Empowered by Innovation NEC

カーネルサイズ・使用メモリ自動検証ツール

2006年1月20日
NEC Linux推進センター
池田 宗広
m-ikeda(__at_mark__)ds.jp.nec.com

0. 目次

1. 背景
2. どんなツールか？
3. ツールの構成と機能
4. ツールはどう使えるか？
5. 測定準備 ～baseカーネルの検討～
6. これまでの測定結果
7. 今後の予定
8. 議論

1. 背景 (1)

- LinuxTinyの分析・効果検証（2005年7月）

カーネル種類	vmlinux[KB]	bzImage[KB]	free mem[MB]
vanilla-full	7709	3071	78.6
vanilla-size	2833	1061	85.0
tiny	2360	869	86.0

- **カーネル種類**
  - vanilla-full　：ほとんどディフォルト設定。
  - vanilla-size　：最小限のドライバ・fsのみenable、Optimize for size = y
  - tiny　：LinuxTiny適用。LinuxTinyで追加された設定項目は全てサイズが縮小する方向に設定
- **測定条件**
  - Kernel 2.6.10
  - x86(PC conpatible)
  - gcc 3.3.5
  - Machine : Celeron 400MHz / 96MB RAM / 10GB HD
  - free mem は各種デーモン停止（起動しない）状態で起動直後にfreeで測定

LinuxTinyはカーネルサイズ縮小に効果あり。
しかし、カーネルサイズを縮小するためにより重要なのは
カーネルコンフィグレーションの絞り込み。

1. 背景 (2)

- **サイズ・使用メモリを最小化するためにカーネルコンフィグレーションを絞り込むには・・・**

  1.make menuconfig
  2.make
  3.サイズ測定
  4.インストール、リブート
  5.free(1)

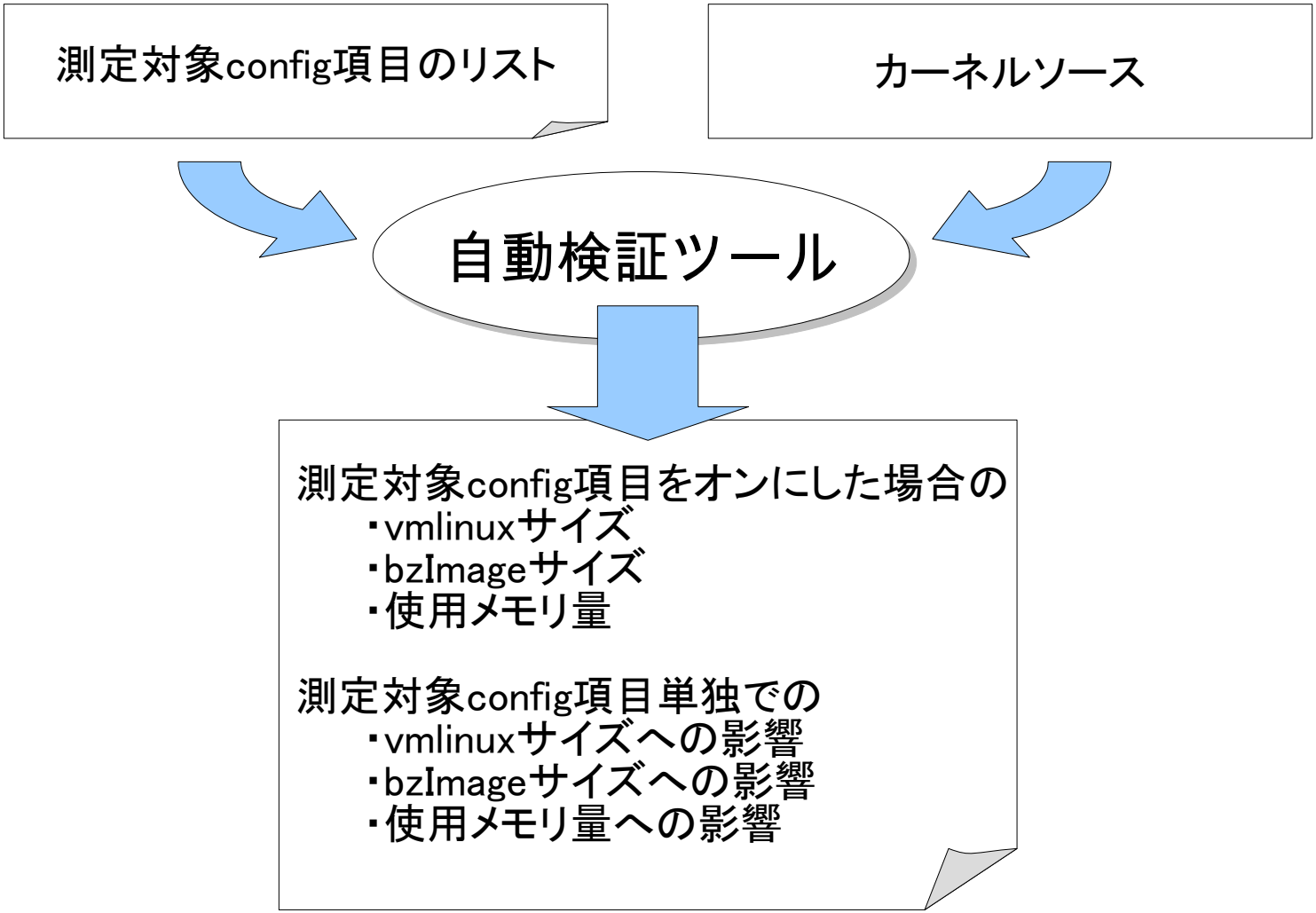
  

- **Config項目数 (2.6.12.3)**
  - Total : 5338 (共通 3028 + arch以下 2310)
  - ARCH=i386 : 3187 (共通 3028+ arch/i386以下 159)
  - ARCH=arm : 3314 (共通 3028 + arch/arm以下 286)

