5A-1

デジタル情報機器におけるメモリ削減の検討

大和 仁典† 矢野 啓二郎† 安井 啓介† 上床 克樹† 佐久間 毅† 島田 智文†
株式会社 東芝†

1 はじめに

近年のデジタル情報家電などの組込み機器では、ネットワーク機能をはじめとして、最新技術へのタイムリーな追従のために、Linux などのオープンソースを利用する例が少なくない。しかし、デジタル情報機器では製品コストを抑えるために搭載されるメモリ容量が制限されたものになる場合が多く、Linux を使用する場合において、メモリ使用量の削減が重要な課題となっている。

そこで本論文では、圧縮効率の高いファイルシステムとデマンドページングによるメモリ使用量の削減について検討する。

2 メモリ使用量の削減手法

2.1 圧縮効率の高いファイルシステム

デジタル情報機器ではメモリ上にファイルシステムイメージを置くことがあり、圧縮効率の高いファイルシステムを用いることでメモリの使用量の削減が可能である。以下、Linux にて使用できる圧縮ファイルシステムである cramfs[1] および squashfs[2]について説明する。

表 2.1 cramfs と squashfs の特徴

	cramfs	squashfs
ブロックサイズ	4KB	0.5KB～64KB
メタデータ圧縮	不可	可
フラグメントブロックの利用	不可	可
XIP	対応	非対応
デマンドページング	対応	対応

表 2.1 に cramfs、squashfs の特長についてまとめた。cramfs、squashfs 共にデータを一定の大きさのブロックを単位として圧縮するが、cramfs はブロックサイズが 4KB 固定であり squashfs は 0.5KB ～64KB の間で可変である。ブロックサイズを大きくするほど圧縮率が高くなる。また、squashfs ではメタデータ(ファイルシステムの管理情報)の圧縮、フラグメントブロック(ブロックサイズに満たないファイルをまとめたブロック)を用いることにより、さらに圧縮効率を高めている。

Reduction of the memory usage for the embedded devices
†Embedded System Platform
Development Department, CORE TECHNOLOGY CENTER,
TOSHIBA CORPORATION

2.2XIP とデマンドページング

アプリケーションの実行イメージをメモリ上に配置する手段として、XIP[3]とデマンドページングがある。XIP はファイルシステム上の実行イメージを直接実行する方法である。圧縮ファイルシステム上であっても圧縮せずに実行イメージをメモリ上に配置しておく必要があり、メモリ使用量が大きくなる。

一方、デマンドページングは実行時に必要になったときだけファイルシステムにアクセスし、実行イメージの一部を配置する方法である。圧縮ファイルシステム上にアプリケーションがある場合、使用しない実行イメージは圧縮されているため、XIP と比較してメモリ使用量が小さくなる。

デマンドページングは cramfs、squashfs のどちらでも利用可能であるが、XIP は cramfs でのみ利用可能である。本論文ではアプリケーションの動作が最も高速である cramfs と XIP を用いた構成(以下 cramfs(XIP))と最もメモリ削減効果の高い squashfs とデマンドページングを組み合わせた構成(以下 squashfs(DP))についての比較を行う。

2.3 メモリマップ

cramfs(XIP)および squashfs(DP)のメモリマップを図 2.3 に示す。

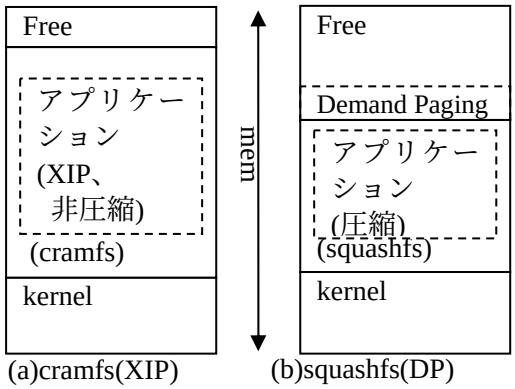

図 2.3 各構成のメモリマップ

(a)の構成では、実行速度を優先したいアプリケーションの実行イメージのみ XIP で配置している。一方で、(b)の構成に示すようにデマンドページングを用いる場合、メモリの空きの一部を実行イメージの配置のために用いる。

3 評価

3.1 メモリ削減量

メモリの空き容量を段階的に削減し、それに対するアプリケーションの正常動作を確認することでメモリ削減可能容量の評価を行った。本論文ではメモリ領域の大きさを示すパラメータ（カーネル起動時のパラメータ mem）を変更することでメモリの空き容量の削減を行った。

cramfs(XIP)、squashfs(DP)それぞれについて、削減可能容量を表 3.1 に示す。(本評価、および以降の全ての評価において、特に明記しない限り squashfs のブロックサイズは 64KB である。)

