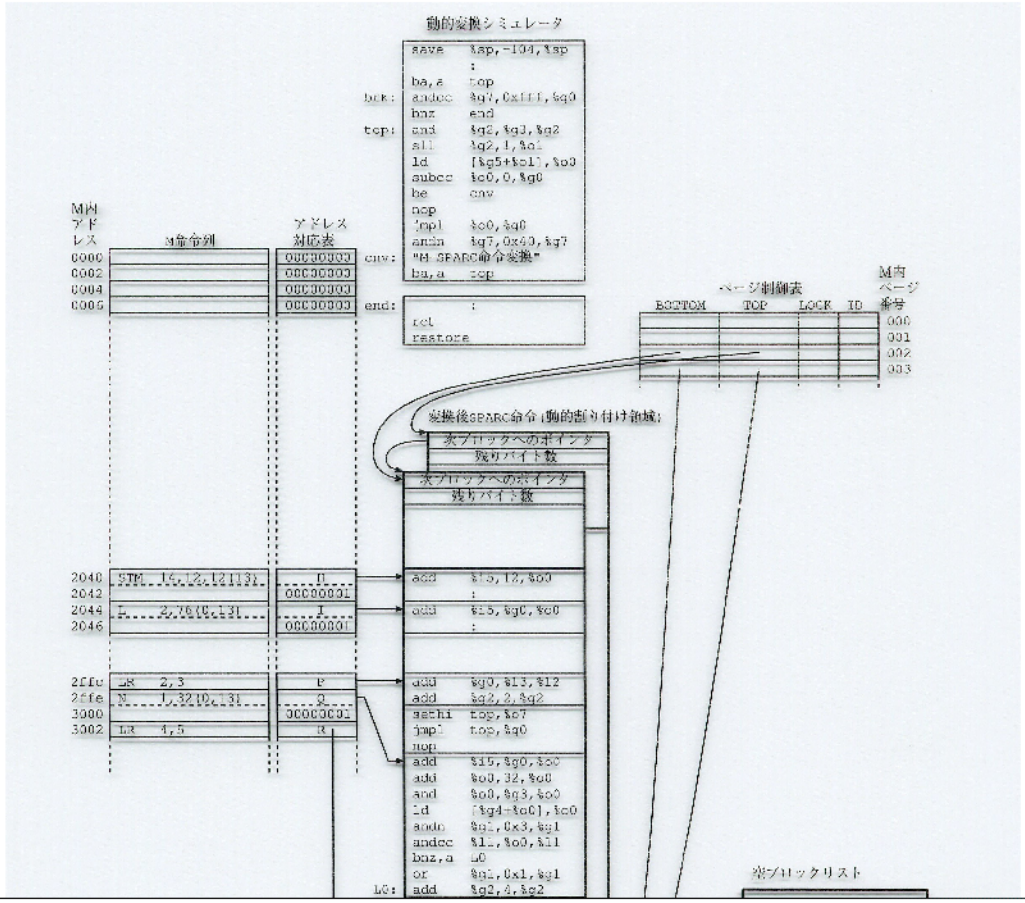
§8. 6 命令変換前の状態



§8. 7 変換しながら実行中



§8. 8 全体像



§8. 9 実行速度の見積り

	FMIX−RH FMIX−AIM	SBS_BL2 S+P	step数
BC（taken）	18%	16%	11+12
BC			7
BCR（taken）	2%	2%	10+12
BCR			7
L	15%	15%	5
ST	7%	8%	5
TM	7%	6%	10
LA	5%	4%	7
LR	5%	5%	2
LTR	3%	1%	5
SLR	2%	2%	7
MVC（4b）	2%	2%	22
MVI	2%	2%	5
		加重平均	9

§8. 10 実測して検証(小さいプログラム)