コンフィグレーションごとのサイズ・メモリ使用量増分を自動的に測定・検証するツールを開発・公開し、データを共有すればみんなハッピー！

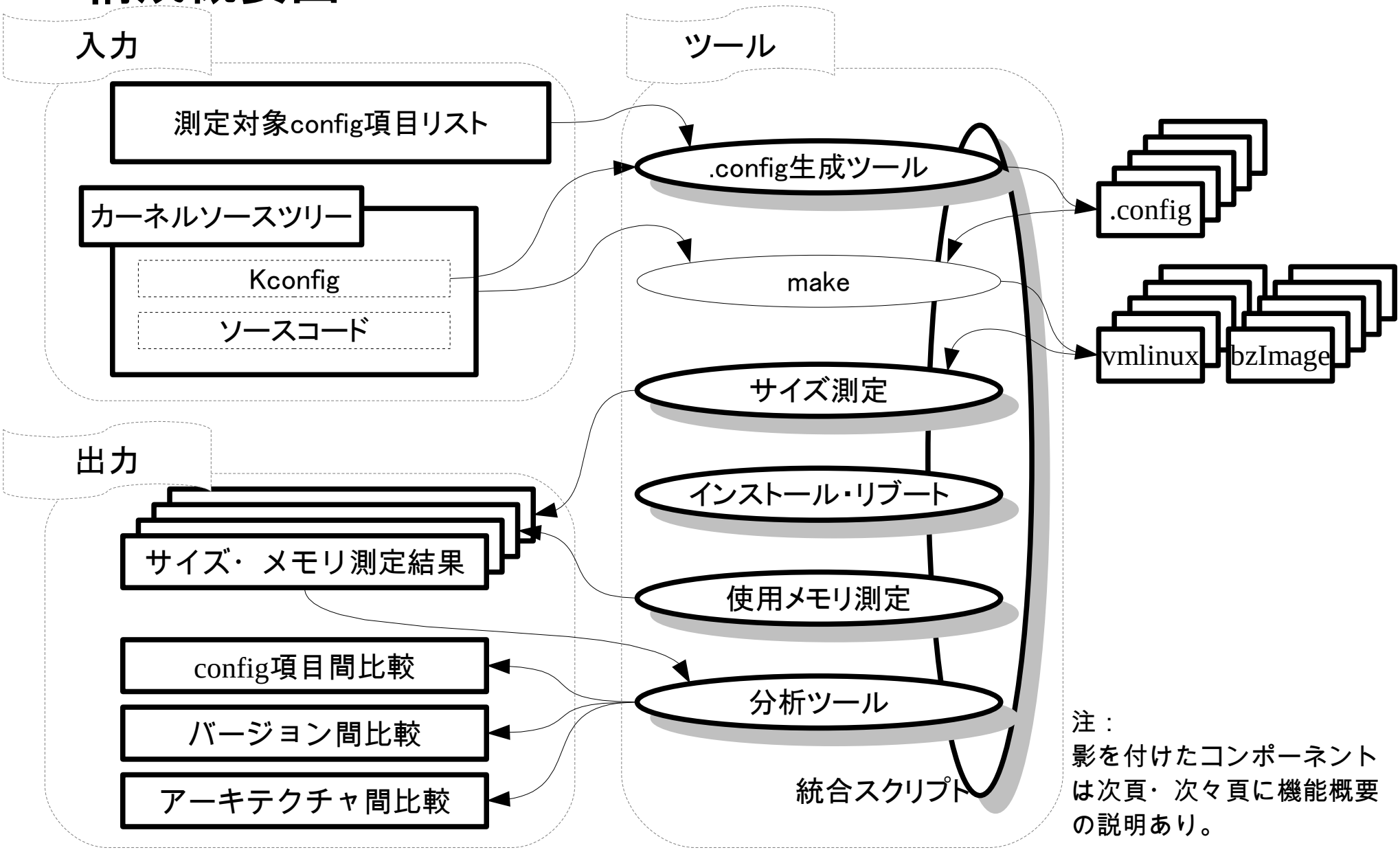
2. どんなツールか？

- 指定したカーネルコンフィグレーション項目とサイズ・メモリ使用量との関係を自動的に測定するツール（群）。



3. ツールの構成と機能 (1)

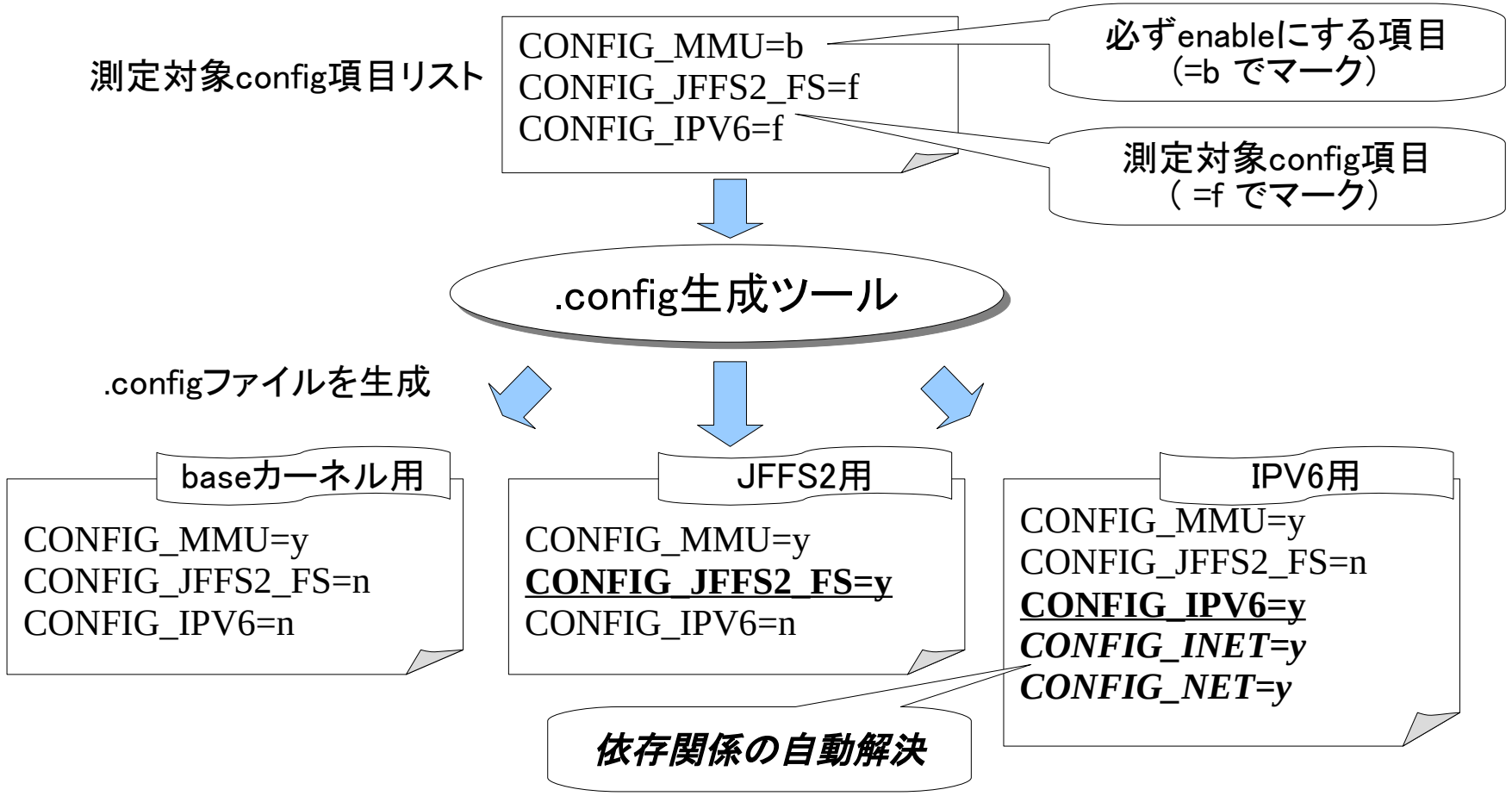
- **構成概要図**



注：
影を付けたコンポーネントは次頁・次々頁に機能概要の説明あり。

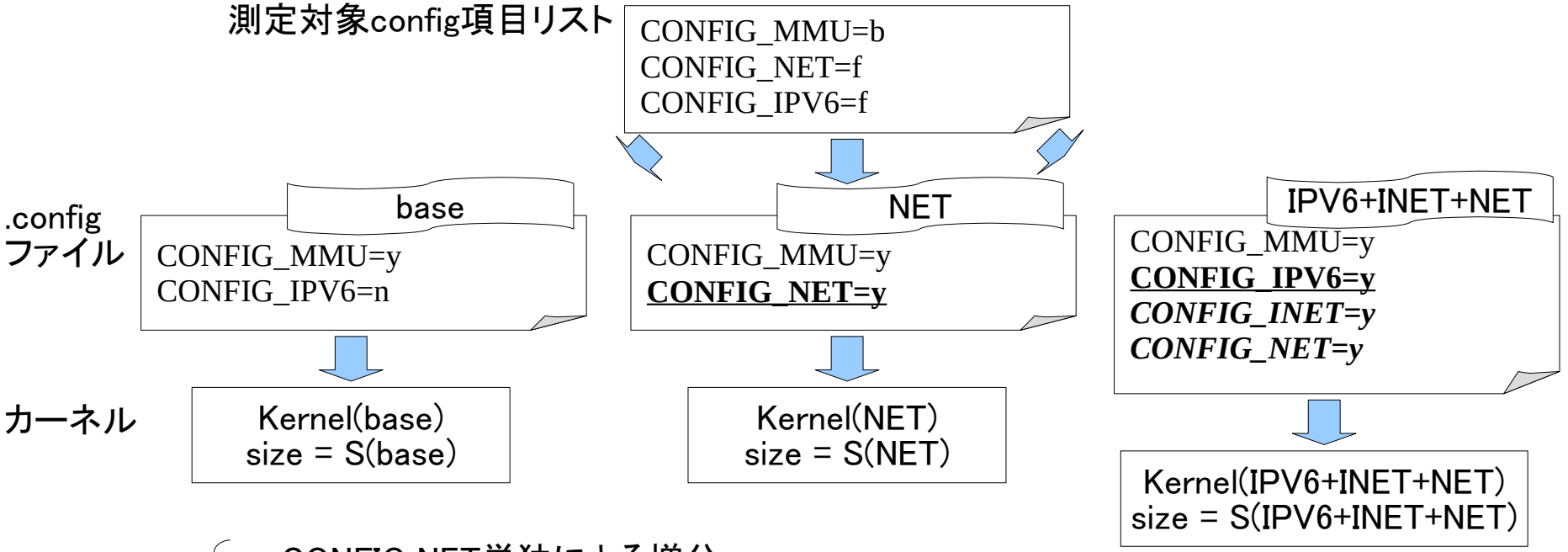
3. ツールの構成と機能 (2)