表 3.1 メモリ削減可能容量の見積もり

	cramfs(XIP)	squashfs(DP)
削減可能容量	13MB	19MB

表 3.1 より squashfs(DP)は cramfs(XIP)と比較して 6MB 削減できた。ただし、アプリケーション次第でこの値は変化する。

3.2 性能低下を考慮に入れたメモリ削減量

メモリ削減に伴う性能低下、及び性能改善について述べる。3.1 節と同様にしてメモリの空き容量を削減し、アプリケーションの操作に対する応答時間を測定した。

cramfs(XIP)、 squashfs(DP)のメモリの削減を行わない状態、およびメモリの空き容量を 13MB 削減した状態、および squashfs(DP)のメモリの空き容量を 16MB、19MB 削減した状態で評価を行った。図 3.2 に評価結果を示す。(図中 cramfs(XIP)および squashfs(DP))

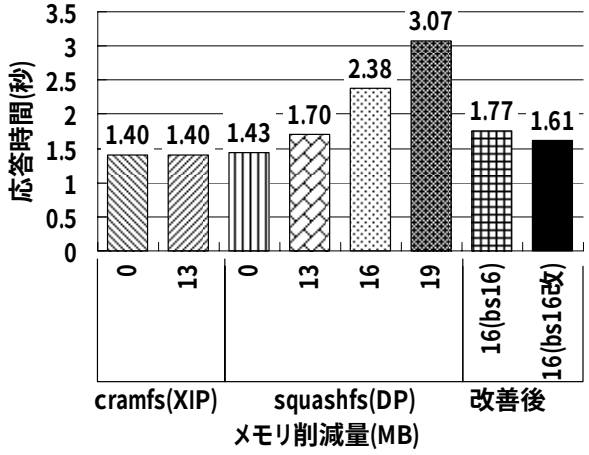

図 3.2 アプリケーションの実行性能

評価結果より、squashfs(DP)においてメモリの削減量が大きくなるほど大きく性能が低下した。squashfs ではブロック単位でページの圧縮、および伸張を行うため、ブロックサイズを大きくするほど余分な実行イメージの伸張を行う。そこでブロックサイズを小さくし、無駄な伸張処理を削減することで性能低下を緩和した。本論文ではブロックサイズを 16KB に変更して評価を行った。(図 3.2 の改善後、16(bs16))

また、squashfs のソースコードを調査したところ、圧縮データの読み出しにブロックデバイスドライバが介在していたため、メモリから直接読み出すように修正して評価を行った。メモリ空き容量の削減量は 16MB、ブロックサイズは 16KB である。(図 3.2 の改善後、16(bs16 改))

以上の 2 点の改善を行うことで、メモリの空き容量を 16MB 削減した場合に、改善前と比較して 0.8 秒程度アプリケーションの応答速度が早くなった。ただし、アプリケーション次第でこの値は変化する。

4 考察

squashfs(DP)においてブロックサイズを 64KB から 16KB へ変更した場合、余分な伸張処理が減少しただけでなく、必要な実行イメージが削除されにくくなったことで性能向上したと考えられる。特にメモリの削減量が大きい場合、デマンドページングに使用できる空きメモリが限られているため、ブロックサイズを小さくし、スラッシングの頻度を減少させることで大きく性能向上できる。

ただし、ブロックサイズを小さくするとファイルシステムイメージの圧縮率が低下し、デマンドページングに使用するメモリの空き容量が減少するため、トレードオフを考慮してブロックサイズを調整する必要がある。

5 まとめ

cramfs(XIP)を用いた場合、図 3.2 の評価結果より本論文で用いたアプリケーションでは性能低下を引き起こさずにメモリの削減が可能である。一方で squashfs(DP)は cramfs(XIP)よりメモリの削減において有利であるが、メモリの削減に伴って性能低下を引き起こす。ただし、ブロックサイズの調整や squashfs の改善を行うことで、対象とするアプリケーションの実行性能を改善することができる。

参考文献

[1] Linux カーネル付属ドキュメント
(http://www.kernel.org)
[2] squashfs (http://squashfs.sourceforge.net/)
[3] ApplicationXIP
(http://tree.celinuxforum.org/CelfPubWiki/ApplicationXIP)