【命令ワーキングセットが小さいプログラム】

	Dhrystone-2.1 回転数=10000	Stanford-integer
プログラム実行時間 (a)	1．3秒	2．9秒
実行M命令数 (b)	9．6Mstep	24．Mstep
変換を要したM命令数 (c)	1．2Kstep	3．2Kstep
再利用率 ((b-c)/b)	99．99%	99．99%
実行SPARC命令数 (d)	94．Mstep	223Mstep
命令数の比 (d/b)	9．8倍	9．3倍
SPARC-CPI (a×60MHz/d)	0．83	0．78
M-MIPS値 (b/a)	7．4MIPS	8．3MIPS
変換後命令を蓄積しない場合 (e)	16秒	38秒
実行時間比 ((e-a)/a)	11倍	12倍
1次キャッシュミス回数/M命令	0．52回	0．17回
2次キャッシュミス回数/M命令	0．048回	0．012回
ミスアラインのオーバヘッド (f)	0．00%	0．00%

§8. 10 実測して検証(大きいプログラム)

【実運用状態に近いI/O頻度のジョブ】

	自己変更検出無し	自己変更検出あり	
		書込み保護による検出	ストア命令による検査
プログラム実行時間 (a)	457秒	1．4K秒	(机上計算)
実行M命令数 (b)	518Mstep	←	
・内スーパバイザ	193Mstep	←	
・内プロブレム	325Mstep	←	
変換を要したM命令数 (c)	114Kstep	←	
再利用率 ((b-c)/b)	99．98%	←	
M-MIPS値 (b/a)	1．1MIPS	0．35MIPS	自己変更検出無しの1割減
・モニタ除くM-MIPS値	2．7MIPS	0．45MIPS	
1次キャッシュミス回数/M命令	0．88回	1．09回	
2次キャッシュミス回数/M命令	0．14回	0．20回	
ミスアラインのオーバヘッド (f)	1．83%	0．63%	