- .config生成ツール
  - 測定対象config項目リストを読み込み、測定対象config項目を一つ一つenableにした.configファイルを生成する。
  - 依存関係を自動解決。
  - カーネルソースのKconfigパーサ（scripts/kconfig/以下）を流用。



3. ツールの構成と機能 (3)

- **サイズ測定**
  - カーネルイメージ（vmlinux）、圧縮イメージ（bzImage）のファイルサイズを測定する。
  - 測定対象config項目単独でのサイズ増分を（可能な限り）算出。



○算出可能
- CONFIG_NET単独による増分
  ⊿s(NET) = S(NET) − S(base)
- CONFIG_IPV6+CONFIG_INETによる増分
  ⊿s(IPV6+INET) = S(IPV6+INET+NET) − S(base) −⊿s(NET)

✗算出不能
- CONFIG_IPV6単独による増分
  ⊿s(IPV6) = S(IPV6+INET+NET) − S(base) − ⊿s(INET+NET)
  . . . ⊿s(INET+NET) が得られていないため算出できない。

3. ツールの構成と機能 (4)

- **インストール・リブート**
  - ターゲットボードの電源制御用リレーボックスをホストのパラレルポートに接続、制御するプログラム。
- **使用メモリ測定**
  - 起動直後に sysinfo(2) で測定し、結果ファイルに記録するプログラム。(sysinfo を直接使うのは、procfs をサポートしない場合に備えて)
  - 測定対象config項目単独での使用メモリ増分を(可能な限り)算出。
- **統合スクリプト**
  - アーキテクチャ、カーネルバージョン、測定対象config項目それぞれのループで測定を実行するスクリプト。
  - 進捗をファイルに記録するため、リブート時に異常が発生した場合でも続きから実行可能。
- **分析ツール**
  - 測定対象config項目間のサイズ・使用メモリ比較
  - ある測定対象config項目の、カーネルバージョン間でのサイズ・使用メモリ比較
  - アーキテクチャ間のカーネルサイズ・使用メモリ比較

4. ツールはどう使えるか？

- **組み込みシステム開発者にとって**
  - 機能的には同じようなドライバ、fsがいくつかあるけれど、一番サイズの小さいものを選択したい。
    ➔各ドライバ、fsに対応するconfig項目の測定データを元に選び出すことが可能。
  - 新しいバージョンのカーネルを使いたいけれど、どのくらいサイズが大きくなるんだろうか？
    ➔バージョン間のサイズ比較データから見積もりが可能。
- **カーネル開発者にとって**
  - 非効率的な実装箇所はどこだ？
    ➔config項目間のサイズ比較、バージョン間のサイズ比較から抽出が可能。
  - パッチがサイズに与える影響はどのくらいだろうか？
    ➔パッチ適用カーネルとパッチ非適用カーネルからデータを自動測定、結果を比較して見積もりが可能。

5. 測定準備 ～baseカーネルの検討～ (1)

- **baseカーネル**

  各測定対象config項目のサイズ・メモリ使用量を測定するためには、測定対象config項目がひとつもenableにされていない基準カーネル
  (＝baseカーネル)が必要。

- **Config項目の種類**

  Config項目は以下の3種類に大別できる。

  1. **起動に最低限必要なconfig項目**
     例 : CPUアーキテクチャ指定、rootfsのあるデバイスのドライバ etc.
  2. **独立config項目**
     サイズがその項目単独で計測できるもの。
     例 : 各種ドライバ、fs etc.
  3. **非独立config項目**
     サイズがその項目単独では計測できない(他のconfig項目の設定によってサイズが変動する)もの。
     例 : SMPサポート、printkサポート、procfs、サイズ最適化 etc.

- **非独立項目の組み合わせパターン**

  「非独立config項目」は単独でのサイズ測定が行えない。よって組み合わせパターンを検討し、baseカーネルに適用する必要がある。つまり、

  baseカーネルでenableにするconfig項目
  ＝ 測定対象configリストで「=b」とマークするconfig項目
  ＝ 起動に最低限必要なconfig項目 ＋ 非独立config項目

  であり、baseカーネルおよび測定対象config項目についての測定結果は非独立config項目の組み合わせにパターンに依存する。

5. 測定準備 ～baseカーネルの検討～ (2)

- **非独立項目の組み合わせパターン (i386)**

  今回の検証では、下表に示す複数の非独立config項目組み合わせパターンを使用した。

  各パターンのbaseカーネルでenableにされるconfig項目
  ＝ 各パターン用の測定対象configリストで「=b」とマークされるconfig項目
  ＝ 起動に最低限必要なconfig項目 ＋ 下表で「enabled」の非独立config項目

  である。

非独立Config項目の組み合わせパターン（i386）

		組み合わせパターン名												
		up	smp	small	cc_size	silent	module	procfs	sysctl	preempt	hotplug	pnp	pm	
非独立Config項目	SMP	×	●	×	×	×	×	×	×	×	×	×	×	-
	CC_OPTIMIZE_FOR_SIZE	×	×	●	●	×	×	×	×	×	×	×	×	-
	PRINTK	●	●	×	●	×	●	●	●	●	●	●	●	-
	BUG	●	●	×	●	×	●	●	●	●	●	●	●	-
	KMOD	●	●	×	●	●	×	●	●	●	●	●	●	Ref.1
	PROCFS	●	●	×	●	●	●	×	●	●	●	●	●	-
	SYSCTL	●	●	×	●	●	●	●	×	●	●	●	●	-
	PREEMPT	●	●	×	●	●	●	●	●	×	●	●	●	-
	HOTPLUG	●	●	×	●	●	●	●	●	●	×	●	●	-
	PNP	●	●	×	●	●	●	●	●	●	●	×	●	-
	PM	●	●	×	●	●	●	●	●	●	●	●	×	-

● : ENABLED
× : disabled

Ref.1　CONFIG_KMOD は depends on CONFIG_MODULES のため、CONFIG_KMOD=b と設定すれば CONFIG_MODULES も自動的に enable になる。

5. 測定準備 ～baseカーネルの検討～ (3)

- **非独立項目の組み合わせパターン (ARM)**

今回の検証では、下表に示す複数の非独立config項目組み合わせパターンを使用した。
各パターンのbaseカーネルでenableにされるconfig項目
＝ 各パターン用の測定対象configリストで「=b」とマークされるconfig項目
＝ 起動に最低限必要なconfig項目 ＋ 下表で「enabled」の非独立config項目
である。

非独立Config項目の組み合わせパターン (ARM)

		組み合わせパターン名												
		up	smp	small	cc_size	silent	module	procfs	sysctl	preempt	hotplug	pnp	pm	
非独立Config項目	SMP	×		×		×		×	×	×	×			Ref.1
	CC_OPTIMIZE_FOR_SIZE	●		●		●		●	●	●	●			Ref.2
	PRINTK	●		×		×		●	●	●	●			Ref.3
	BUG	●		×		×		●	●	●	●			-
	KMOD	×		×		×		×	×	×	×			Ref.1
	PROCFS	●		×		●		×	●	●	●			-
	SYSCTL	●		×		●		●	×	●	●			-
	PREEMPT	●		×		●		●	●	×	●			-
	HOTPLUG	●		×		●		●	●	●	×			-
	PNP	×		×		×		×	×	×	×			Ref.1
	PM	×		×		×		×	×	×	×			Ref.1

● : ENABLED
× : disabled

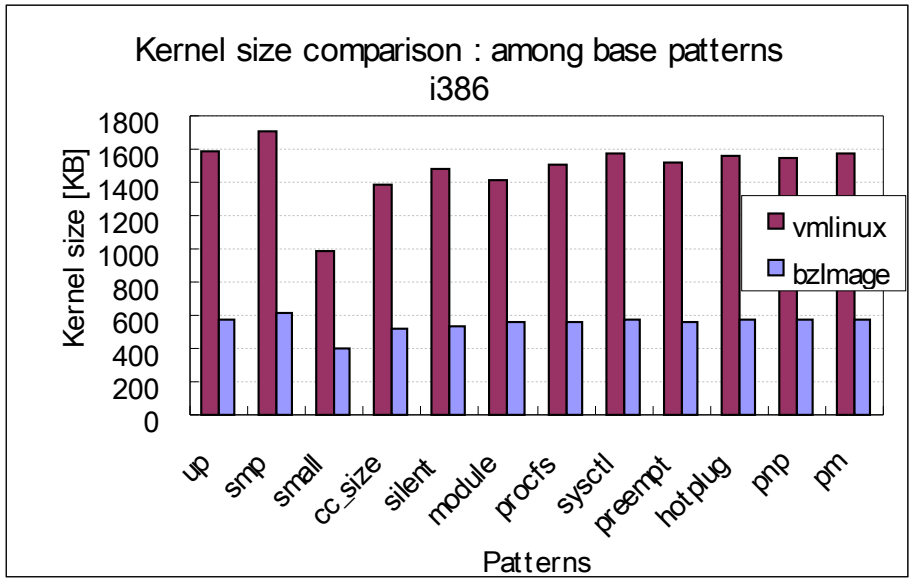
Ref.1 CONFIG_SMP, CONFIG_KMOD, CONFIG_PNP, CONFIG_PM を enableにするとコンパイルできない。

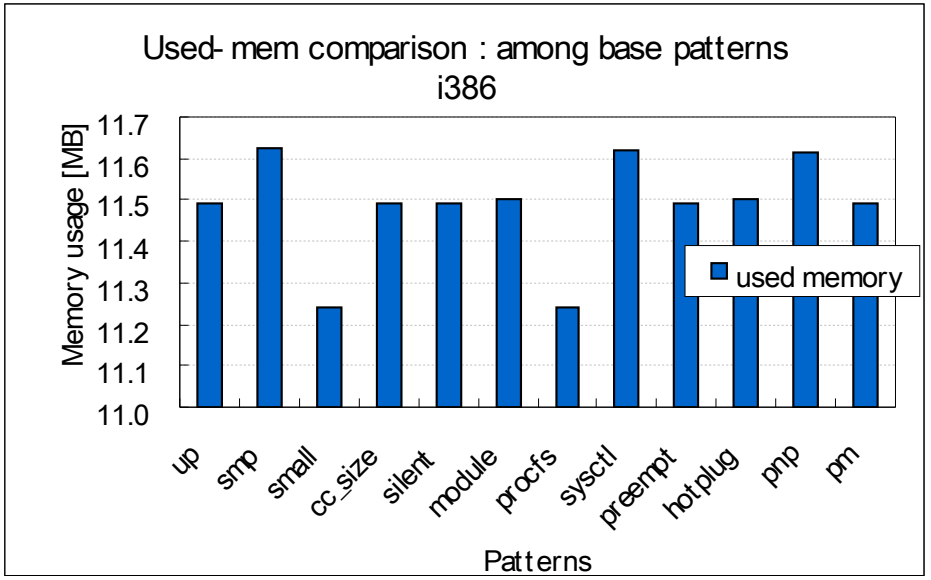
Ref.2 CONFIG_CC_OPTIMIZE_FOR_SIZE を disable にするとコンパイルできない。

Ref.3 CONFIG_PRINTK を disable にするとブート失敗。(6. (2)も参照)

6. これまでの測定結果 (1)

- **baseカーネルのサイズ・使用メモリ測定結果(i386)**
  - baseカーネルサイズ・使用メモリ量を非独立項目の組み合わせパターン間で比較。
  - 測定条件
    - Kernel : 2.6.12.3
    - gcc : 3.3.2 / binutils : 2.15
    - Target machine : Pentium IV 2.2GHz, RAM 256MB, HD 40GB

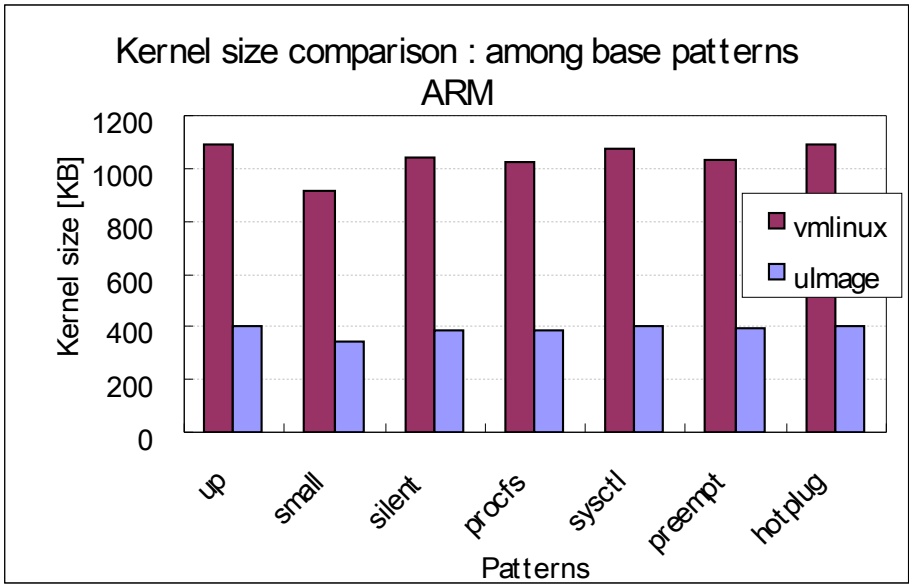


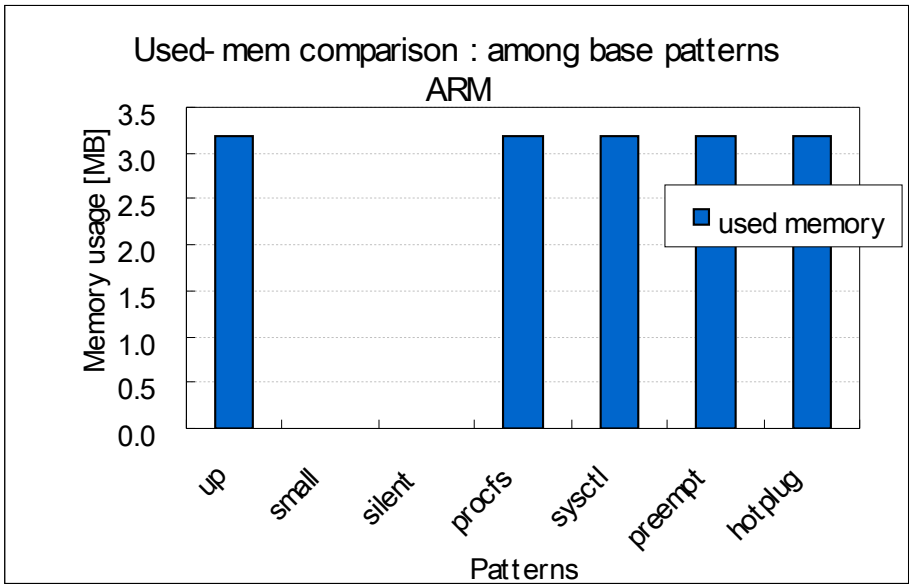


- **smallパターンの bzImageサイズ ＝ 400KB**
  ・・・ 2.6.12(i386)カーネルの限界最小サイズ
- **procfsパターンの消費メモリ量が少ない**
  ・・・ procfs の消費メモリが大きい。