§8. 11 コンピュータの特性が変わる

【命令キャッシュの挙動】

M命令列 ⇒ 命令量 １０倍 ⇒ ＳＰＡＲＣ命令列

ヒット率低下

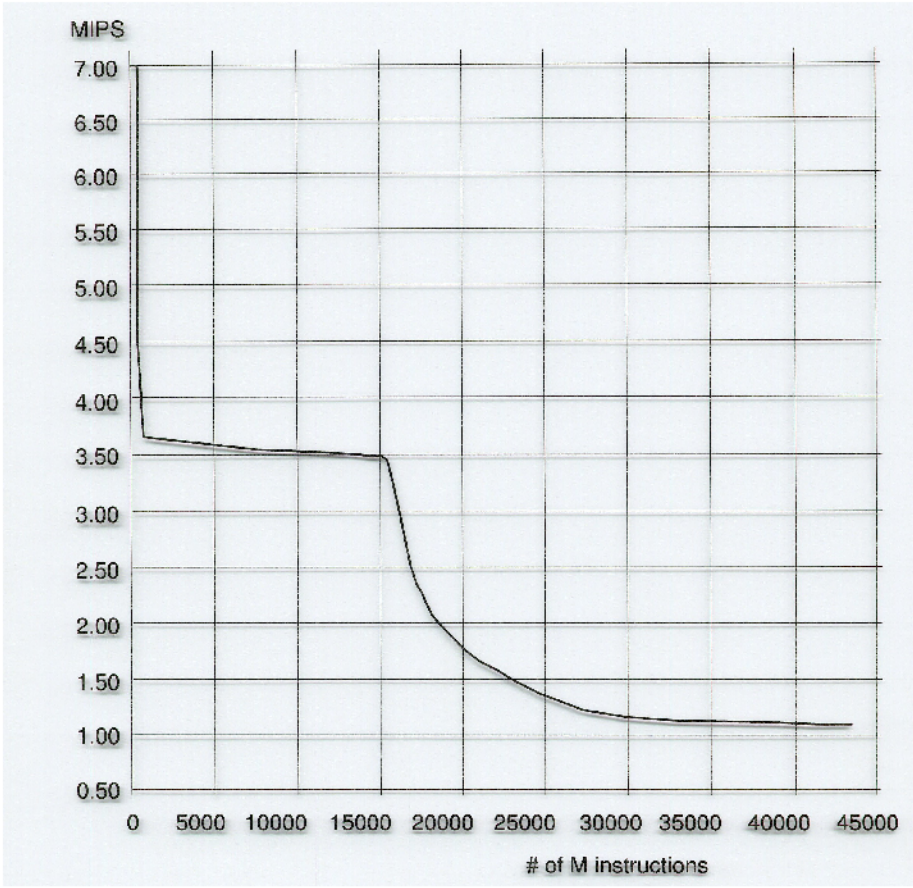
【データキャッシュの挙動】

ＬＤ命令 ⇒ 出現頻度 １／１０ ⇒ ＬＤ命令

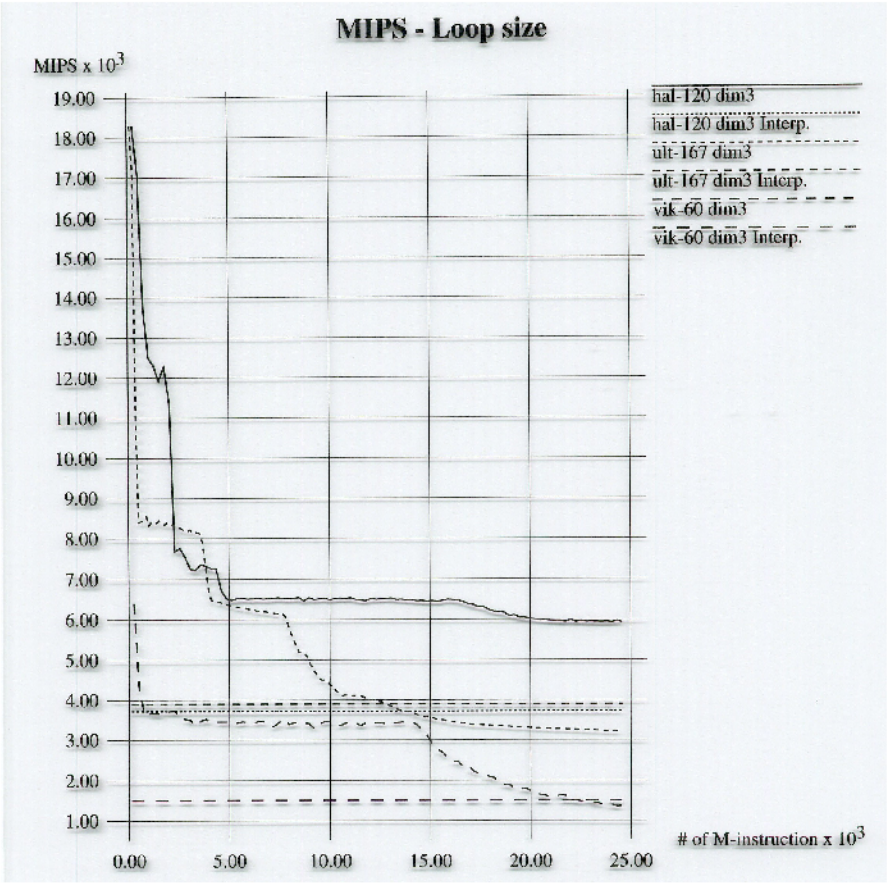
ヒット率同等

ミスの影響は薄まる

§8. 12 命令キャッシュが小さく見える



§8. 13 ホストコンピュータによる違い



§8. 14 製品化

大型汎用コンピュータ用OS(MSP/XSP)をSPARCパソコンで

- 1998年～ 試験機稼働(エミュレータ開発は僅か3名)
  意外にも，CEなど保守部門が抵抗
  保守方法が変わるのはイヤだ！

オフィスコンピュータ用OS(ASP)をIntelパソコンで

- 1992年～ 当時は独自専用CPUを採用
  Kシリーズ
- 1999年～ ハードはIntel-PC(動的命令変換)
  GRANPOWER6000シリーズ
- 2003年～ ハードはIntel-PC(動的命令変換)
  PRIMERGY6000シリーズ

今日はここまで