6. これまでの測定結果 (2)

- **baseカーネルのサイズ・使用メモリ測定結果(ARM)**
  - baseカーネルサイズ・使用メモリ量を非独立項目の組み合わせパターン間で比較。
  - 測定条件
    - Kernel : 2.6.12.3
    - gcc : 3.3.2 / binutils : 2.15
    - Target machine : OSK5912 (OMAP5912(ARM926EJ-S) 192MHz, RAM 32MB, Flash 32MB)

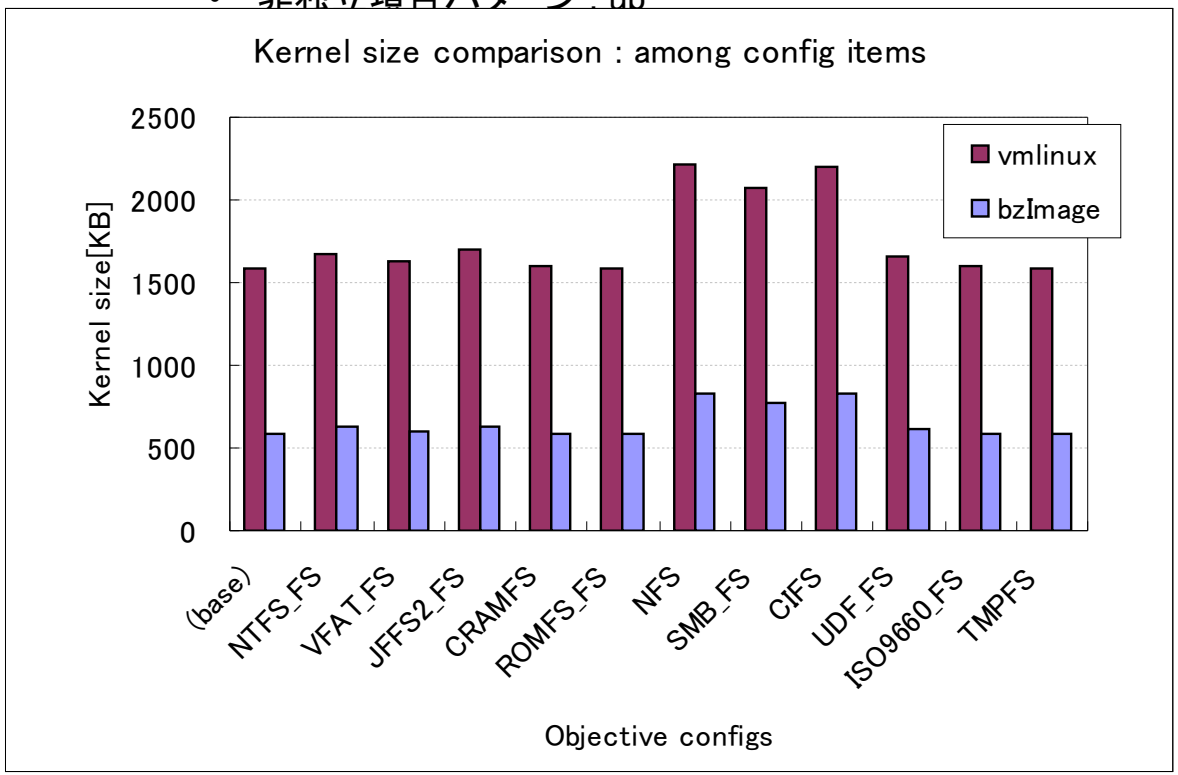




- **smallパターンの uImage サイズ ＝ 340KB**
  ・・・ 2.6.12(ARM)カーネルの限界最小サイズ
- **メモリ消費量はどのパターンもあまり変わらず。**
- **printkサポートを disable にするとブートせず。(small, silent パターン)**
  kernel/printk.c : console_setup() を定義すれば解決すると思うが・・・未検証。
  (http://www.selenic.com/pipermail/linux-tiny/2005-August/000216.html)

6. これまでの測定結果 (3)

- **Config項目間比較**
  - 各種fsのサイズを比較。
  - 測定条件
    - Kernel : 2.6.12.3
    - gcc : 3.3.2 / binutils : 2.15
    - Target machine : Pentium IV 2.2GHz, RAM 256MB, HD 40GB
    - 非独立項目パターン : up



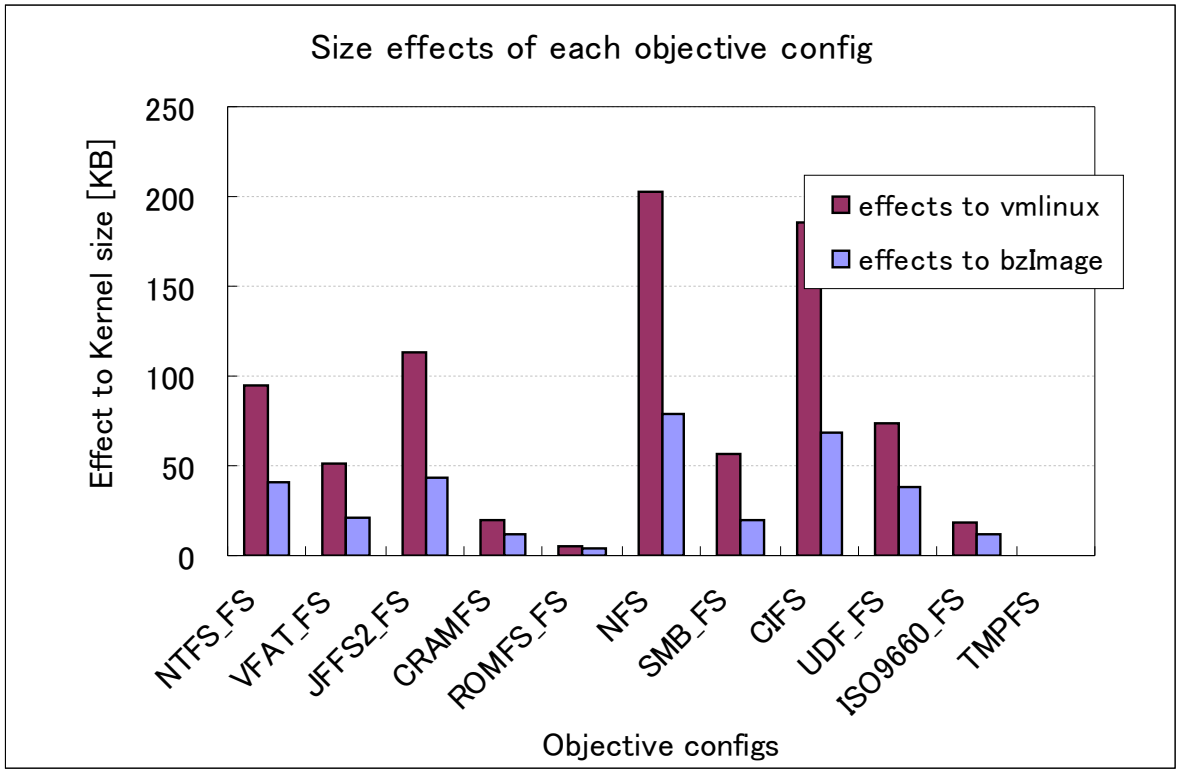
測定対象項目	実際にenableになったConfig項目
(base)	(base)
NTFS_FS	NTFS_FS+NLS
VFAT_FS	VFAT_FS+NLS+FAT_FS
JFFS2_FS	JFFS2_FS+CRC32+MTD
CRAMFS	CRAMFS+ZLIB_INFLATE
ROMFS_FS	ROMFS_FS
NFS	NFS_FS+SUNRPC+LOCKD+INET+NET
SMB_FS	SMB_FS+NLS+INET+NET
CIFS	CIFS+NLS+INET+NET
UDF_FS	UDF_FS
ISO9660_FS	ISO9660_FS
TMPFS	TMPFS

- **NFS、CIFS、SMBFSのサイズが大きい**

… ように見えるが、各カーネルは、依存性解決によってenableにされた測定対象項目以外のconfig項目も含んでいる。(次ページへ)

6. これまでの測定結果 (4)

- **Config項目間比較（つづき）**
  - 依存性解決によってenableになったconfig項目による増分をできる限り差し引き、測定対象項目単独での増分を算出した結果が以下。



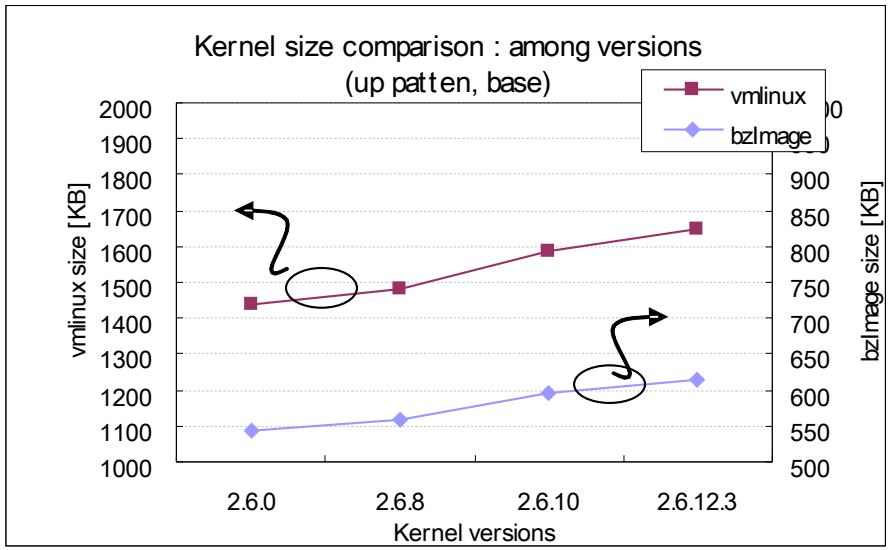
測定対象項目	増分を算出したConfig項目
NTFS_FS	NTFS_FS
VFAT_FS	VFAT_FS+FAT_FS
JFFS2_FS	JFFS2_FS
CRAMFS	CRAMFS+ZLIB_INFLATE
ROMFS_FS	ROMFS_FS
NFS	NFS_FS+SUNRPC+LOCKD
SMB_FS	SMB_FS
CIFS	CIFS
UDF_FS	UDF_FS
ISO9660_FS	ISO9660_FS
TMPFS	TMPFS

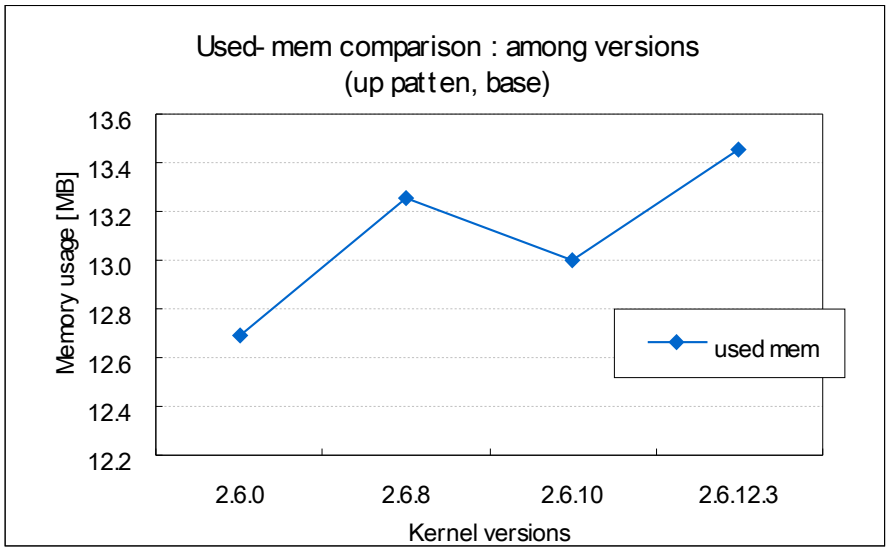
- **単独でのサイズが大きいのは、CIFS、JFFS2、NTFS。**

… 依存性解決によってenableになったconfig項目の測定結果がないものは、測定対象項目単独での増分が算出できていない。（上の例では VFAT と CRAMFS, NFS）

6. これまでの測定結果 (5)

- **バージョン間比較(1)**
  - カーネル2．6．xのサイズ・使用メモリをバージョン間で比較。
  - upパターンのbaseカーネルを使用。
  - 測定条件
    - gcc : 3.3.2 / binutils : 2.15
    - Target machine : Pentium IV 2.2GHz, RAM 256MB, HD 40GB
    - 非独立項目パターン : up



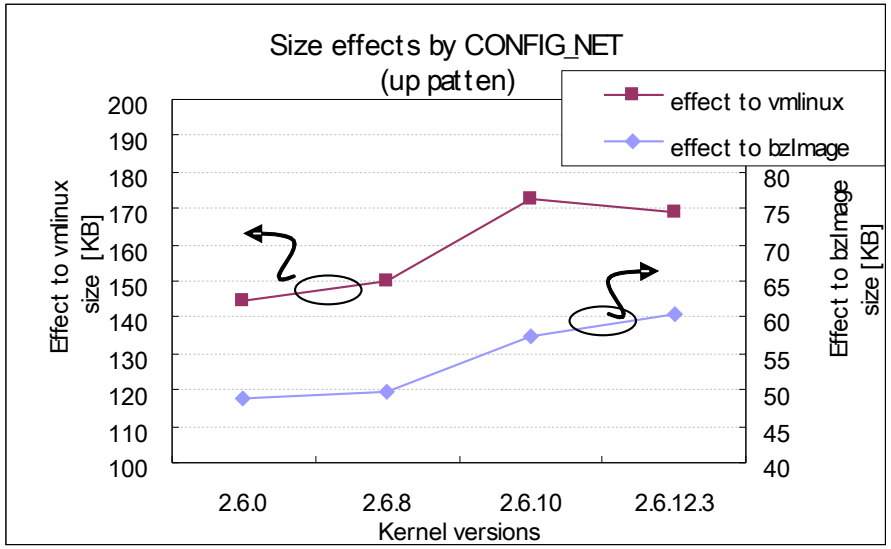


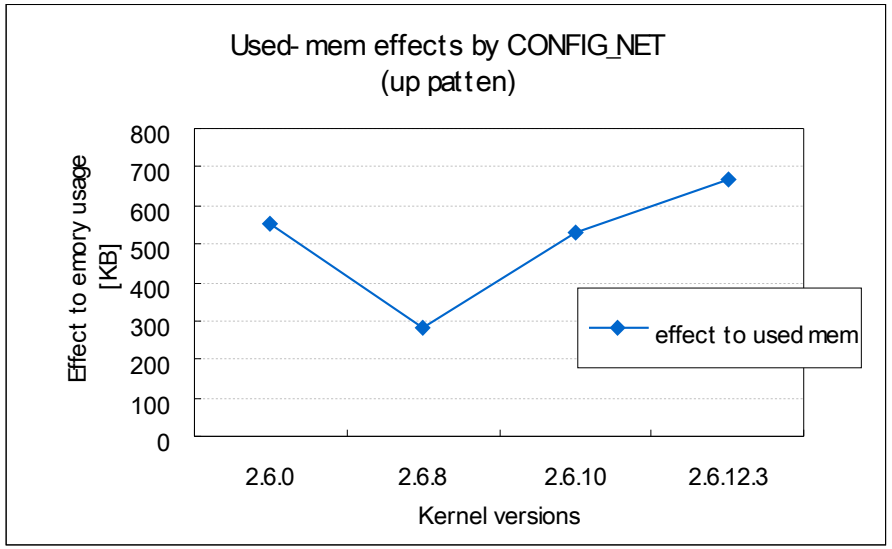
- **カーネルは順調に育っている。**

  2．6．0～2．6．12間に、
  vmlinux :200[KB]、bzImage:100[KB]、使用メモリ:750[KB]
  程度の成長．．．サイズ肥大化傾向が明らかになった。

6. これまでの測定結果 (6)

- **バージョン間比較(2)**
  - CONFIG_NET 単独のサイズ・使用メモリをバージョン間で比較。
  - 測定条件
    - gcc : 3.3.2 / binutils : 2.15
    - Target machine : Pentium IV 2.2GHz, RAM 256MB, HD 40GB
    - 非独立項目パターン : up



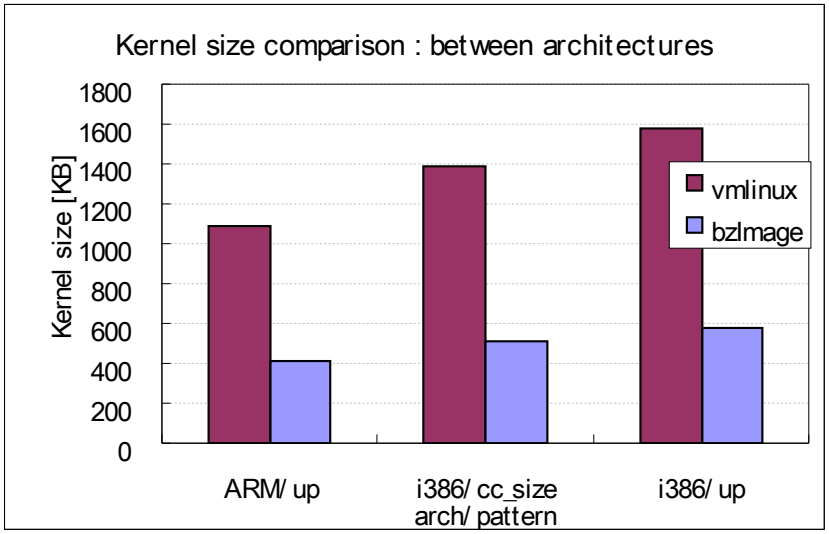


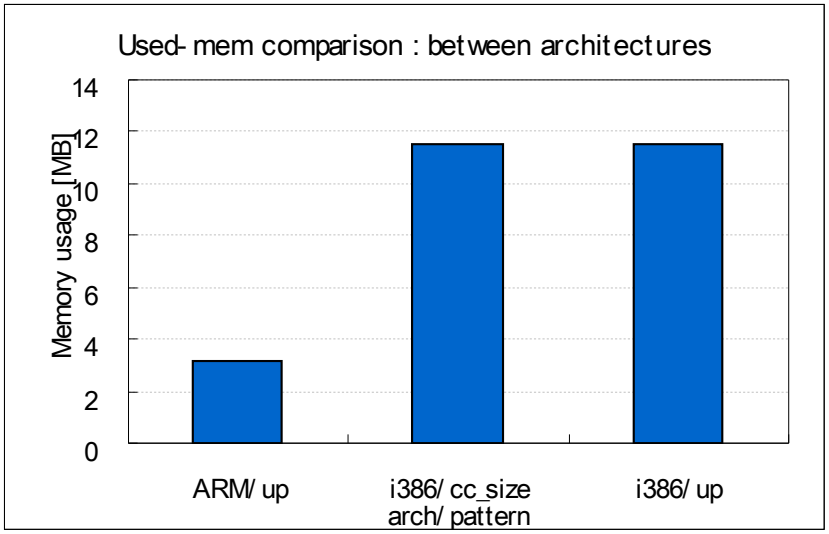
- **CONFIG_NET に関連したコードサイズは徐々に大きくなっている。**

  2．6．0～2．6．12間に、
  vmlinux：25[KB]、bzImage：10[KB] 程度の増加。
- **使用メモリは一旦減少した後再び増加傾向にある。**

6. これまでの測定結果 (7)

- アーキテクチャ間比較 (1)
  - i386/ARMのカーネルサイズ・使用メモリを比較。
  - 測定条件
    - Kernel : 2.6.12.3
    - gcc : 3.3.2 / binutils : 2.15
    - Target machine :
      - ARM : OSK5912 (OMAP5912(ARM926EJ-S) 192MHz, RAM 32MB, Flash 32MB)
      - i386 : PC (Pentium IV 2.2GHz, RAM 256MB, HD 40GB)
    - 非独立項目パターン :
      - ARM : up
      - i386 : up, cc_size ... ARMはupパターンでCONFIG_CC_OPTIMIZE_FOR_SIZE が enable なため。

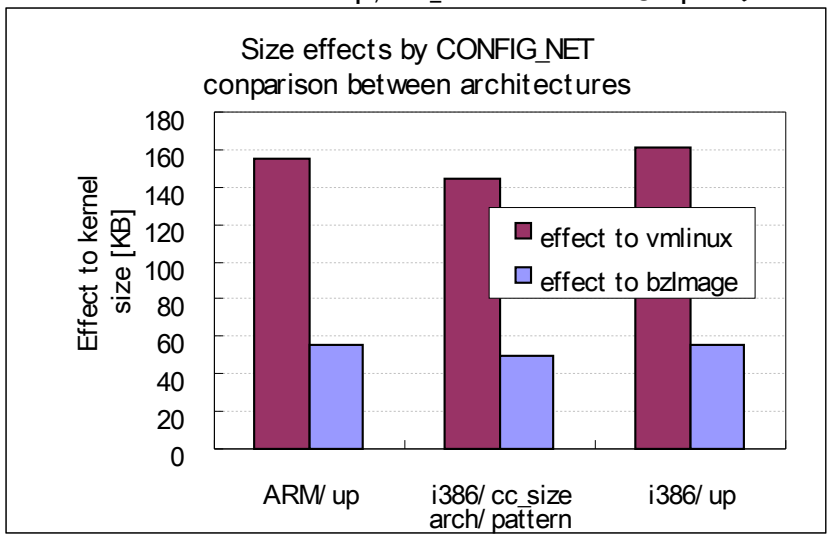


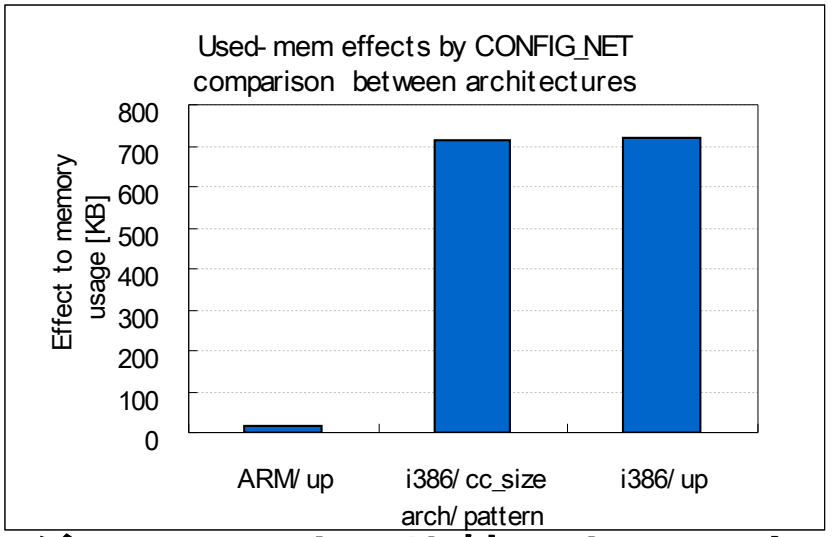


- **カーネルサイズ・使用メモリともに、ARMの方が小さい。**
  (ただし、i386はIDE, PCIドライバ等を含むため直接の比較はできない)
- **ARMの使用メモリが極端に少ない理由は・・・?**

6. これまでの測定結果 (8)

- アーキテクチャ間比較 (2)
  - i386/ARMの、CONFIG_NET単独でのカーネルサイズ・使用メモリ増分を比較。
  - 測定条件
    - Kernel : 2.6.12.3
    - gcc : 3.3.2 / binutils : 2.15
    - Target machine :
      - ARM : OSK5912 (OMAP5912(ARM926EJ-S) 192MHz, RAM 32MB, Flash 32MB)
      - i386 : PC (Pentium IV 2.2GHz, RAM 256MB, HD 40GB)
    - 非独立項目パターン :
      - ARM : up
      - i386 : up, cc_size ... ARMはupパターンでCONFIG_CC_OPTIMIZE_FOR_SIZE が enable なため。

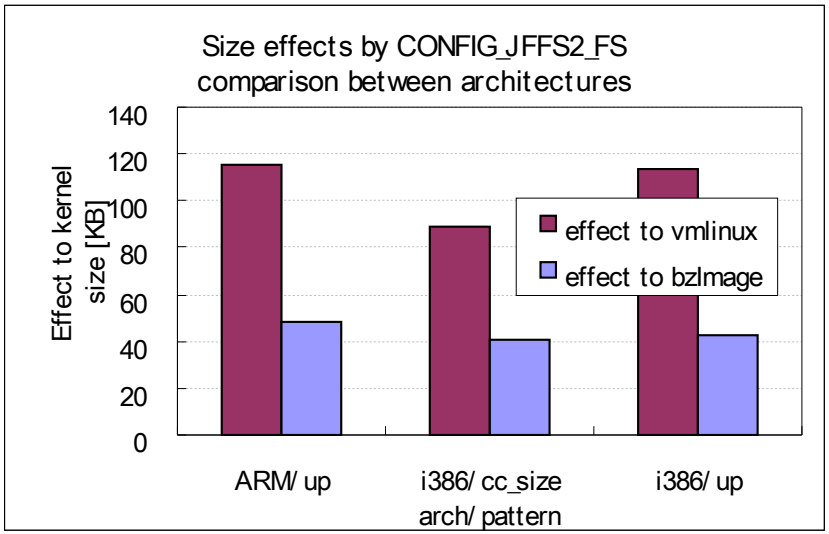


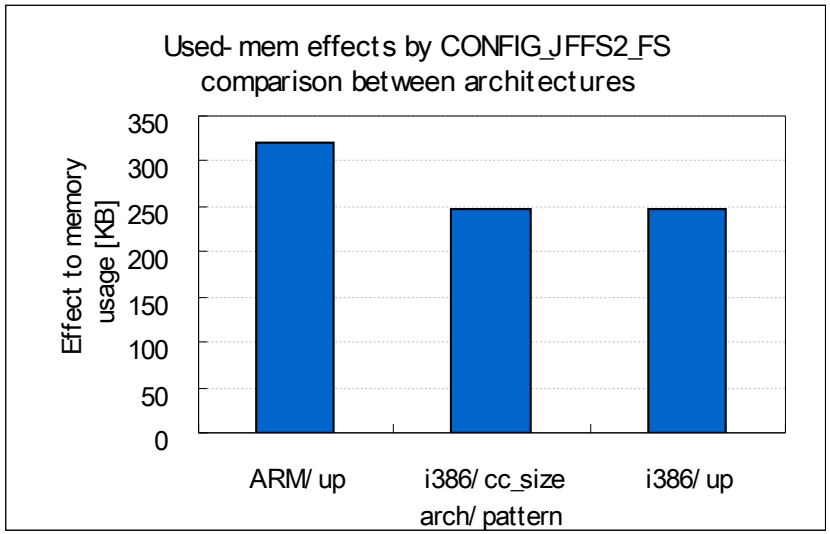


- CONFIG_CC_OPTIMIZE_FOR_SIZE が enable同士の比較では、ARMカーネルのサイズ増分が若干大きい。(RISC/CISCの差と推測される)
- ARMの使用メモリ増分が極端に少ない理由は・・・?

6. これまでの測定結果 (9)

- アーキテクチャ間比較 (3)
  - i386/ARMの、CONFIG_JFFS2_FS単独でのカーネルサイズ・使用メモリ増分を比較。
  - 測定条件
    - Kernel : 2.6.12.3
    - gcc : 3.3.2 / binutils : 2.15
    - Target machine :
      - ARM : OSK5912 (OMAP5912(ARM926EJ-S) 192MHz, RAM 32MB, Flash 32MB)
      - i386 : PC (Pentium IV 2.2GHz, RAM 256MB, HD 40GB)
    - 非独立項目パターン :
      - ARM : up
      - i386 : up, cc_size ... ARMはupパターンでCONFIG_CC_OPTIMIZE_FOR_SIZE が enable なため。





- **CONFIG_CC_OPTIMIZE_FOR_SIZE が enable同士の比較では、カーネルサイズ増分・メモリ使用量増分ともにARMカーネルが若干大きい。(RISC/CICSの差と推測される)**

6. これまでの測定結果 (10)

- **測定結果まとめ**
  - カーネル2.6.12での限界最小サイズ：
    - i386　：vmlinux = 920KB , bzImage = 400KB
    - ARM　：vmlinux = 900KB , uImage = 340KB
      ※ i386カーネルはARMカーネルにはないIDE, PCIドライバ等を含んでいるため、両者の直接比較はできないことに注意。
  - 限界最小メモリ消費量については結論出ず。
    - i386/ARM間のメモリ消費量差の原因が不明。
    - ARMでは printkサポート disable 時にブート不能。
  - 検証したfsの中では、CIFSのサイズ寄与が最大。続いてJFFS2。
    - CIFS　：vmlinux 190KB, bzImage 70KB
    - JFFS2：vmlinux 110KB, bzImage 40KB
      (いずれもi386/up パターンでの結果)
  - 2.6系列内で、カーネルサイズの肥大化傾向が明らかになった。
    - 2.6.0～2.6.12の間に、
      vmlinux 200KB, bzImage 100KB, 使用メモリ 750KB程度の肥大化。
      (いずれもi386/up パターンでの結果)
  - i386/ARM間で同一config項目の単独サイズを比較すると、ARMカーネルの方が若干大きい傾向が見られる。(RISC/CICSの差と推測される)

7. 今後の予定

- **実装・機能のブラッシュアップ**
  - カーネルサイズ予測機能
    - 独立Config項目のサイズ増分を足し合わせることで、カーネルのサイズを.configファイルから予測できるのでは？
    - 予備検証の結果：
      - 154種の独立Config項目で検証。

Baseカーネルサイズ	1195.0 [KB]	(a)
独立Config項目の増分総和	3083.4 [KB]	(b)
予測カーネルサイズ	4278.4 [KB]	(c) = (a) + (b)
実際のカーネルサイズ	4082.9 [KB]	(d)
誤差	195.5 [KB]	(e) = (c) - (d)
	4.8%	(f) = (e)/(d)

➔それなりに使えるレベル？

  - ツール設定の簡素化
    - 現状では設定がとても複雑・・・
      （本資料では触れなかったが、動作させるために色々な設定が必要）
  - バグ修正
- **ツール公開・データの共有**
  - 2月初旬までにCELF WiKiにて現状の測定対象configリスト、測定結果を公開予定。
  - ツールは2月中に、同じくCELF WiKiにて公開予定。
  - SystemSize W/G ML を議論の場としたい。
  - CELF Test Lab.にて取得データの拡充を図りたい。

8. 議論

- **以下について、ご意見いただけないでしょうか？**
  - 「こんなデータがほしい」
  - 「こんな用途に使いたい」
  - etc. …
- **データ・ツールを公開した後、以下について検討・ご協力いただける方を募集します。（アナウンスは追って CELF SystemSize W/G ML にて）**
  - 検証のアプローチは正しいか？
  - baseカーネルの設定は妥当か？
  - 測定対象config項目は妥当か？
  - 実際にデータを取得していただける方
  - サイズ肥大化原因について、解析していただける方
  - サイズを縮小するためのパッチを書いていただける方
  - etc. …

Empowered by Innovation